Hindawi
Complexity
Volume 2020, Article ID 2919615, 15 pages
https://doi.org/10.1155/2020/2919615

Research Article

Shortest-Path Optimization of Ship Diesel Engine Disassembly and Assembly Based on AND/OR Network

Deng-Zhi Chen,[1] Chen Wei,[2] Guo-Ling Jia,[3] and Zhi-Hua Hu[2]

[1]*Merchant Marine College, Shanghai Maritime University, Shanghai, 201306, China*
[2]*Logistics Research Center, Shanghai Maritime University, Shanghai, 201306, China*
[3]*School of Highway, Chang'an University, Xi'an, Shaanxi Province, 710054, China*

Correspondence should be addressed to Guo-Ling Jia; jgl@chd.edu.cn

Received 28 August 2019; Revised 20 November 2019; Accepted 13 January 2020; Published 18 February 2020

Academic Editor: Dimitri Volchenkov

Ship diesel engine disassembly and assembly (SDEDA) is essential for ship inspection and maintenance and navigation safety. The SDEDA consists of various machinery parts and operations. It is crucial to develop a system of SDEDA operations to improve the efficiency of disassembly and assembly (D&A). Considering the "AND" and "OR" relations (modeled as links) among the D&A operations (modeled as nodes), an "AND/OR" network is developed to extend a specialized graph model for the D&A sequencing problem in the context of education and training. Then, we devised a mixed-integer linear program (MILP) to optimize the SDEDA sequence based on the AND/OR network. Considering the flow balance in the AND/OR network, we developed exact algorithms and random search algorithms using breadth-first, branch cut and depth-first strategies to minimize the cost of the shortest path that represents an optimal sequence of D&A operations. To the best of our knowledge, it is the first try to formulate the D&A operations by an extended network model. Numerical experiments show that the proposed algorithms are practical for solving large-scale instances with more than 2000 D&A operations. The breadth-first shortest-path algorithm outperforms the MILP solver from the perspective of solution quality and computing time, and all developed algorithms are competitive in terms of computing time.

1. Introduction

A diesel engine is ship equipment that is essential for navigation and various shipping operations [1]. Ship diesel engine consists of a wide variety of machinery parts that are interconnected and mutually constrained. A single part is connected to several connecting components and is a pre-constraint of multiple parts. Ship diesel engine disassembly and assembly (D&A) (simplified as SDEDA) operations are complicated. It requires the operators to be aware of the parts' connections. When repairing or replacing specific parts, the operator develops a disassembly plan following consideration of the predisassembly conditions of the parts, subsequent disassembly constraints, and the state of the parts [2]. Besides, it is essential to establish a reasonable and efficient part D&A operational system to reduce redundant operations and improve equipment maintenance efficiency. Notably, different from general D&A operations that are conducted at specialized workshops or even factories, the SDEDA is generally handled on ships in ocean shipping. The skills of the seafarer operators are so crucial at such occasions that the D&A operations must be optimized, and thus, we try to standardize the optimal SDEDA processes for education purposes. In the context of education and training, typical test scenarios include finding an optimial sequence of disassemble sequence or assembling some parts of an engine with minimal steps.

The SDEDA process includes operations of D&A, maintenances, and overhauls. A widely used diesel engine (the type number is 6135) is taken as an example in this study due to the following reasons: this diesel engine is general and widely used in the shipping industry; it is a basic

configuration for the labs in maritime universities, and so we can use it for experimental studies. The disassembly process of this diesel engine involves 509 possible operations of 60 parts when we disassemble all these parts through various paths. Generally, we use 47 machinery tools to remove the diesel engine's block components or parts, replace or clean the piston, inject the oil, and handle the inner circulating water pipes. Due to the complexity of the equipment structure, the disassembly process needs to strictly follow the procedural principles: "from top to bottom," "from the outside to the inside," and "first removing, last installing." We conclude these principles according to the positions and the sequences of operating the parts. In real operations, the order of D&A of parts is usually determined and guided by experienced operators. So, D&A efficiency is affected by the experiences of the operators. Therefore, we aim at developing a system of reasonable D&A operations, to optimize the operations' sequences to improve the efficiency and accuracy of SDEDA [3].

The main problems faced by SDEDA are as follows: first, with the rapid development of the shipping industry, the scale of waterborne transportation continues to expand, which in turn leads to the aggravation of the maintenance work of ship diesel engines. Meanwhile, the aging problems in ships stimulate the increment of maintenance costs. The increasing amount of upgraded maintenance tasks and workload continuously requires improving maintenance efficiency. Second, modern ship engines are big and configured with complex electromechanical systems. In the D&A operations, the illegal activity will give rise to unimaginable damages or loss to the engines. It is not even possible for the operators to quickly locate the faulty components, which affects maintenance efficiency as well. In the case of time-sensitive requirements, the optimized D&A sequence and skilled operations have direct impacts on the ship repairing and operational costs.

We organize the remainder of the paper as follows: Section 2 presents related studies on machinery D&A. Section 3 introduces the methods used to model the D&A operational network. Section 4 devises the shortest-path model for D&A sequence optimization; Section 5 describes the algorithms for the shortest-path optimization based on the AND/OR network; Section 6 gives numerical studies to show the effectiveness of the proposed models and algorithms. The remarks and conclusion summarize the findings and indicate future directions in Section 7.

2. Related Studies

The research on SDEDA mainly concentrates on D&A technology and tools. First, we can classify the D&A technology research as a virtual and real type. In terms of real D&A, Hu et al. reviewed assembly representation methods, sequence generation methods, and assembly line balancing methods. They discussed the operational complexity of the assembly system and the role of the human operator from the perspective of product types, and disassembly and remanufacturing are challenging in the presence of a large number of product types [4]. Zhu et al. established an SDEDA database for D&A, ship management, ship repair, and ship craft teaching [5]. The database has various functions (such as browsing query, display, addition, deletion, and modification) and can provide services such as D&A preparation, D&A methods, operation procedures, and disassembly precautions. Tan et al. studied the disassembly process of offshore oil and gas platforms, considered the risks and costs of offshore operations, adopted the reverse installation method to minimize the elevator time, designed the A* algorithm, and optimized the D&A process of the components [6]. Chang et al. believed that the integration of upscale manufacturing equipment and maintenance services is not sufficient to ensure functional availability [7]. Therefore, they established an integrated product-service network model based on complex network and operational readiness. The model incorporates functional units, structural module units, and service execution activities to reveal modeling processes based on functionality, structure, service network, and network dependencies. The D&A technology used in the real process can be applied in practical directly but needs more cost. So, Qing et al. used Catia and Maya's modeling techniques to build an engine-overhaul plant simulation platform to avoid the disadvantages of real operation [8]. Hong and Qi-Long used a Petri net to obtain a general disassembly modeling method and expressed the disassembly process in an orderly manner [9]. Apart from single real or virtual technology, Chen et al. proposed a combination of virtual and practical assembly/disassembly [1]. The D&A of the crankshaft connecting rod piston parts was taken as an example to introduce the contents and steps of combined D&A. Feng considered the sustainability issue of D&A in the context of product recovery [10]. The D&A sequencing problem may incur uncertainties that make the problem even challenging [11].

Second, there are some research studies which focus on the selection and design of D&A equipment, sequencing, planning, and decision-making support systems. Güngör believes that D&A design is before product maintenance and remanufacturing, and the most critical issues are the choice of connectors [12]. Nahas formulated the machine type as decision variables to establish an optimization design model of the D&A manufacturing network and proposed an optimization method based on the proxy algorithm [13]. Starting from the overall structure of the process tree, Xie et al. proposed a process sequence sorting strategy to divide the processes into internal processes and external ones and optimize the scheduling sequence [14]. Pellegrinelli et al. studied motion planning and scheduling methods to reduce the cycle time of mission planning. These methods involve trajectory selection, task sequence, and task assignment [15].

Among these related studies, D&A operation optimization emerges as the primary research stream in the literature, as studied in Table 1. We reviewed 19 journal papers by using the querying keywords, disassembly, and assembly. We use four dimensions to examine the research characteristics of these studies. In the "D&A" dimension, we can find that 12 studies focus on "Disassembly" and six studies focus on "Assembly," while only one handles D&A simultaneously. In the "Issues" dimensions, 14 papers target at the

Table 1: Pioneering studies on D&A operations optimization.

	D&A	Issues	Modeling features	Algorithm
[16]	D	Sequencing	Precedence graph	Scatter search
[17]	D	Sequencing	Precedence graph	Network flow
[18]	D	Sequencing	Precedence graph	Three-stage iterative procedure
[19]	D	Sequencing	AND/OR graph	Iterative method
[20]	D&A	Sequencing	Precedence graph	Genetic algorithm
[21]	D	Sequencing	AND/OR graph	A two-phase algorithm
[22]	D	Sequencing	AND/OR graph	Graph algorithm
[23]	D	Sequencing	Precedence graph	Decision trees
[24]	A	Sequencing	Network and elimination	Case-based reasoning algorithm
[25]	D	Sequencing	Network and elimination	Efficient encoding and decoding strategy
[26]	A	Sequencing	Exploded view generation	Hybrid conjugated algorithm
[27]	D	Sequencing	Descriptive model	Hybrid ACO
[28]	D	Sequencing	Decision tool	Descriptive
[29]	A	Subassembly	Hierarchy	Graph algorithm
[30]	A	Classification	Hierarchy	Ant colony optimization
[31]	A	Solution space	Precedence graph	Ant colony optimization
[32]	D	Line balancing	Assignment model	Hybrid genetic algorithm
[33]	A	Graph generation	AND/OR graph	MILP
[34]	D	Presentation	Precedence graph	Genetic algorithm
#	D&A	Sequencing	AND/OR graph	MILP, graph algorithm, heuristics

"Sequencing" topic, while other studies investigate different aspects of sequencing and its system. In the column of "Modelling features," graph-based models are dominant in this research field. Precedent graphs are baselines of almost all studies, while "AND/OR" graphs are delicate ones considering the relations between an operation and its successors. Most papers describe the modeling results as a comparison base of the algorithms. Some studies utilize the network generation process [24–26]; and some studies describe the models by the ideal hierarchy structures [29, 30]. We can group the algorithms into four categories, as studied in the last dimension in Table 1. First, intelligent algorithms are dominant, including genetic algorithm, scatter search, and ant colony optimization. Second, the reviewed studies use experience-based heuristics widely, e.g., multistage and iterative algorithms. Third, the scholars increasingly concern the exact algorithms due to its computing performance, mainly including graph algorithms. Fourth, some scholars developed mathematical programs to describe the models formally and possibly be solved optimally by on-the-shelf mixed-integer linear program (MILP) solvers.

The end line of Table 1 describes the characteristics of four dimensions for this study. The issues studied in this paper are kinds of sequencing problems. We also use the AND/OR graph as a basic model to formulate the complicated process of D&A. However, besides general D&A operations, we consider tools, parts, and backgrounds to revise the general D&A operations and so the network is an extended one. Besides, we also developed formal mathematical programs. As for the solution algorithms, MILP solver, graph traversal algorithms, and heuristics are all used because their strengths are different. We extended the graph-based algorithms by considering the AND/OR relationships.

In summary, although the above research results offer useful ideas and inspiration for the SDEDA, a wide variety of tools and parts involved in the process contribute to the high complication. The current procedures are still facing the problems of time-consuming and requiring additional studies. Because of these issues, this paper establishes the AND/OR network optimization procedures to investigate them.

The contributions of this manuscript are as follows. First of all, the paper construct the AND/OR network for the equipment D&A process where the directed arc represents the process direction, node means D&A operations, the "OR" node represents the optional process of the part, and the predecessor of the node describes the pre-D&A requirements of the part. Secondly, we formulated a MILP to minimize the path between two nodes. Lastly, the breadth-first search algorithm and search strategy are designed for the AND/OR network to optimize the shortest path between nodes.

3. Methods

The AND/OR network describes the logical relationship of "AND" and "OR" in the connection between nodes, and the node that has these relationships is called "AND" or "OR" node. The "AND" node indicates that the node can be finished only by completing all its precursors (nodes). And the "OR" node means that we end a node as long as we end one of its successors in an AND/OR network [35]. The existing studies can generally address the precedence and "AND" relationship issues, but not thoroughly discuss the "OR" relations in the developed solution methods [36].

In the complex context of SDEDA, AND/OR network G is used to describe the network of D&A operations. $G = (V, A)$ defines a directed graph, where V is a set of nodes representing D&A operations, consisting of "AND," "OR," and common nodes; A is a set of directed arcs indicating the process, linking operation and one of its subsequent operations. An "AND" node indicates that the corresponding operation can be performed only by finishing all its precursor

operations. And an "OR" node means that we must conduct at least one of the successor nodes. For example, at the point of "checking the connecting rod bearing" (Figure 1), whether to replace the connecting rod bearing is determined based on the crack. From the perspective of education and training operators, the choices of replacing rod bearing (T74) or cleaning rod bearing (T75) are both available. Therefore, the operation of "checking the connecting rod bearing bush" (T48) is expressed as an "OR" node in G, as shown in Figure 1.

In the AND/OR network, the shortest path is a sequence between two specified nodes in the network such that their constituent nodes link with each other according to the AND/OR relations and the sum of the weights of their constituent arcs is minimal. An "AND" node on the shortest path may connect with multiple nodes due to its "AND" relation. An "OR" node connects with one and only one of its successor nodes. The shortest path is significant for the D&A operations because it represents a sequence of assembling or disassembling parts of the engine with the minimum time cost and operational cost. Figure 2 shows an AND/OR network consisting of five "AND" node (a, b, d, e, g), one "OR" node (c), one common node (f), eight arcs, and a feasible shortest path from node a to g as well, where the square represent common nodes, the circles represents the "AND" node, the diamond represents "OR" nodes, the dotted lines with arrows indicates the directed arc, and the solid lines with arrow indicates the directed arcs through which the shortest path passes. The shortest path between the node a and the node g is formed as follows. We construct the shortest path by starting from the destination node g and then connect all the precursors of node g, covering node b and f. Next, as node d and e are the successor nodes of the OR node c, we add only one of them to the shortest path, such as node d. Then, we add the nodes a and c to the shortest path because they are the precursors of node b and d. Finally, we connect node c and its precursor node a. This shortest path needs to pass through at least five nodes and six arcs from node a to reach point g. Similarly, (a, b, c, e, f, g) constitutes another shortest path with five nodes and six arcs from node a to g.

4. Model

We established an MILP ((1)–(12)) to minimize the arc costs of the shortest path according to the balance of inflow and outflow in the path as well as AND/OR logic relationship between the nodes by using the notations as follows.

Sets:

V^A, a set of "AND" nodes
V^R, a set of "OR" nodes
V^T, a set of common nodes
$V\{1, \ldots, n\}$, a set of nodes, $V = V^A \cup V^R \cup V^T$
$A\{(i, j) \mid i, j \in V\}$, the set of arcs linking the nodes in V

Parameters:

s, an original node of the shortest path, $s \in V$
d, sestination of the shortest path, $d \in V$
C_{ij}, the real number, time cost for arc (i, j), $(i, j) \in A$
I_i, integer, in-degree of node i in the network, $i \in V$
I_i^M, integer, the required minimum in-degree of node i if node i is in the shortest path, $i \in V$, $I_i^M = \begin{cases} 1, & i \in V^T, \\ I_i, & i \in V^A \cup V^R \end{cases}$

Variables:

$x_{ij} \in \{0, 1\} x_{ij} = 1$, if arc (i, j) is in the shortest path; otherwise, $x_{ij} = 0$, $(i, j) \in A$.
$u_i \in \{0, 1\} u_i = 1$, if node i is on the shortest path; otherwise, $u_i = 0$:

$$\min f = \sum_{(i,j)\in A} C_{ij} x_{ij}. \tag{1}$$

which subjects to

$$\sum_{(i,s)\in A} x_{si} \geq 1, \tag{2}$$

$$\sum_{(i,d)\in A} x_{id} = I_d^M, \tag{3}$$

$$u_s = 1, \tag{4}$$

$$u_d = 1, \tag{5}$$

$$M \cdot u_i \geq \sum_{(i,j)\in A} x_{ij}, \quad \forall i \in V,\ M = |V|, \tag{6}$$

$$u_i \leq \sum_{(i,j)\in A} x_{ij}, \quad \forall i \in V, \tag{7}$$

$$M \cdot u_i \geq \sum_{(j,i)\in A} x_{ji}, \quad \forall i \in V,\ M = |V|, \tag{8}$$

$$u_i \leq \sum_{(j,i)\in A} x_{ji}, \quad \forall i \in V, \tag{9}$$

$$x_{ji} \geq 1 + M \cdot (u_i - 1), \quad \forall i \in V^A (j, i) \in A,\ M = |V|, \tag{10}$$

$$\sum_{(i,j)\in A} x_{ij} \geq 1 + M \cdot (u_i - 1), \quad \forall i \in V^R,\ M = |V|, \tag{11}$$

$$\begin{aligned} x_{ij} &\in \{0, 1\}, \\ u_i &\in \{0, 1\}, \\ &\forall i, j. \end{aligned} \tag{12}$$

Constraint (1) minimizes the cost of the shortest path; (2) ensures that the out-degree of the initial node in the shortest path is not less than one; (3) indicates that the in-degree of the end node is equal to its required minimum in-degree. (4) and (5) suggest that the shortest path must contain an origin and a destination node. (6) suggests that if the out-degree of the node i in the shortest path is greater than 1, the node i is included in the shortest path; (7) indicates that if the out-degree of the point i in the shortest path is 0, the node i is excluded from the shortest path; similarly, (8) indicates that if the in-degree of the node i is

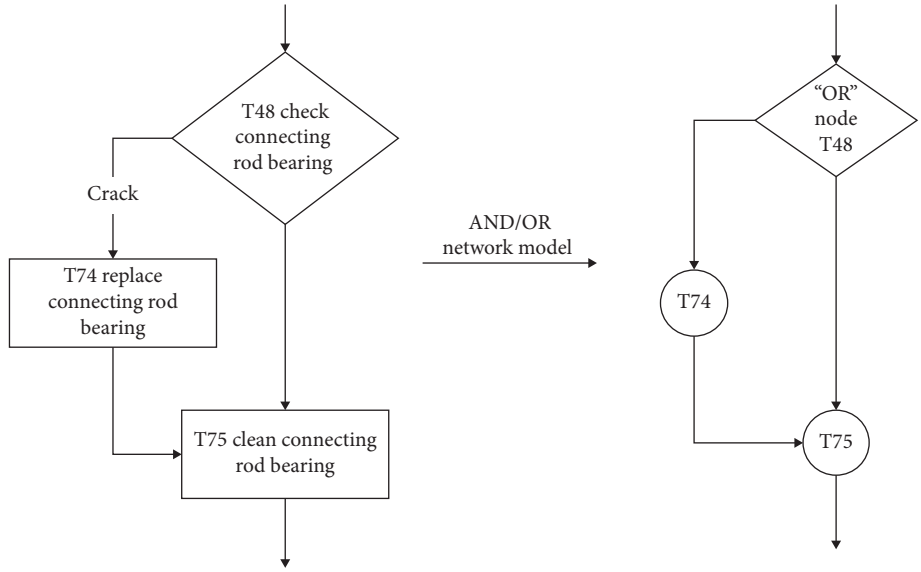


Figure 1: The "OR" node representing an operation of selectable successor operations in D&A operations.

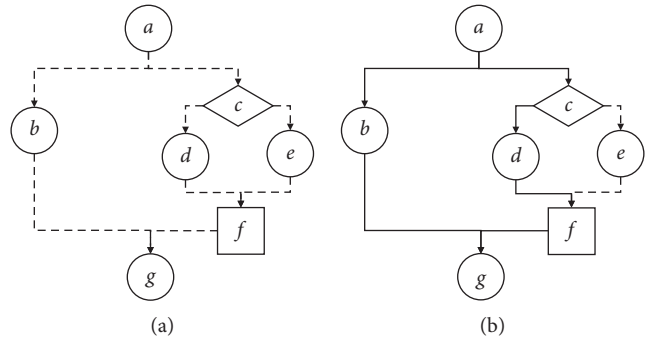


Figure 2: (a) AND/OR network and (b) the shortest path from a to g (solid line).

greater than 1, the node i is included in the shortest path, and (9) suggests that if the in-degree of the node i is 0, the node i is excluded from the shortest path; (6) to (9) constitute the flow balance of outflow and inflow of the nodes in the shortest path. Flow balance constraint indicates that the outflows and inflows of a node should be equal; for the shortest path, they should equal one. Constraint (10) guarantees that the in-degree of the "AND" node is not less than its minimum in-degree. Constraint (11) ensures that the out-degree of the "OR" node in the shortest path is not less than 1, indicating the "OR" node must select at least one successor. In (12), the decision variables x and u must be 0/1 integers.

Notably, in the model for the investigated D&A problem, we use s and d to represent the original and target part in a sequence of D&A operations. Generally, s is a virtual origin when we try to sequence the parts for removing d gradually; d is a virtual origin when we try to sequence the parts for assembling them into a whole from s.

5. Algorithms

In the AND/OR network, there exist three types of nodes, namely, common, "AND," and "OR" nodes. A typical node requires that we complete one of its precursors; an "AND" node requires that we complete all of its precursors; an "OR" node requests that the algorithm travels one of its successors. In the shortest path in the AND/OR network, one AND node connects with multiple nodes. We cannot directly apply the traditional breadth and depth search algorithms of the shortest path to the AND/OR Network. Therefore, we proposed the following three optimization strategies for searching the shortest-path search in the AND/OR Network.

5.1. Breadth-First Search Based on AND/OR Relations. Some of the nodes in the AND/OR network may have both "AND" and "OR" links with precursor nodes. In the AND/OR network, one node can be a node of type "AND" and "OR" simultaneously, according to the required number of its precursors and the required number of its candidate successors. The coupling of "AND" and "OR" relations in a single node increases the computing complexity of the shortest path. Therefore, we developed Algorithm 1 to determine the branches of choices from OR nodes and mark the conflict nodes that excluded each other from the shortest

Input

$G = (V, A)$: original network;
s: origin node of the shortest path;
d: destination of the shortest path.

Sets

V_{or}: a set of nodes;
V_{de}: a set of deleted nodes;
A_{de}: a set of deleted arcs;
A_{add}: a set of new-added arcs;
B_i: a sub-set of "OR" nodes.
V^{OS}: set of nodes that have successor relationships with node s;
V^{DP}: set of nodes that have precursor relationships to nodes p;
V^N: set of nodes in the cleaned network.
A^N: set of arcs in the cleaned network.
V^O: set of OR nodes in V^N;
b: a branch of an OR node, consisted of nodes;
B_i: set of branches of OR node i, $B_i = \{b_1, b_2, \ldots\}$;
V_i^C: set of nodes that are excluded from a path if node i is in the path

Output

$G^* = (V^N, A^N)$: the network starting from node s and ending in node d;
V_i^C: set of conflict nodes of node i;
B_i: branches of OR node i;

Process

Step 1: find the successor nodes to which can be reached by a route from s, denoted as V^{OS},
$V^{OS} = \{i \mid (s,i) \in A \text{ or } (j,i) \in A, j \in V^{OS}, i \in V\}$
Step 2: find the precursor nodes from which the destination d can be reached by a route, denoted as V^{DP},
$V^{DP} = \{i \mid (i,d) \in A \text{ or } (i,j) \in A, j \in V^{DP}, i \in V\}$
Step 3: select the common nodes in both V^{OS} and V^{DP}, denoted as V^N, $V^N = V^{OS} \cap V^{DP}$.
Step 4: select the arcs among the nodes of V^N to form A^N, $A^N = \{(i,j) \mid (i,j) \in A, i, j \in V^N\}$.
Step 5: for v in V^O
Step 5.1: select the successors of v, denoted as V^S
Step 5.2: for v^s in V^S
Step 5.2.1: add the nodes having a successor of v^s, denotes as $b_{v^s}^V$ generate $b_{v^s} = \{V_1, V_2, \ldots\}$, where V_1 contains the nodes that are successors of v^s, V_2 contains the nodes that are successors of nodes in V_1
Step 5.2.2: for v^N in $b_{v^s}^V$
Step 5.2.2.1: if v^N exists in all $b_{v^s}^V$, $v^s \in V^S$
set $v^T = v^b$
go to Step 5.2
Step 5.3: for v^s in V^S
Step 5.3.1: for v^b in reverse (b_{v^s}): reverse (Set) means reversely visit the elements in Set
Step 5.3.1.1: if $v^T \notin v^b$
remove v^b from b_{v^s}
else
go to Step 5.3.1
Step 5.3.2: update $b_{v^s}^N$, $b_{v^s}^N = \{i \mid i \in v^b, v^b \in b_{v^s}\}$
Step 5.4: for v^s in V^S
Step 5.4.1: for v^N in $b_{v^s}^N$
Step 5.4.2: $V_{v^N}^C \longleftarrow U_{v^{s*}}(b_{v^{s*}}^N)$, $v^{s*} \in V^S$, $v^s \neq v^{s*}$
Step 6: output $G^* = (V^*, A^*)$, V_i^C

Algorithm 1: Network cleaning and Branches Preprocessing.

path. We removed the redundant nodes that are not necessarily in the path from the origin node to the destination node.

Algorithm 2 is a breadth-first shortest-path algorithm with the AND/OR network G. Algorithm 2 starts the search by initializing one path containing the destination node and recording the destination as the new-added node in the path. Then, the algorithm adds all the precursors of the new-added node to the path and logs the precursors as new-added nodes. During the process of adding precursors,

Input

V: a set of nodes in the network G
A: a set of arcs in the network G.
C_{ij}: cost of arcs (i, j), $(i, j) \in A$.
s: origin node of the shortest path.
d: destination of the shortest path.
V_i^C: set of nodes that are excluded from the path p if p contains node i.

Sets

P: the path sets $\{p_1, p_2, \ldots\}$, in which p_i contains the nodes of the path from node s to d.
V_i^F: a set of forbidden nodes for the path p_i, $V^F = \{V_1^F, V_2^F, \ldots\}$.
V_i^N: a set of the newly added node to the path p_i, $V^N = \{V_1^N, V_2^N, \ldots\}$.
m_i: binary variable, $m_i = 1$ if path p_i is completed; 0, otherwise. $M = \{m_1, m_2, \ldots\}$

Output

p^*: set of nodes in the shortest path;
f: the cost of the shortest path p^*.

Process

Step 1: initialize $p_1 = \{d\}$, $m_1 = 0$, $V_i^F = \varnothing$, $V_i^N = \{d\}$, $c_1 = 0$
Step 2: set $f = 2 \cdot \sum_{(i,j)\in A} C_{ij}$
Step 3: while $\exists m_i = 0, \quad p_i \in P$
Step 3.1: select the first path p_i that satisfies $m_i = 0$, $p_i \in P$
Step 3.2: if $c_i > f$
Step 3.2.1: set $m_i = 1$
Step 3.2.2: go to Step 3
Step 3.3: if $V_i^N = s$ or $V_i^N = \varnothing$
Step 3.3.1: set $f = c_i$
Step 3.3.2: set $p^* \longleftarrow p_i$
Step 3.3.3: set $m_i = 1$
Step 3.3.4: go to Step 2
Step 3.4: for v^N in V_i^N:
Step 3.4.1: select the precursor nodes of v, denoted by V^{Pre}
Step 3.4.2: for v^{Pre} in V^{Pre}:
Step 3.4.2.1: select $V_{v^{\text{Pre}}}^C$ (the conflict nodes of v^{Pre}), denoted by V^C
Step 3.4.2.2: set $v^C \longleftarrow V^C \cap p_i$
Step 3.4.2.3: if $v^C = \varnothing$
Step 3.4.2.3.1: add v^{Pre} to p_i, $p_i \longleftarrow p_i \cup \{v^{\text{Pre}}\}$
Step 3.4.2.3.2: delete v^N in V_i^N, $V_i^N \longleftarrow V_i^N / \{v^N\}$
Step 3.4.2.3.3: add v^{Pre} to V_i^N, $V_i^N \longleftarrow V_i^N \cup \{v^{\text{Pre}}\}$
Step 3.4.2.3.4: set $c_i \longleftarrow c_i + \sum_{v\in p_i} C_{v^{\text{Pre}},v}$
else
Step 3.4.2.3.5: generate a new path $p_{|P|+1} = p_i \cup \{v^{\text{Pre}}\}/V^C$
Step 3.4.2.3.6: set $m_{|P|+1} = 0$, $V_{|P|+1}^N = V_i^N \cup \{v^{\text{Pre}}\}/V^C$, $V_{|P|+1}^F = V_i^F \cup V^C$
Step 3.4.2.3.7: set $c_{|P|+1} \longleftarrow c_i - \sum_{v\in p_i} C_{v^C,v} + \sum_{v\in p_{|P|+1}} C_{v^{\text{Pre}},v}$
Step 3.4.2.3.8: add $p_{|P|+1}$ to P
Step 3.4.2.3.9: set $c_{|P|+1} = 0$
Step 4: output p^* and f

Algorithm 2: Breadth-first shortest-path algorithm.

when a node v_i to be added conflicts the nodes already in the path p, a new path is generated from p so that the new path contains node v_i and excludes the nodes conflicted by node v_i. To increase the efficiency of the search of path with minimal cost, we cut the paths with costs higher than the current minimal cost in the algorithm. After all the paths have added the precursors of their new-added nodes, Algorithm 2 selects the path with the minimum cost as the shortest path.

Figure 3 illustrates the generation of a new path during the search for Algorithm 2. Algorithm 2 starts the search by adding the precursors $\{v_1, v_4\}$ of the destination v_5 and obtains the path containing nodes $\{v_5, v_1, v_4\}$. Then, the algorithm adds the precursors $\{v_2, v_3\}$ of node v_4 and constructs a path ($\{v_5, v_1, v_4, v_2\}$) firstly. However, node v_3 is forbidden to be added to the path $\{v_5, v_1, v_4, v_2\}$ since v_2 and v_3 belong to different branches of the "OR" node. Next, a new path $\{v_5, v_1, v_4, v_3\}$ is generated by adding the node v_3 to

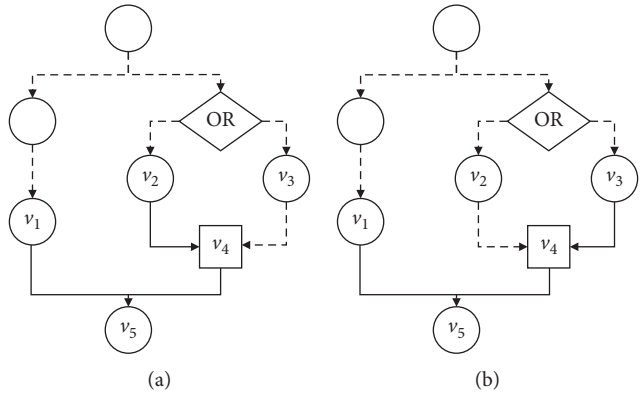


Figure 3: Algorithm 2 generates paths which selects the branches of an "OR" node. (a) A constructing path $\{v_5, v_1, v_4, v_2\}$ and (b) a new constructing path $\{v_5, v_1, v_4, v_3\}$.

Input

s: origin node.
d: destination node.
V: a set of nodes in the network G;
A: a set of arcs in the network G;
C_{ij}: cost of arc (i, j), $(i, j) \in A$.

Sets

B_i: set of branches of OR node i, $i \in V$;
V_i^{Pre}: set of precursor nodes of node i, $V_i^{\text{Pre}} = \{j \mid (i, j) \in A, i \in V\}$.
p: the shortest path, represented by the nodes in the shortest path.

Output

p: the shortest path p;
f: cost of path p

Process

Step 1: initialize $p = \{d\}$.
Step 2: select the precursor nodes of the nodes in p, denoted as V^{Pre}, $V^{\text{Pre}} = \{j \mid (j,i) \in A, j \notin p, i \in p\}$.
Step 3: while $V^{\text{Pre}} \neq \varnothing$
Step 3.1: add V^{Pre} to p, $p = p \cup V^{\text{Pre}}$
Step 3.2: update V^{Pre}, $V^{\text{Pre}} = \{j \mid (j,i) \in A, j \notin p, i \in p\}$.
Step 4: select all the OR nodes in p, denoted as V^O.
Step 5: for v in V^O
Step 5.1: select the branch b with minimal cost in B_v.
Step 5.2: select the nodes of other branches, denoted by V^B
Step 5.3: remove the nodes in V^B from path p, $p \longleftarrow p/V^B$
Step 6: calculate the cost f of the arcs (i, j) in p, $i, j \in p$
Step 7: output the path p and the cost f

Algorithm 3: Node cut-off search based on the precursor.

path $\{v_5, v_1, v_4\}$ and removing the nodes conflicted by node v_3. Then, Algorithm 2 will expand the path $\{v_5, v_1, v_4, v_2\}$ and a new path $(\{v_5, v_1, v_4, v_3\})$ into feasible paths linking the original node and destination and calculate their costs.

5.2. Node Cut-Off Search Based on Precursor. Algorithm 3 searches the shortest path by cutting off redundant branches of OR nodes based on a full-linked network G. First, Algorithm 3 selects all the nodes that have precursor relationship of the destination and adds the arcs among the selected nodes to the path, denoted by G; then, Algorithm 3 reserves one branch of each OR node in G and cuts out other branches to form a feasible path p connecting the original node and destination; finally, the algorithm calculates the arc costs in p and outputs p and its cost.

Figure 4(a) shows a full-linked network connecting the origin node s and the destination d, as a result, generated

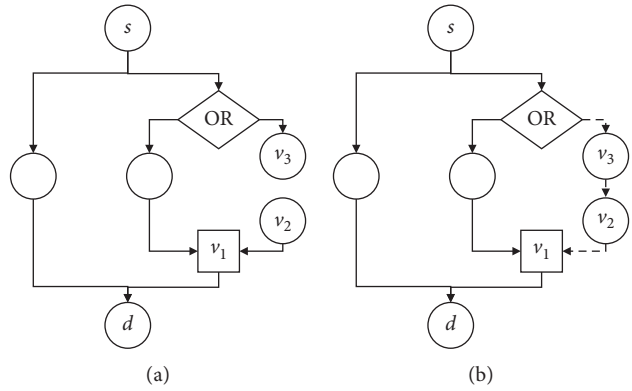


Figure 4: Algorithm 3 cuts off redundant branches of the OR node: (a) a full-linked network connecting s and d and (b) cut-off one branch of node OR.

from Step 1 to Step 3 of Algorithm 3. The network G is constructed by initializing from destination d and recursively adding all the precursors of the nodes in the network, in which solid arrow lines denote the linked arc. There exists one OR node (denoted as OR in Figure 4) in G. As the results of the cutting process from Step 4 to Step 5 of Algorithm 3, a feasible shortest path is obtained by cutting off the redundant branches of node *OR*, as shown in Figure 4(b). The arcs, $\{(\text{OR}, v_3), (v_3, v_2), (v_2, v_1)\}$, denote the cut-off branch and are represented by dotted arrow lines in Figure 4(b).

5.3. Random Depth-First Search. The random depth-first search initializes the destination as the first node in the path and adds the precursors of the nodes in the path until the path includes the origin node and its successors. When adding new nodes that belong to different branches of a single OR node, the algorithm randomly selects one node according to a predetermined threshold value. Algorithm 4 iterates multiple times to generate various paths and choose the path with the minimal cost as the solution. Algorithm 4 presents the search processes.

Figure 5 displays the random selection of nodes among different branches of a single OR node. When Algorithm 4 adds one node (v_3) to the path containing conflict nodes (v_2), we generate a random number (α) and a threshold value β, and then compare the results when using them in the algorithm. If $\alpha \leq \beta$, the algorithm reserves the existing node (v_2) in the path and excludes the node (v_3) from the path; otherwise, the algorithm eliminates the current node (v_2) out and adds the node (v_3) to the path.

6. Numerical Study

Taking the diesel engine (type number 6135) as an example, we establish the AND/OR network for its D&A according to the predecessor and subsequent relationship of component, and then develop the shortest steps to disassemble and assemble two specific parts. We used the Matlab 2018a for the coding of the shortest-path planning model and the search algorithm. And the model is solved by Gurobi 7.25 (https://www.gurobi.com/). We execute the program in the environment of the 64-bit Windows 7 operating system in a computer with Intel Core i7-5500U dual-core CPU, 8 GB RAM.

6.1. Network Construction. As for the single node in the AND/OR network, the parts and component operations of the diesel engine are defined through the following attributes: parts or components to be disassembled, such as instrument boxes, water supply valves, and return pipes; tools to be used, such as inner diameter gauges, torque wrenches, and particle adsorption tools; specific actions performed, such as the circle assembly, visual inspection, and wiping; specific scenarios conducting performance, such as the water supply line area, control panel, and fuel supply area. For a given power facility and workspace, the above four attributes uniquely identify an operation. Figure6shows the AND/OR network of D&A operations for the diesel engine, where the diamond node represents the "OR" node. The network has 394 common nodes, 115 "OR" nodes, and 1143 directed arcs.

Figure 7 is the result of conceptualizing the AND/OR network by a real-world case using the diesel engine (type number 6135). Our team took about one year to construct and verify this diagram with specialized operators in our lab and the cooperative companies. Because we need to find all possible successors of operation when it is determined, we must try all possibilities of D&A methods. As studied in the above paragraph, we denote all four categories of information for every node. Then, we coded the diagram into structured text tables corresponding to the four groups. So, we named this diagram a conceptualized AND/OR network for the D&A operations because we denoted the nodes and arcs with the four categories of information. We also added annotations to the nodes and arcs. Then, we used Algorithm 1 to convert the conceptualized network to a structured network. The "OR" nodes are in green color.

To test the effectiveness of the mathematical programming model and the search algorithms, we generate the data sets of AND/OR networks, in which the number of nodes ranges from 10 to 5000 and the number of arcs ranges from 10 to 8000. Using the above network as a base, we generated

Input

V: a set of nodes in the network G
A: a set of arcs in the network G.
C_{ij}: cost of arcs (i, j), $(i, j) \in A$.
s: origin node of the shortest path.
d: destination of the shortest path.
V_i^C: set of nodes that are excluded from the path p if p contains node i.
β: a given threshold value for choosing the node in branches of the "OR" node.
N: the maximal number of iterations

Sets

P: the path sets $\{p_1, p_2, \ldots\}$, in which p_i contains the nodes of the path from node s to d.
V^F: the set of forbidden nodes.
V^N: the set of new-added nodes to path p.

Output

p^*: set of nodes in the shortest path;
f: the cost of the shortest path p^*.

Process

Step 1: initialize $i = 1$, $f = \sum_{(i,j)\in A} C_{ij}$
Step 2: while $i <= N$
  Step 2.1: initialize $p_n = \{d\}$, $V^F = \varnothing$, $V^N = \{d\}$
  Step 2.2: while $V^N \neq \varnothing$
    Step 2.2.1: for v^N in V^N:
      Step 2.2.1.1: select the precursor nodes of v, denoted by V^{Pre}
      Step 2.2.1.2: for v^{Pre} in V^{Pre}:
        Step 2.2.2.2.1: select $V^C_{v^{\text{Pre}}}$ (the conflict nodes of v^{Pre}), denoted by V^C
        Step 2.2.2.2.2: if $V^C \cap p_i = \varnothing$
          add v^{Pre} to p, $p \longleftarrow p \cup \{v^{\text{Pre}}\}$
          delete v^N in V^N, $V^N \longleftarrow V^N / \{v^N\}$
          add v^{Pre} to V^N, $V^N \longleftarrow V^N \cup \{v^{\text{Pre}}\}$
        else
          randomly generate a value α, $\alpha \in [0, 1]$
          if $\alpha > \beta$
            add v^{Pre} to p, $p \longleftarrow p \cup \{v^{\text{Pre}}\}$
            delete the conflict nodes of v^{Pre} in p, $p \longleftarrow p / V^C$
            delete the conflict nodes of v^{Pre} in V^N, $V^N \longleftarrow V^N / V^C$
            add v^{Pre} to V^N, $V^N \longleftarrow V^N \cup \{v^{\text{Pre}}\}$
          else
            go to Step 2.2.1
  Step 2.3: calculate the cost c_i of arcs among the nodes of p_i
  Step 2.4: if $c_i < f$
    Step 2.4.1: set $f = c_i$
    Step 2.4.2: set p_i as the best path p^*, $p^* \longleftarrow p_i$
Step 3: output p^* and f

Algorithm 4: Random depth-first search.

the test data sets by removing nodes or inserting nodes to the existing networks by similar methods in the literature [24–26].

6.2. Results and Discussion. In the following, we conduct the experiments on the generated data sets by using the MILP model, Algorithms 2–4. We used Gurobi 7.25, an on-shelf solver of mathematical programs to solve the MILP model. The run time of the MILP solver for solving an instance once is limited to 3600 seconds. The number of iterations of Algorithm 4 for one example is 100.

Table 2 presents the experimental results of the model and the search algorithm. The computing time (CT) includes the code compiling time and the solving time. As an indicator for comparing the results between the proposed algorithms and the MILP model, a gap between the results is calculated by $\text{Gap} = (f^{\text{Algorithm}} - f^{\text{MILP}})/f^{\text{MILP}} \cdot 100$. For the instances with more than 2000 nodes, the MILP solver cannot return an optimal result before the run time

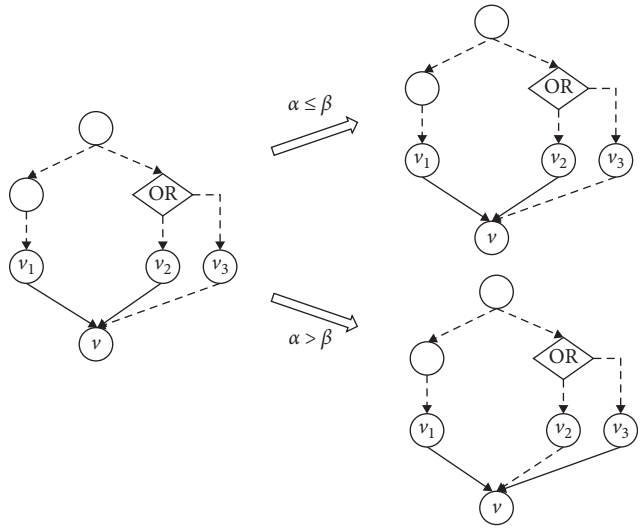


Figure 5: Branches of an "OR" node in Algorithm 4 according to α and β.

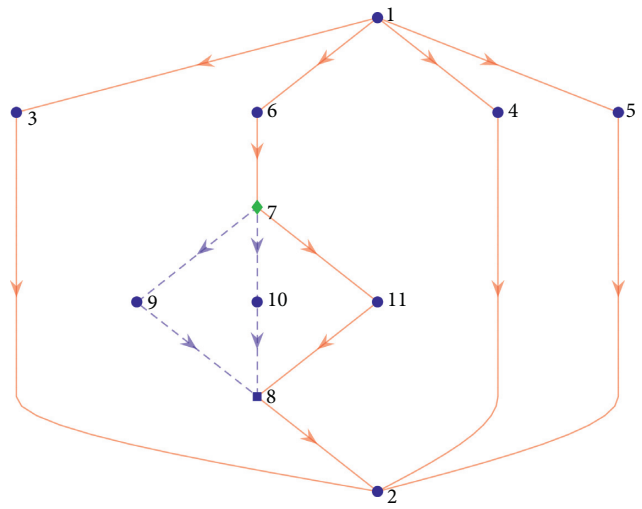


Figure 6: AND/OR network of the D&A operations for the diesel engine.

limitation, and then the objective value at the termination time is recorded as a result.

The MILP model can obtain the optimal results for the instances within 1500 nodes under the run time limitation, and for the example with more than 2000 nodes, near-optimal results are generated by the solver with termination conditions. The execution time of the MILP model increases rapidly when the number of nodes rises, driven by the time for handling the increasing number of the variables and constraints. Algorithm 2 generates the result close to the MILP solver and outperforms the MILP solver. Furthermore, Algorithm 2 requires much less computing time to obtain the solution, although we observed some fluctuations in the computing time when the number of nodes and arcs changes. Algorithm 3 has advantages against the MILP solver and other algorithms from the perspective of computing time, and it takes less than four seconds to solve the largest instance. However, the gaps in the results of Algorithm 3 are higher than other methods, and the gaps grow much when the scale of the problem expands. Algorithm 4 can find the optimal solutions for the instances of less than 100 nodes, and it outperforms Algorithm 3 in the view of the resultant values. Compared with Algorithm 2, Algorithm 4 spends less time to solve the problem according to the configuration of the iteration number, although they use a similar search strategy.

The experimental results show that the complete disassembly of the diesel engine requires a minimum of 954 steps, as shown in Figure 8. Figures 6 and 9 present the optimization results for the 11 and 52 nodes.

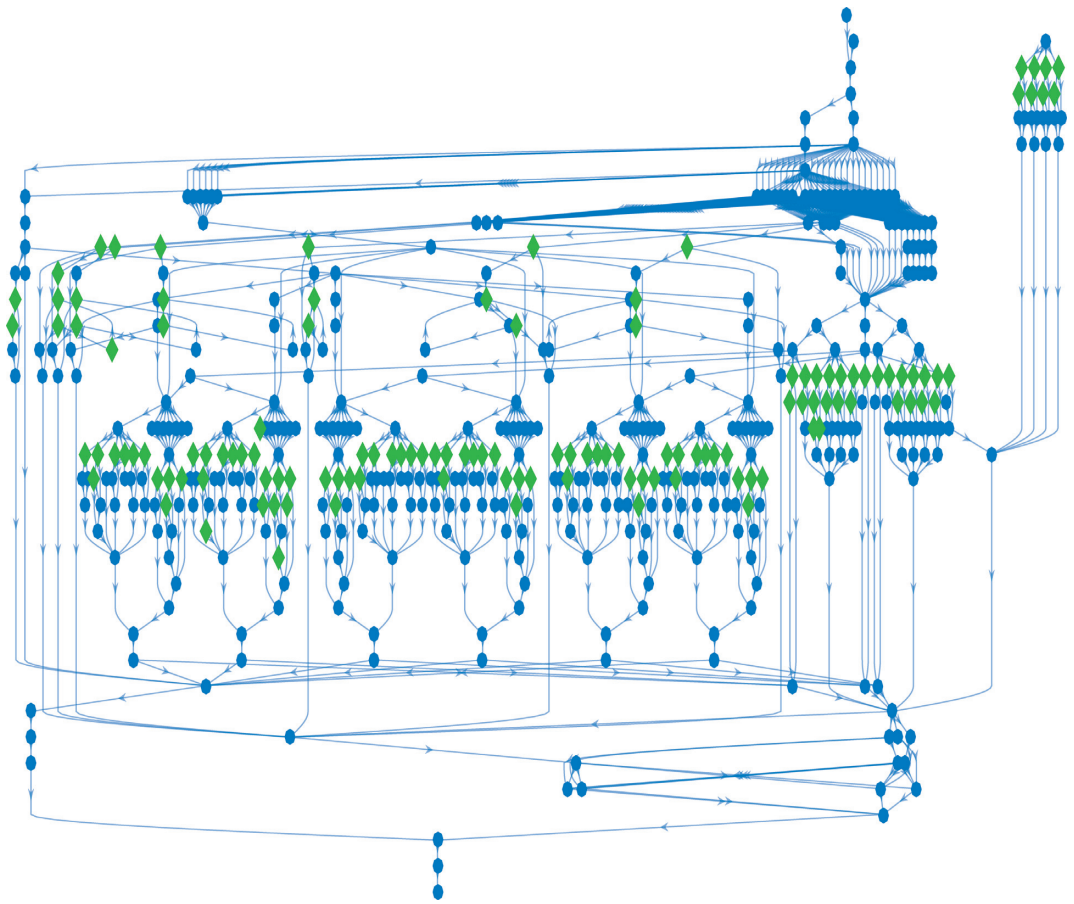


Figure 7: The AND/OR network of disassembly operations for the diesel engine.

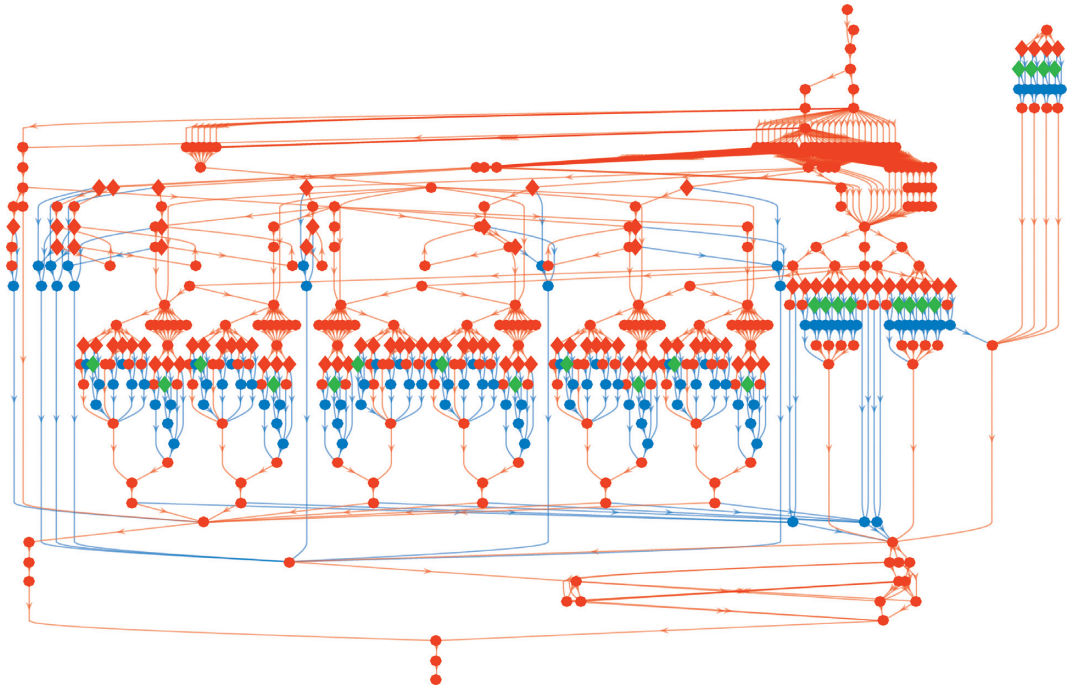
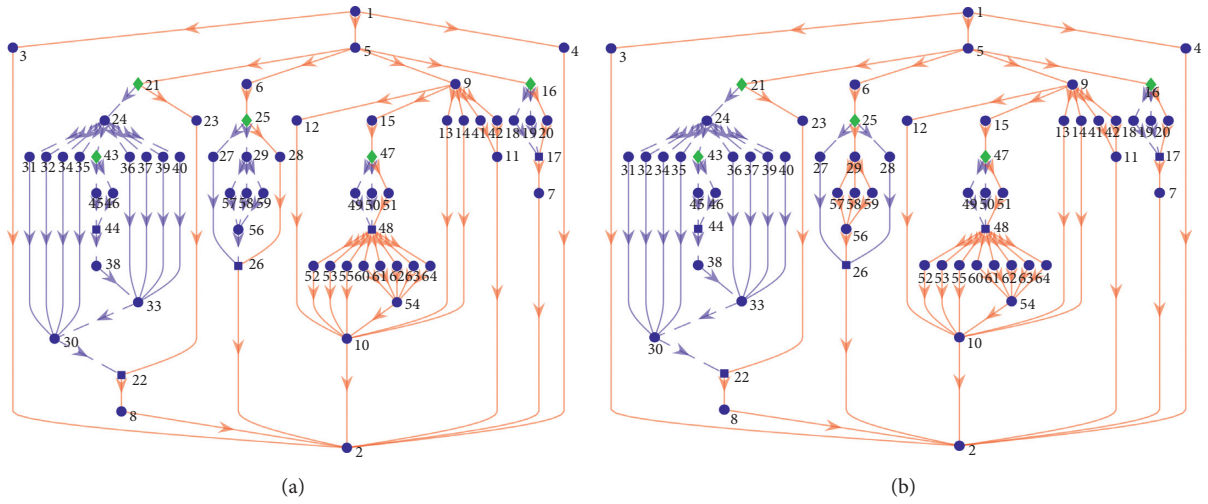


Figure 8: The shortest path of a network of 10 nodes and 15 edges.

Table 2: Comparison between different algorithms.

Node	Arc	MILP solver		Algorithm 2			Algorithm 3			Algorithm 4		
		f	CT/s	f	CT/s	Gap (%)	f	CT/s	Gap (%)	f	CT/s	Gap (%)
11	15	11	0.45	11	0.08	0.00	11	0.01	0.00	11	0.07	0.00
20	31	25	0.76	25	0.06	0.00	25	0.01	0.00	25	0.04	0.00
30	46	15	1.23	15	0.02	0.00	38	0.01	153.33	15	0.01	0.00
43	65	16	2.21	16	0.04	0.00	37	0.01	131.25	45	0.02	181.25
52	77	47	3.06	47	0.07	0.00	62	0.01	31.91	47	0.03	0.00
64	98	54	4.42	54	0.04	0.00	72	0.00	33.33	54	0.02	0.00
72	105	16	5.57	16	0.01	0.00	72	0.00	350.00	46	0.01	187.50
83	126	21	7.22	21	0.01	0.00	76	0.00	261.90	25	0.01	19.05
90	146	112	8.46	112	0.08	0.00	119	0.00	6.25	112	0.03	0.00
101	163	129	10.57	129	0.07	0.00	135	0.00	4.65	129	0.06	0.00
204	310	80	42.26	80	0.04	0.00	207	0.01	158.75	118	0.02	47.50
303	472	42	92.92	42	0.03	0.00	335	0.03	697.62	214	0.02	409.52
401	631	356	162.85	356	0.21	0.00	504	0.02	41.57	448	0.11	25.84
501	797	79	254.59	79	0.07	0.00	628	0.14	694.94	79	0.06	0.00
602	957	516	367.41	516	0.31	0.00	780	0.03	51.16	679	0.24	31.59
701	1103	413	498.12	413	0.63	0.00	863	0.13	108.96	533	0.26	29.06
802	1260	697	652.40	697	0.63	0.00	971	0.17	39.31	762	0.37	9.33
901	1453	614	823.42	614	0.99	0.00	1165	0.93	89.74	672	0.49	9.45
1005	1604	853	1024.83	853	1.16	0.00	1313	0.19	53.93	943	0.62	10.55
1504	2387	1007	2302.60	1007	3.48	0.00	1825	0.87	81.23	1514	3.13	50.35
2000	3168	1314	3600.00	1186	4.68	-9.74	2517	2.09	91.55	1686	2.14	28.31
2501	3984	1925	3600.00	1869	4.28	-2.91	3076	2.24	59.79	2208	3.47	14.70
3002	4768	744	3600.00	738	1.46	-0.81	3764	2.32	405.91	1298	1.35	74.46
3504	5565	869	3600.00	802	2.80	-7.71	4396	6.08	405.87	1461	1.44	68.12
4003	6366	950	3600.00	861	2.47	-9.37	4969	7.90	423.05	3393	1.60	257.16
4500	7118	2066	3600.00	1921	4.08	-7.02	5389	19.92	160.84	2795	2.46	35.29
5003	7944	968	3600.00	881	3.69	-8.99	6131	36.04	533.37	1332	2.60	37.60



(a) (b)

Figure 9: The shortest path of the AND/OR network with 64 nodes and 98 edges: (a) integer-programming model optimization results and (b) precursor-based "OR" node cut-off search.

7. Remarks and Conclusions

There are strict pre- and postrelationships between the operation of parts and components of the ship diesel engine D&A process, and some of them need to be selected by experienced judgments. Although the process manual stipulates the D&A steps, precautions, and component installation sequence, the actual D&A of specific parts often relies on manual experience due to the wide variety of parts and the complicated D&A steps.

This paper first establishes the AND/OR network for the operation based on the predecessor relationship of the node and the subsequent selection of the judgment node. Secondly, we developed the mathematical program

and designed the heuristics algorithms to optimize the shortest path between the nodes. Considering the "AND" relationship between nodes and the selective successor of the "OR" node, the shortest-path search and optimization method of the OR network is proposed.

Taking a typical ship diesel engine as an example, we tested the effectiveness of the model and algorithm for application. Based on the designed AND/OR network covering multiple nodes and arcs, we compared the optimization efficiency of the planning model and algorithms. The results show that the breadth-first shortest-path algorithm can achieve solutions close to the optimal ones returned by solving the mathematical program. Therefore, the method is useful for solving SEPDA problems. As for future research, we will study various network traversal search algorithms, network evaluation methods, D&A sequence evaluation, and data-driven and intelligent D&A evaluation and optimization methods. Additionally, we will extend the studies on general D&A sequencing problems by considering new features in marine diesel engine D&A problems for generality.

Data Availability

The data files are available in the Github repository (https://github.com/zhihuah0115/SDEDA).

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

The National Natural Science Foundation of China (no. 71871136), Science and Technology Commission of Shanghai Municipality (no. 17DZ2280200), and Shanghai Maritime University Graduate Innovation Fund (no. 2015ycx076) partially supported this study.

References

[1] Y.-H. Chen, W.-J. Wang, F.-X. Zhu, Y.-L. Li, and X.-W. Zhao, "Research on virtual disassembly and assembly of ship power equipment and its application," *China Water Transport*, vol. 2014, pp. 17-18, 2014, in Chinese.

[2] J.-H. Hu, D.-B. Hu, and J.-B. Xiao, "Research on comprehensive evaluation method of crew's post operation ability," *Computer and Digital Engineering*, vol. 45, pp. 1287–1293, 2017, in Chinese.

[3] Y.-W. Zhang, X.-L. Sun, Y.-W. Ding, and P.-T. Sun, "Design of intelligent diagnosis system for marine power equipment," *Chinese Journal of Ship Research*, vol. 13, pp. 140–146, 2018, in Chinese.

[4] S. J. Hu, J. Ko, L. Weyand et al., "Assembly system design and operations for product variety," *CIRP Annals*, vol. 60, no. 2, pp. 715–733, 2011.

[5] F.-X. Zhu, W.-J. Wang, J.-S. Lu, Y.-L. Li, B.-J. Wu, and Y.-G. He, "Development and application of ship power equipment disassembly database," *Journal of Zhejiang Ocean University (Natural Science)*, vol. 34, pp. 461–464, 2015, in Chinese.

[6] Y. Tan, Y. Song, X. Liu, X. Wang, and J. C. P. Cheng, "A BIM-based framework for lift planning in topsides disassembly of offshore oil and gas platforms," *Automation in Construction*, vol. 79, pp. 19–30, 2017.

[7] F. Chang, G. Zhou, X. Xiao, C. Tian, and C. Zhang, "A function availability-based integrated product-service network model for high-end manufacturing equipment," *Computers & Industrial Engineering*, vol. 126, pp. 302–316, 2018.

[8] Z. Qing, Z. Yan, G. Qing, and Z. Hong-Li, "Simulation research on rapid assembley and disassembly of aeroengine based on motion capture equipment," *Computer Simulation*, vol. 35, pp. 257–262, 2018.

[9] G. Hong and W. Qi-Long, "Research on disassembly and installation process modeling method for virtual maintenance tasks," *Manufacturing Automation*, vol. 37, pp. 35–38, 2015.

[10] Y. Feng, Y. Gao, G. Tian, Z. Li, H. Hu, and H. Zheng, "Flexible process planning and end-of-life decision-making for product recovery optimization based on hybrid disassembly," *IEEE Transactions on Automation Science and Engineering*, vol. 16, no. 1, pp. 311–326, 2019.

[11] G. Tian, M. Zhou, and P. Li, "Disassembly sequence planning considering fuzzy component quality and varying operational cost," *IEEE Transactions on Automation Science and Engineering*, vol. 15, no. 2, pp. 748–760, 2018.

[12] A. Güngör, "Evaluation of connection types in design for disassembly (DFD) using analytic network process," *Computers & Industrial Engineering*, vol. 50, no. 1-2, pp. 35–54, 2006.

[13] N. Nahas, M. Nourelfath, and M. Gendreau, "Selecting machines and buffers in unreliable assembly/disassembly manufacturing networks," *International Journal of Production Economics*, vol. 154, pp. 113–126, 2014.

[14] Z. Xie, X.-H. Zhang, Y.-L. Gao, and Y. Xing, "Time-selective integrated scheduling algorithm considering the compactness of serial processes," *Journal of Mechanical Engineering*, vol. 54, no. 6, pp. 191–202, 2018.

[15] S. Pellegrinelli, A. Orlandini, N. Pedrocchi, A. Umbrico, and T. Tolio, "Motion planning and scheduling for human and industrial-robot collaboration," *CIRP Annals*, vol. 66, no. 1, pp. 1–4, 2017.

[16] B. González and B. Adenso-Díaz, "A scatter search approach to the optimum disassembly sequence problem," *Computers & Operations Research*, vol. 33, no. 6, pp. 1776–1793, 2006.

[17] A. J. D. Lambert, "Exact methods in optimum disassembly sequence search for problems subject to sequence dependent costs," *Omega*, vol. 34, no. 6, pp. 538–549, 2006.

[18] S. C. Sarin, H. D. Sherali, and A. Bhootra, "A precedence-constrained asymmetric traveling salesman model for disassembly optimization," *IIE Transactions*, vol. 38, no. 3, pp. 223–237, 2006.

[19] A. J. D. Lambert, "Optimizing disassembly processes subjected to sequence-dependent cost," *Computers & Operations Research*, vol. 34, no. 2, pp. 536–551, 2007.

[20] Y.-J. Tseng, H.-T. Kao, and F.-Y. Huang, "Integrated assembly and disassembly sequence planning using a GA approach," *International Journal of Production Research*, vol. 48, no. 20, pp. 5991–6013, 2010.

[21] Y.-S. Ma, H.-B. Jun, H.-W. Kim, and D.-H. Lee, "Disassembly process planning algorithms for end-of-life product recovery and environmentally conscious disposal," *International Journal of Production Research*, vol. 49, no. 23, pp. 7007–7027, 2011.

[22] R. Edmunds, M. Kobayashi, and M. Higashi, "Using constraint-satisfaction to optimise disassembly sequences generated from AND/OR information," *International Journal of Production Research*, vol. 50, no. 15, pp. 4105–4126, 2012.
[23] N. Ghoreishi, M. J. Jakiela, and A. Nekouzadeh, "A non-graphical method to determine the optimum disassembly plan in remanufacturing," *Journal of Mechanical Design*, vol. 135, no. 2, 2013.
[24] R. B. Hadj, I. Belhadj, M. Trigui, and N. Aifaoui, "Assembly sequences plan generation using features simplification," *Advances in Engineering Software*, vol. 119, pp. 1–11, 2018.
[25] Y. Ren, C. Zhang, F. Zhao, H. Xiao, and G. Tian, "An asynchronous parallel disassembly planning based on genetic algorithm," *European Journal of Operational Research*, vol. 269, no. 2, pp. 647–660, 2018.
[26] M. V. A. R. Bahubalendruni, A. Gulivindala, M. Kumar, B. B. Biswal, and L. N. Annepu, "A hybrid conjugated method for assembly sequence generation and explode view generation," *Assembly Automation*, vol. 39, no. 1, pp. 211–225, 2019.
[27] H.-E. Tseng, C.-C. Chang, S.-C. Lee, and Y.-M. Huang, "Hybrid bidirectional ant colony optimization (hybrid BACO): an algorithm for disassembly sequence planning," *Engineering Applications of Artificial Intelligence*, vol. 83, pp. 45–56, 2019.
[28] M.-L. Bentaha, A. Voisin, and P. Marangé, "A decision tool for disassembly process planning under end-of-life product quality," *International Journal of Production Economics*, vol. 219, pp. 386–401, 2020.
[29] Y. Wang and J. Liu, "Subassembly identification for assembly sequence planning," *The International Journal of Advanced Manufacturing Technology*, vol. 68, no. 1–4, pp. 781–793, 2013.
[30] S. Li, Y. Liu, J. Wang, and H. Zeng, "An intelligent interactive approach for assembly process planning based on hierarchical classification of parts," *The International Journal of Advanced Manufacturing Technology*, vol. 70, no. 9–12, pp. 1903–1914, 2014.
[31] H. Wang, Y. Rong, and D. Xiang, "Mechanical assembly planning using ant colony optimization," *Computer-Aided Design*, vol. 47, pp. 59–71, 2014.
[32] C. B. Kalayci, O. Polat, and S. M. Gupta, "A hybrid genetic algorithm for sequence-dependent disassembly line balancing problem," *Annals of Operations Research*, vol. 242, no. 2, pp. 321–354, 2016.
[33] A. J. D. Lambert, "Generation of assembly graphs by systematic analysis of assembly structures," *European Journal of Operational Research*, vol. 168, no. 3, pp. 932–951, 2006.
[34] J. R. Li, L. P. Khoo, and S. B. Tor, "A novel representation scheme for disassembly sequence planning," *The International Journal of Advanced Manufacturing Technology*, vol. 20, no. 8, pp. 621–630, 2002.
[35] S. Tao and Z. S. Dong, "Scheduling resource-constrained project problem with alternative activity chains," *Computers & Industrial Engineering*, vol. 114, pp. 288–296, 2017.
[36] G. Tian, Y. Ren, Y. Feng, M. Zhou, H. Zhang, and J. Tan, "Modeling and planning for dual-objective selective disassembly using and or graph and discrete artificial bee colony," *IEEE Transactions on Industrial Informatics*, vol. 15, no. 4, pp. 2456–2468, 2019.