Research Article

Satellite Mission Instruction Sequence Generation Algorithm Using a Flexible Weighted Directed Graph

Zhang Yahang,1,2 Zhang Jiakai,3 Yang Mengfei,1 Yang Peiyao,2 and Song Xiangshuai3

1College of Computer Science and Technology, Xidian University, Xi’an 710071, China
2Beijing Institute of Spacecraft System Engineering, Beijing 100094, China
3College of Aerospace and Civil Engineering, Harbin Engineering University, Harbin 150001, China

Correspondence should be addressed to Zhang Yahang; zhangyahang@163.com

Received 9 August 2021; Accepted 16 September 2021; Published 20 October 2021

Academic Editor: Jiafu Liu

Copyright © 2021 Zhang Yahang et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

A satellite mission instruction generation algorithm based on a flexible weighted directed graph has been proposed. This algorithm can sort the instruction into sequence according to the instruction execution relationship with constraints, which can promote satellite operation performance of the ground station. Concepts like flexible edge, flexible zone, and implement zone were introduced, and the flexible weighted directed graph (FWDG) model was proposed. Based on this model, the satellite instruction sequence generation algorithm was designed. After practice and research, the result showed that according to the new algorithm, the instruction sequence can be consistent with all implement zones, consequently guaranteeing the correctness of the sequence.

1. Introduction

A low earth orbit remote sensing satellite with high resolution is the main channel to acquire ground information, which plays an increasingly important role in an earth optical and microwave survey [1]. A remote sensing satellite meets the users’ needs through “task planning-instruction generation” [2]. The missions of a low earth orbit remote sensing satellite are intermittent, diverse, and uncertain, because of orbit revisit characteristics, weather conditions, onboard electronic constraints, and ground receiving resources. It is one of the most concerned problems for users and also a research hotspot in the academic field that a large space system takes advantage of a mission planning system to optimize the satellite-ground resources and accomplish the most imaging tasks with the least resources [3].

Lots of satellite task planning algorithms were presented based on artificial intelligence techniques [4]. Bensana et al. established the constraint satisfaction model and the integer programming model [5] and proposed the branch and bound algorithm and taboo search approximate algorithms to obtain the feasible solution. Hall and Magazine used a greedy algorithm and dynamic programming technique to solve a task programming model [6]. Vasquez and Hao map the SPOT5 satellite routine mission planning problem to a knapsack problem. Lin et al. use a Joseph-Louis Lagrange relaxation technique and tabu search algorithm to solve the imaging scheduling problem [7]. Mansour and Dessouky design the evolutionary algorithm and compare it with the traditional algorithm [8]. The main problem with these methods is that scheduling decisions cannot be changed dynamically. In References [9–11], the method of dynamically adjusting the results of planning such as task replanning is proposed, but the deviation between task planning and task execution cannot be considered.

On this basis, the satellite instruction sequence generation algorithm based on the weighted directed graph (WDG) presented in References [12, 13] can sort the satellite instruction sequence according to the constraint relation, and the operation efficiency of the ground control center to the spacecraft is greatly reduced. However, the interval between the instruction nodes of the noncritical path may
be extended by default when the topological sorting algorithm is applied to generate the instruction sequence; it does not consider the need for strict protection of the execution interval of some spacecraft instructions.

In this paper, the concepts of directed acyclic flexible edge, flexible interval, and reasonable instruction execution interval of a directed acyclic graph are introduced, and a flexible weighted directed graph (FWDG) model is proposed to modify the instruction generation algorithm, and an instruction sequence generation algorithm based on the flexible weighted digraph is proposed. The FWDAG model is an essential module in onboard software, and it takes observation mission as input and instruction sequence as output, as shown in Figure 1. The example shows that the modified algorithm can satisfy the spacecraft instruction execution interval.

2. Satellite Instruction Sequence Generation Algorithm Based on the Directed Graph Model

2.1. Directed Graph Model. The satellite mission instruction sequence is transformed into a digraph model $G(V, E)$ consisting of instructions and their constraints. Among them,

$(1)$ Graph $G = (V, E)$: source vertex represents the start of the task; sink vertex represents the end of the task

$(2)$ $V = \{V_1, V_2, V_3, \ldots, V_N\}$ is the set of vertices of a directed graph; each vertex $V_j (j = 1, 2, \ldots, N)$ corresponds to an instruction. $\text{TIME}(V_j)$ represents the execution time of the instruction

$(3)$ $E = \{E_1, E_2, \ldots, E_M\}$ is the set of edges of a directed graph; edges $E_i = \{V_j \rightarrow V_k\} (i = 1, 2, \ldots, M; j, k = 1, 2, \ldots, N)$ represent instruction $V_j$ preceding instruction $V_k$; $V_j$ is the forward vertex of $V_k$, and $V_k$ is the backward vertex of $V_j$; the length of the edge $L_i$ represents the execution time interval between two instructions

In graph $G$, $V$ is the set of vertices corresponding to all instructions contained in the task and $E$ is the edge set corresponding to the instruction timing constraint that completes the task. In order to describe $G(V, E)$ in detail, more symbols are introduced; they are as follows:

$N$ is the number of all vertices of a digraph $G(V, E)$; $M$ is the number of all edges of a digraph $G(V, E)$. Source is the start of the instruction sequence for a task.

Sink is the end of the instruction sequence for a task.

$V(T_0)$ is the instructions at time $T_0$ ($T_0$ is the beginning time of imaging).

$V(T_1)$ is the instructions at time $T_1$ ($T_1$ is the beginning time of data transfer).

$X = \{S, X_1, X_m, \ldots \}$, where $S$ is the start time of a mission and $\{X_1, X_m, \ldots \}$ is the interval between the instruction sequence $\{V_1, V_m, \ldots \}$ and the first instruction.

2.2. Task Instruction Sequence Generation Algorithm Based on the Digraph Model. In this section, we introduce the directed graph-based task instruction sequence generation algorithm [12]:

A unified instruction topological sequence can be generated finally by Algorithm 1, after digraph modeling of a sensing task, according to the start time of the mission on the satellite, the interval of the basic instruction sequence, and the dependence of different instruction sequences.

However, in the second and fourth steps of the algorithm, when the earliest $\text{EST}(V_j)$ and the latest $\text{LST}(V_j)$ of each vertex are calculated, the default edge can be stretched [14, 15].
1. Hierarchizing the graph $G(V, E)$ according to the following definition.
   (i) Define $Source$ as 0 − layer vertex;
   (ii) 1 − layer vertices only take the 0 − layer vertices as forward vertices;
   (iii) 2 − layer vertices only take the 1 − layer vertices or $Source$ as forward vertices;
   (v) ... 
   (vi) $n$ − layer vertices take $Source$, 1 − layer, ..., $n$ − 1 − layer vertices as forward vertices.

2. Start with $Source$, the earliest execution time interval $EST(V_i)$ of each vertex $V_i$ relative to $Source$ is computed by layer.
3. The earliest execution time interval of $Sink$ relative to $Source$ is the minimum execution time of the task instruction sequence.
4. Starting from $Sink$, the latest execution time interval $LST(V_i)$ of each vertex relative to $Source$ is calculated in reverse step by step.
5. For instruction $V_i$, like shutdown, the execution time delay $X_p$ selects the earliest execution time interval relative to $Source$, and for instruction $V_i$, like switch on, the execution time delay $X_p$ selects the latest execution time interval relative to $Source$; Ensure that the command sequence satisfies the user’s task, starting as late as possible and shutting down as early as possible.
6. The execution time of the $Source$ vertices is the time $T_0$ subtracts the earliest execution time of the $Source$ to $T_0$, which is also the start time $S$ that the sequence starts to execute.
7. ...

**Algorithm 1: Instruction sequence generation algorithm based on the directed acyclic graph model.**

![Instruction model of the general weighted directed acyclic graph.](image1.png)

![Instruction model of the general weighted RE acyclic graph.](image2.png)

According to the algorithm in [12, 13], for the model shown in Figure 2, if $T(V_0) = 0$, then

$$
T(V_0) = 0,
$$

$$
EST(V_1) = 5,
$$

$$
EST(V_2) = 7,
$$

$$
EST(V_3) = \max \{EST(V_1) + E_3, EST(V_2) + E_4\} = 15,
$$

$$
LST(V_3) = EST(V_3) = 15,
$$

$$
LST(V_1) = LST(V_3) - E_3 = 11,
$$

$$
LST(V_2) = LST(V_3) - E_4 = 7,
$$

$$
LST(V_0) = \min \{LST(V_1) - E_1, LST(V_2) - E_2\} = 0.
$$

(1)

Obviously, for vertices $V_2$ and $V_1$, we have $EST(V_1) \neq LST(V_1)$ and $EST(V_2) \neq LST(V_2)$.

Algorithm 1 holds the execution time $T(V_i)$ and $T'(V_i)$ for $V_i$ and $V_3$, respectively, which is reasonable, as long as $T(V_i) \in [EST(V_i), LST(V_i)]$ and $T'(V_i) \in [EST(V_i), LST(V_i)]$. If $V_1$ is the shutdown instruction, according to Algorithm 1, $T'(V_1) = EST(V_1) = 5$. But consider an application scenario in which there are certain instructions and intervals between instructions that are fixed; that is, some edges $E_i$ cannot be stretched and shortened. In this paper, we call this “rigid edge,” denoted as RE.

As shown in Figure 3, the solid line $RE_3$ indicates that the edge cannot be neither shortened nor stretched, and the dash line indicates that the edge can be stretched but cannot be shortened. Obviously, $T(V_i)$ must be a fixed value; $T'(V_3) = T(V_3) - 4 = 11$. However, Algorithm 1 cannot take into account this constraint, and the calculated results cannot meet this requirement.

**3. Instruction Generation Algorithm Based on the FWDAG Instruction Model**

In order to meet the more realistic and strict requirements of a satellite instruction transmission time slot, we propose an instruction generation algorithm based on a flexible weighted directed acyclic graph (FWDAG).

**3.1. FWDAG Instruction Model.** For the more general case, some instruction intervals can be within a set interval. For example, instruction $V_1$ must be executed after instruction $V_0$ is completed 3 − 4 s; then, obviously the interval $[3, 4]$ is the variable interval of $E_1$. In this case, the concept of flexible interval edges is introduced into weighted directed acyclic graphs. The basic definitions are listed below:
Flexible Edge. In a directed acyclic graph, the edges with variable weights are called flexible edges and are denoted as FE.

Flexible Zone. The flexible interval of edge weight, which we call the flexible zone, is denoted as FZ. It is obvious that the rigid edge is a special flexible edge whose upper and lower limits of the flexible interval are equal.

Implemet Zone. In a directed acyclic graph, the time of a node changes within a reasonable range, which becomes a reasonable range of execution, referred to as the implement zone, recorded as IZ.

Addition operation of IZ and FZ: the addition operation of the implement zone and the flexible zone is defined as follows:

\[
IZ = IZ + FZ = [IZ_{down} + FZ_{down}, LZ_{up} + FZ_{up}].
\]

(2)

Subtraction operation of IZ and FZ: the subtraction operation of the implement zone and the flexible zone is defined as follows:

\[
IZ' = IZ - FZ = [IZ_{down} - FZ_{up}, LZ_{up} - FZ_{down}] I(0, +\infty).
\]

(3)

Obviously, if \( LZ_{up} < FZ_{down} \), \( IZ = \emptyset \).

In view of the above definition, the following theorem is proposed and proved.

**Theorem 1.** Suppose that the node \( V_i \) in a directed acyclic graph has a precursor node \( V_{i}^{pre} \), the corresponding flexible edge is \( FE_{i}^{pre} \), and the flexible zone is \( FZ(\widehat{FE}_{i}) = [FZ(\widehat{FE}_{i})_{down}, FZ(\widehat{FE}_{i})_{up}] \). If \( IZ(V_{i}) = [IZ(V_{i})_{down}, LZ(V_{i})_{up}] \) has been determined, then \( IZ(V_i) = IZ(V_{i}^{pre})_{down} + FZ(\widehat{FE}_{i})_{down}, LZ(V_{i}^{pre})_{up} + FZ(\widehat{FE}_{i})_{up} \).

**Proof.** Let \( x = T(V_{i}^{pre}), y = T(\widetilde{FE}_{i}), \) and \( z = T(V_i) \). According to the flexible edge constraints of \( \widetilde{FE}_{i} \), the objective function is \( z = x + y \); here, \( x \in [IZ(V_{i}^{pre})_{down}, IZ(V_{i}^{pre})_{up}] \) and \( y \in [FZ(\widehat{FE}_{i})_{down}, FZ(\widehat{FE}_{i})_{up}] \). According to the property of the linear equations with two unknowns, the range of the objective function \( z = x - y \) is \([IZ(V_{i})_{down} - FZ(\widehat{FE}_{i})_{down}, LZ(\widehat{FE}_{i})_{up} - FZ(\widehat{FE}_{i})_{up}]\), and the range is continuous within its domain of definition \( IZ(V_i) = IZ(V_{i}^{pre})_{down} + FZ(\widehat{FE}_{i})_{down}, LZ(V_{i}^{pre})_{up} + FZ(\widehat{FE}_{i})_{up} \). □

**Inference 2.** Suppose that a node \( V_i \) in a directed acyclic graph has \( J \) precursor nodes \( V_{i}^{pre} j = 1,2, \ldots, J \). One of the implement zones \( IZ(V_{i,j}) \) of node \( V_{i,j}^{pre} \) is determined by implement zone \( IZ(V_{i,j}^{pre}) \) of each precursor and flexible zone \( FZ(\widehat{FE}_{j}) \) of FE \( J \). All entries of this node determine the intersection of the execution ranges of this node, which is equal to the execution ranges of this node. That is, \( IZ(V_i) = \bigcap_{j=0}^{J} (IZ(V_{i,j}^{pre}) + FZ(\widehat{FE}_{j})) \).

**Theorem 3.** Suppose that the node \( V_i \) in a directed acyclic graph has a backward node \( V_{i}^{next} \), the corresponding flexible edge is \( FEnext \), and the flexible zone is \( FZ(FEnext) = [FZ(FEnext)_{down}, FZ(FEnext)_{up}] \). If \( IZ(V_{i}^{next}) = [IZ(V_{i}^{next})_{down}, LZ(V_{i}^{next})_{up}] \) has been determined, then \( IZ(V_i) = [IZ(V_{i}^{next})_{down} - FZ(FEnext)_{down}, LZ(\widehat{FE}_{i})_{up} - FZ(FEnext)_{up}] \).

**Proof.** Let \( x = T(V_{i}^{next}), y = T(\widehat{FE}_{i}), \) and \( z = T(V_i) \). According to the flexible edge \( FEnext \) constraints, it has an objective function \( z = x - y \); here, \( x \in [IZ(V_{i}^{next})_{down}, IZ(V_{i}^{next})_{up}] \), and \( y \in [FZ(FEnext)_{down}, FZ(FEnext)_{up}] \). According to the property of the linear equations with two unknowns, the range of the objective function \( z = x - y \) is \([IZ(V_{i}^{next})_{down} - FZ(FEnext)_{down}, LZ(FEnext)_{up} - FZ(FEnext)_{up}]\), and the range is continuous within its domain of definition \( IZ(V_i) = [IZ(V_{i}^{next})_{down} - FZ(FEnext)_{down}, LZ(FEnext)_{up} - FZ(FEnext)_{up}] \). □

**Inference 4.** Suppose a directed acyclic graph has a node \( V_i \) and \( j \) rear-driver nodes \( V_{i}^{next} j = 1,2, \ldots, J \). Each rear-driver node \( V_{i}^{next} j \) implement zone \( IZ(V_{i}^{next} j) \) and the flexible zone \( FZ(\widehat{FE}_{j}) \) of FE determine an implement zone \( IZ(V_{i,j}) \) of node \( V_i \); all entries of this node determine the intersection of the execution ranges of this node, which is equal to the execution ranges of this node. That is, \( IZ(V_i) = \bigcap_{j=0}^{J} (IZ(V_{i}^{next}) - FZ(\widehat{FE}_{j})) \).

**3.2. Instruction Generation Algorithm Based on FWDAG.** In this paper, an instruction sequence generation algorithm based on the FWDAG model is proposed. A flexible edge FE is generated by the dependence of instructions with \( n \) nodes.

**4. Application Example.** Figure 4 is an instruction model of a satellite imaging task that is described by FWDAG. According to Algorithm 2, the procedure for calculating the graph above is

\[
IZ(V_0) = [0, 0], IZ(V_1) = [0, +\infty], IZ(V_2) = [0, +\infty], IZ(V_3) = [0, +\infty],
IZ(V_{1,1}) = [0, +\infty], IZ(V_{2,1}) = [0, +\infty], IZ(V_{end}) = [0, +\infty],
IZ(V_{1}) = IZ(V_0) + FZ(\widehat{FE}_1) = [5, 7],
IZ(V_{2}) = IZ(V_0) + FZ(\widehat{FE}_2) = [2, 3],
IZ(V_{3}) = IZ(V_0) + FZ(\widehat{FE}_3) = [2, 3],
IZ(V_{1,1}) = IZ(V_{1}) + FZ(\widehat{FE}_{1,1}) = [2, 3] \cup [5, 7] = [7, 10],
IZ(V_{2,1}) = IZ(V_{2}) + FZ(\widehat{FE}_{2,1}) = [2, 3] \cup [2, 9] = [4, 12],
IZ(V_{1,1})' = (IZ(V_{1}) + FZ(\widehat{FE}_{1,1})) \cap IZ(V_{1,1}) = [9, +\infty] \cap [4, 12] = [9, 12],
IZ(V_{end}) = IZ(V_{1,1}) + FZ(\widehat{FE}_{end}) = [7, 10] + [120, 145] = [127, 155],
IZ(V_{end})' = IZ(V_{end}) + FZ(\widehat{FE}_{end})' = [14, 22] \cap [127, 155] = \emptyset.
\]

(4)

There is no solution to the program break.

Obviously, according to the antenna path \( V_0 \rightarrow V_{1,1} \rightarrow V_{end} \), the earliest execution time of \( V_{end} \) is 127, and according to the camera path \( V_0 \rightarrow V_{2,1} \rightarrow V_3 \), the latest
1. The implement zone $IZ(V_0) = [0, 0]$ for the task starting node $V_0$. The implement zone for all other nodes is set to $IZ(V_i) = [0, +\infty)$ ;
2. Starting from $V_0$, select a node with degree 0 (no precursor node) from the digraph as the current node $V_i$. According to equation (2), calculate implement zone of all the backward nodes of the current node whose degree edges are connected.
3. According to Inference 2, the implement zone of the new node calculated by step 2 is updated by “And operation”. If the result of “And operation” is “Empty”, operation will be quit.
4. Deletes the current node and all flexible edges coming from this node.
5. Repeat steps 2 to step 4, until there are no more nodes in the directed graph.
6. Starting with $V_N$, select a node from the directed graph with an outdegree of 0 (without a trailing node) as the current node $V_i$, according to equation (3), select $T(V_i)$ from $IZ(V_i)$ according to the strategy of selecting node time (the last instruction is to select the earliest value, the start instruction is to be as late as possible, and the shutdown instruction is to be as early as possible), at the same time update the current node into all degrees edge connected to the precursor node implement zone;
7. According to Inference 4, the result of step 6 is updated by “And operation”. The result of “And operation” should not be “Empty”.
8. Deletes the current node and all flexible edges that enter this node.
9. Repeat steps 6 to step 8, until the node is no longer present in the directed graph.

**Algorithm 2:** An instruction sequence generation algorithm based on FWDAG.

**Figure 4:** FWDAG instruction model.

**Figure 5:** Adjusted FWDAG instruction model.
execution time of $V_{end}$ is 22; obviously, there is no solution. This means that there are conflicts among camera payload, attitude executable trajectory, and antenna executable path constraint. The period of antenna execution is very long, while the time of camera payload powered on cannot be too long. Hence, the camera standby instruction is inserted into the onboard system, to prolong the camera instruction path $V_0 \rightarrow V_{2.1} \rightarrow V_3$ and then to match the antenna path, as shown in Figure 5.

By the updated instruction model, according to Algorithm 2, recalculated results are

$$IZ(V_0) = [0, 0], IZ(V_1) = [0, +\infty), IZ(V_2) = [0, +\infty], IZ(V_3) = [0, +\infty],$$

$$IZ(V_{1.1}) = [0, +\infty], IZ(V_{2.1}) = [0, +\infty), IZ(V_{end}) = [0, +\infty],$$

$$IZ(V_1) = IZ(V_0) + FZ(FE_{1.1}) = [2, 3],$$

$$IZ(V_2) = IZ(V_0) + FZ(FE_{2.1}) = [2, 3],$$

$$IZ(V_3) = IZ(V_0) + FZ(FE_{3.1}) = [2, 3],$$

$$IZ(V_{1.1}) = IZ(V_1) + FZ(FE_{1.1}) = [2, 3] + [5, 7] = [7, 10],$$

$$IZ(V_{2.1}) = IZ(V_2) + FZ(FE_{2.1}) = [2, 3] + [2, 9] = [4, 12],$$

$$IZ(V_{2.1})' = (IZ(V_2) + FZ(FE_{2.1})) \cap IZ(V_{2.1})' = [9, +\infty) \cap [4, 12] = [9, 12],$$

$$IZ(V_{1.1}) = IZ(V_{1.1}) + FZ(FE_{1.1}) = [9, 12] + [5, 10] = [14, 22],$$

$$IZ(V_{end}) = IZ(V_{end}) + FZ(FE_{2.1.1.1}) \cap IZ(V_{end})' = [16, +\infty) \cap [127, 155] = [127, 155].$$

According to the end node selection of the earliest execution time principle, then

$$T(V_{end}) = 127, LZ(V_{end}) = [127, 127],$$

$$IZ(V_{1.1}) = (IZ(V_{end}) - FZ(FE_{1.1})) \cap IZ(V_{1.1}) = [0, 7] \cap [7, 10] = [7, 7], T(V_{1.1}) = 7,$$

$$IZ(V_1) = (IZ(V_{1.1})' - FZ(FE_{1.1})) \cap IZ(V_1) = [0, 2] \cap [2, 3] = [2, 2], T(V_1) = 2,$$

$$IZ(V_0) = (IZ(V_1)' - FZ(FE_{1.1})) \cap IZ(V_0) = [0, 0] \cap [0, 0] = [0, 0],$$

$$IZ(V_{2.1.1}) = (IZ(V_{end}) - FZ(FE_{2.1.1.1.1})) \cap IZ(V_{2.1.1}) = [0, 125] \cap [14, 22] = [14, 22].$$

(6)

According to the end node selection of the earliest execution time principle, then

$$T(V_{2.1.1}) = 14, LZ(V_{2.1.1})'' = [14, 14],$$

$$IZ(V_{2.1}) = (IZ(V_{2.1.1})' - FZ(FE_{2.1.1}) \cap IZ(V_{2.1}) = [4, 9] \cap [9, 12] = [9, 9], T(V_{2.1}) = 9,$$

$$IZ(V_2) = (IZ(V_{2.1})' - FZ(FE_{2.1}) \cap IZ(V_1) = [0, 7] \cap [2, 3] = [2, 3].$$

(7)

According to the end node selection of the earliest execution time principle, then

$$T(V_2) = 2, LZ(V_2)'' = [2, 2],$$

$$IZ(V_0) = (IZ(V_2)'' - FZ(FE_{2.1}) \cap IZ(V_0) = [0, 0] \cap [0, 0] = [0, 0],$$

$$IZ(V_3) = (IZ(V_2)'' - FZ(FE_{3.1}) \cap IZ(V_3) = [2, 2] \cap [0, 0] = [2, 2], T(V_3) = 2,$$

$$IZ(V_0)_3'' = (IZ(V_2)'' - FZ(FE_{3.1}) \cap IZ(V_0) = [0, 0] \cap [0, 0] = [0, 0],$$

$$IZ(V_0)'' = IZ(V_0)_1' \cap IZ(V_0)_2' \cap IZ(V_0)_3'' = [0, 0], T(V_0) = 0.$$

At last, all the $T$s were calculated:

$$T(V_0) = 0, T(V_1) = 2, T(V_2) = 2, T(V_3) = 2, T(V_{1.1}) = 7, T(V_{2.1.1}) = 9, T(V_{2.1.1.1}) = 14, and T(V_{end}) = 127,$$ and we get the satellite imaging mission’s command sequence.

The proposed scheme and algorithm are to model the existing constraints and find the feasible solution of the instruction sequence under this constraint model, which not only meets the constraint requirements but also improves the task execution energy efficiency as much as possible. If the feasible solution of the instruction sequence cannot be found under the current constraint model, the algorithm will alert the designer. As for how to modify the constraint model, such as the weight of edges or the number of vertices, designers need to adjust the design according to the actual situation of satellite missions and their own experience, so as to adjust the FWDAG model, and finally ensure that the instruction system can find a feasible solution under the adjusted constraint model.

5. Conclusion

An instruction generation algorithm based on the flexible weighted directed acyclic graph is proposed. By introducing the concept of a flexible edge into the directed acyclic graph, the problem that the time interval of instruction execution is fixed in a certain variable region is solved, ensuring that the execution interval of the dependent instruction is within the
preset range. The time complexity of the algorithm is $O(n) = O(n + 2e)$, where $n$ is the number of nodes in the digraph model and $e$ is the number of edges.

**Data Availability**

The data used to support the findings of this study are included within this paper.

**Conflicts of Interest**

The authors declare that there is no conflict of interest regarding the publication of this paper.

**Acknowledgments**

This work is supported in part by the National Natural Science Foundation of China (No. 11772185) and the project D020214.

**References**

[1] X. Wu, H. Zhao, B. Huang, J. Li, S. Song, and R. Liu, “Minimum-learning-parameter-based anti-unwinding attitude tracking control for spacecraft with unknown inertia parameters,” *Acta Astronautica*, vol. 179, pp. 498–508, 2021.

[2] R. He, *Research on Imaging Reconnaissance Satellite Scheduling Problem*, [Ph.D. thesis], National University of Defense Technology, 2004.

[3] J. S. Rodriguez, A. Y. Botero, D. Valle, J. G. Serna, and F. Botero, “Experimental approach for the evaluation of the performance of a satellite module in the CanSat form factor for in situ monitoring and remote sensing applications,” *International Journal of Aerospace Engineering*, vol. 2021, Article ID 868797, 28 pages, 2021.

[4] S. Mu, Y. Chen, and N. Zhang, “A rapid target searching scheme for a small satellite with device limits,” *International Journal of Aerospace Engineering*, vol. 2021, Article ID 6627193, 9 pages, 2021.

[5] E. Bensana, M. Lemaitre, and G. Verfaillie, “Earth observation satellite management,” *Constraints*, vol. 4, no. 3, pp. 293–299, 1999.

[6] N. G. Hall and M. J. Magazine, “Maximizing the value of a space mission,” *European Journal of Operational Research*, vol. 78, no. 2, pp. 224–241, 1994.

[7] M. Vasquez and J. K. Hao, “A ‘logic-constrained’ knapsack formulation and a tabu algorithm for the daily photograph scheduling of an earth observation satellite,” *Computational Optimization and Applications*, vol. 20, no. 2, pp. 137–157, 2001.

[8] M. A. A. Mansour and M. M. Dessouky, “A genetic algorithm approach for solving the daily photograph selection problem of the SPOT5 satellite,” *Computers Industrial Engineering*, vol. 58, no. 3, pp. 509–520, 2010.

[9] X.-j. Huang, M.-h. Ma, D.-s. Qiu, and J.-h. Zhu, “Solving the cooperative reconnaissance of electronic reconnaissance satellite with a hybrid scheduling algorithm,” *Journal of National University of Defense Technology*, vol. 33, no. 1, pp. 132–137, 2011.

[10] B.-f. Wu, Z.-g. Li, J.-y. Li, and J. Shi, “Design of mission-oriented autonomous commands for small satellite,” *Spacecraft Engineering*, vol. 22, no. 4, pp. 68–71, 2013.

[11] C. Hao, L. Jun, J. Ning, X. G. Liu, and Y. Tang, “Scheduling model and algorithms for autonomous electromagnetic detection satellites,” *Acta Aeronauctica et Aeronautica Sinica*, vol. 31, no. 5, pp. 1045–1053, 2010.

[12] T. Hai-Tao, W. Zhong-Guo, W. Da-Bao, and C. Jing, “Satellite utility improvement technique based on dynamic on-board command scheduling,” *Journal of Astronautics*, vol. 35, no. 10, pp. 1105–1113, 2014.

[13] Z. Tian, D. Zhao, and H. Tang, “Satellite system performance improvement techniques based on key path optimization,” in *The 1st China High Resolution Earth Observation Conference*, Beijing, 2012.

[14] S. Wang, *Graph Theory*, Science Press, Beijing, 2004.

[15] Y. Wang, *Discrete Mathematics*, Harbin Institute of Technology Press, 2007.