algorithms

MDPI

Review

A Review: Machine Learning for Combinatorial Optimization Problems in Energy Areas

Xinyi Yang [1], Ziyi Wang [2], Hengxi Zhang [3], Nan Ma [1], Ning Yang [2,*], Hualin Liu [1], Haifeng Zhang [2] and Lei Yang [1]

[1] Key Laboratory of Oil & Gas Business Chain Optimization, CNPC, Petrochina Planning and Engineering Institute, Beijing 100083, China; yangxy234@petrochina.com.cn (X.Y.); manan2013@petrochina.com.cn (N.M.); liuhualin08@petrochina.com.cn (H.L.); yang_lei@petrochina.com.cn (L.Y.)
[2] Institute of Automation, Chinese Academy of Sciences, Beijing 100190, China; wangziyi2021@ia.ac.cn (Z.W.); haifeng.zhang@ia.ac.cn (H.Z.)
[3] Tsinghua-Berkeley Shenzhen Institute, Tsinghua University, Shenzhen 518055, China; zhanghx20@mails.tsinghua.edu.cn
* Correspondence: ning.yang@ia.ac.cn; Tel.: +86-1305-150-9251

Abstract: Combinatorial optimization problems (COPs) are a class of NP-hard problems with great practical significance. Traditional approaches for COPs suffer from high computational time and reliance on expert knowledge, and machine learning (ML) methods, as powerful tools have been used to overcome these problems. In this review, the COPs in energy areas with a series of modern ML approaches, i.e., the interdisciplinary areas of COPs, ML and energy areas, are mainly investigated. Recent works on solving COPs using ML are sorted out firstly by methods which include supervised learning (SL), deep learning (DL), reinforcement learning (RL) and recently proposed game theoretic methods, and then problems where the timeline of the improvements for some fundamental COPs is the layout. Practical applications of ML methods in the energy areas, including the petroleum supply chain, steel-making, electric power system and wind power, are summarized for the first time, and challenges in this field are analyzed.

Keywords: combinatorial optimization problem; machine learning; supervised learning; reinforcement learning; game theory; refinery scheduling; steel-making; electric power system; wind power

Citation: Yang, X.; Wang, Z.; Zhang, H.; Ma, N.; Yang, N.; Liu, H.; Zhang, H.; Yang, L. A Review: Machine Learning for Combinatorial Optimization Problems in Energy Areas. *Algorithms* **2022**, *15*, 205. https://doi.org/10.3390/a15060205

Academic Editor: Akemi Galvez Tomida

Received: 12 May 2022
Accepted: 8 June 2022
Published: 13 June 2022

1. Introduction

The optimization problem in energy areas has always been a hot topic with the development of the various types of industries over the past decades. Due to the natural mathematical similarities between energy models and COPs, many energy problems can be regarded as COPs. To figure out the above issues, an increasing amount of individuals and groups have begun to employ the multi-agent system framework, such as game theory, and intelligent learning approaches, such as machine learning methods, considering their satisfactory solution and implementation flexibility. In this work, we specifically concentrate on the interdisciplinary part among COP, ML, game theory and energy areas presented as Figure 1.

COP is the sub-field of the optimization problems and can be seen almost everywhere in the resource allocation, scheduling, and routing scenarios over the industrial fields. The task of COPs is to search and find the maximum or minimum of an objective function with a discrete domain rather than a continuous space [1–3]. In other words, the objective of COP, in mathematics, is to seek an optimal object combination from a collection of objects, in which the solution is either a discrete set or can be simplified to a discrete set [4]. Basically, a COP is established as either a maximization problem or a minimization one, just depending on the specific scenarios of given objective functions. In algorithmic,

maximization and minimization problems, they are treated equivalently in consideration of the same computational logic.

Meanwhile, in consideration of the abundant COP characteristics in energy scenarios, people have started to treat various industrial energy issues, such as crude oil scheduling, pipeline scheduling, electricity scheduling, and steel making, as well as other energy issues, as COPs. For instance, the task of crude oil scheduling is to utilize limited production equipment for oil manufacturing to achieve a certain objective, such as maximizing profits or minimizing cost [5–9]. A general pipeline operation can be presented as pumping a certain amount of product into the pipeline at the starting point while receiving the same volume of product at the other end of the pipeline [10]. For multi-product pipeline problems, delivery planning and production injection are two essential tasks in real pipeline operations [11,12]. The electricity scheduling aims to improve efficiency, reliability and security through automation and modern communication technologies, which are generally based on the optimization of the whole electricity system [13–16]. Steel making continues to be of high concern around the world since steel is a metal that is widely in usage in the construction of roads, bridges, railways, and other infrastructures, and even a tiny optimization of the manufacturing process will lead to a huge decrease in cost [17]. Any of the manufacturing, such as casting [18], charge calculation and melt [19], and emission [20].

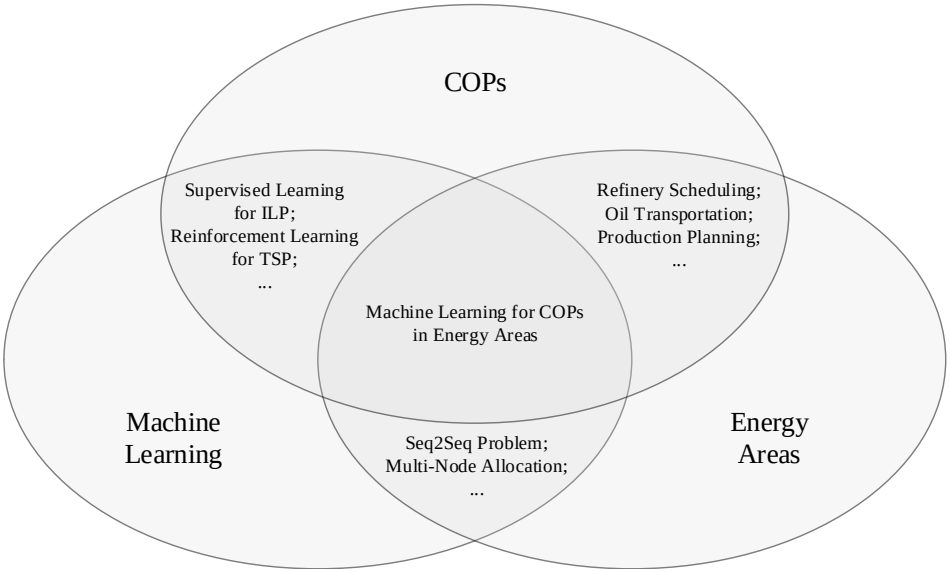

Figure 1. The interdisciplinary areas of COP, ML and energy areas.

Furthermore, there are numerous COPs in energy scenarios where many energy nodes are involved and modeled as different decision makers to optimize the operation efficiency of the whole energy system. Herein, game theory is a general framework that facilitates the comprehension of decision making in cooperative and competitive situations where agents have their own beliefs [21]. According to the type of information and the behaviors of players in the game environment, games can be generally grouped into four categories. First, static game with complete information. A static game is a simultaneous-move game where players must take action at the same time. With complete information, each player in the game has access to so-called common knowledge, including payoff functions and strategies as well as utility functions. The objective for each player is to formulate their strategies and maximize their benefits based on common knowledge. A famous example is the prisoner's dilemma. It introduces a situation where two individuals act for their own best interests and explores the corresponding decision-making policy taken by both [22,23]. Besides, Cournot competition is another instance conceptually similar to the prisoner's dilemma. This game model describes a dual-industry competition scenario where the product output is the competition objective for both industries, and independent decisions are required to be made by them both simultaneously to compete against each

other [24,25]. Second, static game with incomplete information. As opposed to the game with complete information, a game with incomplete information, also known as a Bayesian game, describes such situations where players possess only partial information about other players. For maximizing the benefits or minimizing the penalties of their own, each player needs to form the behavior expectations of others to support their own decision making. A typical instance is called matching pennies. This is a zero-sum game in which two players place a penny on the table at the same time and the profit depends on whether the pennies match [26,27]. Auction is another example of a game with an incomplete information, in which each bidder is required to make a decision only with the utility function of his own, without knowing other bidders' willingness toward the good. The third class is the dynamic game with complete information. On the other hand, the move mechanism in dynamic games is sequential in contrast to the static games. Players act one by one and adjust the strategies accordingly to achieve the objectives on their own. The Stackelberg game is a game formulation that belongs to this category. It is a strategic game, where a leader and a follower move sequentially and compete for quantity [28]. Dynamic game with incomplete information is the forth class. Correspondingly, a dynamic game with incomplete information presents a game scenario in which players sequentially make actions without full information about others to maximize their payoffs. Many primary game models take this form. In the model of job market signaling [29], workers and employers both seek the one that matches their expectations. The employers generally offer the workers a competitive salary equivalent to their expected ability. Another instance is initial public offerings [30], where the owner of a firm and a set of potential investors are modeled as players. The objective of the firm owner is to decide what fraction to sell to the potential investors and how much while the investors need to decide whether to take or refuse. In addition, games can also be generally classified into cooperative and non-cooperative, i.e., competitive games according to the interests of players in the game.

Under the framework of game theory, ML methods can be hopefully designed to help players or agents in the game environment to effectively form and update their strategies or policies. ML is a method for autonomous data analyzing [31], which is a branch of artificial intelligence. The idea behind it is that the system with a learning algorithm inside can learn from data and hence reinforce its experience in recognizing patterns and making decisions on its own. With the naturally interdisciplinary characteristics of both above, people have begun to investigate further a series of ML learning approaches which integrate both their merits and may hopefully enhance the decision-making strategy of the intelligent system. According to the signal feature of the learning environment, we generally divide all ML methods into three categories: supervised learning, unsupervised learning and reinforcement learning. The objective of supervised learning methods is to establish a mathematical mapping relationship of a collection of data that maps the inputs to the expected outputs [32]. As opposed to supervised learning, the data that unsupervised learning approaches take only contain the inputs. Their goal is to seek underlying structures in the data and present them, such as the clustering of distinguishable data [33–35]. Reinforcement learning (RL) is a reward-oriented learning approach, where agents are required to continuously explore the environment and maximize its accumulative reward [36–39]. Moreover, since abundant RL scenarios involve the participation of more than one single agent, an increasing amount of studies has been implemented in recent years, which has directly led to the development of multi-agent RL (MARL). In addition, the agents in the multi-agent system not only interact with the environment, but have to deal with those actions of other agents, which can be treated as a part of the environment, and therefore the equilibrium points in the system must be considered [40,41]. This is where the principles of game theory emerge. With the development of relevant game theory and ML methods, more and more individuals and institutes have started to integrate both of them on industrial issues, such as device-to-device (D2D) communication using the Stackelberg game and MARL method [42], anti-jamming for internet of satellites through the deep RL approach

and Stackelberg game [43], moving target defense using MARL in Bayesian Stackelberg Markov games [40], etc.

In this work, we mainly survey the COPs in energy areas, using a series of modern ML approaches, i.e., the interdisciplinary areas of COPs, ML and energy areas. We first list the categories of the classical COPs, such as the traveling salesman problem (TSP), maximum independent set (MIS) and minimum spanning tree (MST), etc., then investigate the ML techniques that can effectively address the above COPs, where the SL, DL and RL, as well as game-theoretic approaches, are mainly concerned due to their unique abilities. The research works of COPs in energy areas with ML methods are specifically studied afterwards. We mainly survey the petroleum supply chain, steel-making, electric power system, and wind power, which are currently popular applications in energy fields since the resource consumption is fairly high and even a little optimization can help save numerous costs. Finally, we also find some important deficiencies when investigating the above materials, and the challenges of this cross area are also summarized. On the one hand, we hopefully notice that game theory can be appropriately integrated into ML, DL and RL methods for handling the COPs in consideration of the typical issues, such as accuracy, efficiency, and scalability. On the other hand, the applications in energy areas are frequently represented in a complicated manner since the objective functions are often formulated as NP-hard problems with a great many constraints. The learning methods above can probably be a category of promising technique in dealing with these issues.

In addition, the contents are analyzed as follows. The search identified 274 documents containing the theoretical conclusions of various algorithms and the results of applicable approaches. Background information contains relevant knowledge in COPs [44–54], deep learning [55–61] and reinforcement learning [36,62–68]. Supervised learning, reinforcement learning and game theoretic methods are introduced in learning methods, where supervised learning includes methods of B&B [69–74], sequence to vector [57,75–77], GNN [78–82] and end-to-end architecture [83,84]. A few sources are relevant to reinforcement learning [85–88], and different types of games were demonstrated [89–92]. Another aspect mentioned was classic COPs [71,72,84,85,93–97], where ILP, MIS, MC, MVC and TSP were introduced. The rest of the literature references demonstrate application. The aspects contain petroleum supply chain [10,11,98–107], steel-making [108–111], electric power system [112–119], and wind power [120–123]. The aforementioned references were mostly searched from Google Scholar with related key words, such as “optimization”, “combinatorial optimization problem”, “machine learning”, “supervised learning”, “reinforcement learning”, “game theory”, “refinery scheduling”, “steel-making”, “electric power system”, “wind power”, etc.

The rest of this review is organized as follows. In Section 2, we introduce the background of combinatorial optimization problems and several popular approaches of ML in addressing this kind of issue over the past years. The sub-fields of combinatorial optimization problems and more details of these ML methods are further discussed in Section 3. In Section 4, we concentrate on some specific applications in energy field and investigate how the ML algorithms are applied to these issues. In Section 5, we summarize the challenges over these energy applications in which the ML methods are deployed. Then, some conclusions are drawn in facing the development of the ML theory in Section 6.

2. Background

In this section, we present a basic overview of COPs and the corresponding ML learning approaches for resolving them. Several primary COPs and corresponding applications are first reviewed. Then we investigate the attention mechanism and graphic neural networks (GNNs) as well as their typical categories and implementations. Moreover, since both of them can also be regarded as the policy network of RL agents, we further study how this framework is deployed and how it works. At the end of this section, we concentrate on RL and MARL approaches, in which the related game theories are considered a natural combination to improve the performance of the agents in specific scenarios.

2.1. Combinatorial Optimization Problem

The optimization problem can be viewed in terms of a decision problem, in which the total value of current solutions is mathematically evaluated by the objective function established in advance. Through a series of decisions, the task is to search and find the optimal value among different solutions. Typical COPs can be summarized as follows:

2.1.1. Traveling Salesman Problem

We herein would like to take the visit of cities as an example. The objective of the traveling salesman problem (TSP) is to seek the shortest possible path that allows the salesman to visit each city only once and returns to the origin city given the city list and connection distances between any two city nodes. TSP is utilized as a basic formulation for many optimization approaches, such as vehicle routing [44], scheduling [45], path planning [46], logistics [47], DNA sequencing [48] and the computing system [49]. In these applications, the city nodes in the graph represent, for instance, customers, soldering points, or DNA fragments, and each city pair or the distance represents the traveling time duration or cost, or the measurement between DNA fragments. Corresponding algorithms for coping with TSP include reinforcement learning [50], simulated annealing [51], genetic algorithm [52], ant colony [53], tabu search [54], or some mixed ones [124–126].

2.1.2. Maximum Independent Set

Maximum independent set (MIS) is a typical central problem in distributed graph algorithms, also known as the maximal stable set, which is an independent set that does not belong to any other independent sets. That is to say, an independent vertex set of a graph is a subset of the vertices such that no two vertices in the subset indicate an edge of this graph. Given a vertex cover of a graph, all vertices not in the cover define an independent vertex set. In addition, a MIS is also a dominating set in the graph, and each independent dominating set must be maximally independent. Many meaningful applications can be mathematically established as MIS problems, such as the communication system [127,128], computational system and graph coloring problem [129]. For handling MIS problems, some early researchers in studying graphs proposed some interesting algorithms [130–132]. Afterwards, more individuals and groups started to deploy a series of approaches with higher efficiency, such as the ML method [133], small messages rather than large messages [134], local search [135] and exact algorithm [136].

2.1.3. Minimum Spanning Tree

For an edge-weighted undirected graph, a minimum spanning tree (MST) is a tree that connects all vertices while not having any cycles and has a minimum total edge weight. That is, the sum of edge weights is supposed to be as small as possible for a spanning tree. There are many applications related to the MST problem. The direct ones consist of communication networks [137], transportation networks [138], water supply networks [139], and electrical grids [140]. Other practical cases based on MST include COVID-19 pandemic transmission forecasting [141], clustering [142], constructing trees for broadcasting [143], image registration and segmentation [144], circuit design [145] and emotion recognition [146]. There do exist several primary algorithms, namely, classic algorithm and faster algorithm; however, in consideration that such MST-like models will facilitate the development of multifarious industrial areas, more and more individuals have turned to design a variety of intuitive algorithms to figure out such issues, such as reinforcement learning [147], genetic algorithm [148] and fast parallel algorithm [149].

2.1.4. Maximum Cut Problem

The maximum cut (MC) problem is to find a cut such that the amount of edges between two complementary sets in a graph is as large as possible. The applications of the max-cut problem include theoretical physics [150–152], very large scale integration (VLSI) design [153,154], and the protein-folding problem [155,156], Accordingly, various

approaches and algorithms, such as eigenvalues [157], randomized heuristics [158], harmony search and genetic algorithm [159], and scatter search [160] have been proposed to address this problem over the past decades.

2.1.5. Bin Packing Problem

The bin packing problem (BPP) is one of the most classical COPs, where a series of items with different sizes are required to be packed into a series of bins with positive integer capacities such that the number of bins used is minimized. Generally, there are two types of BPPs: online BPP, and offline BPP. The online BPP considers a situation in which the items keep appearing in a given order and they need to be placed inside the bins one by one, while the latter one concerns modifying the given list of items. BPP has many variations, such as 2D packing [161], 3D packing [162], linear packing [163], packing by weight [164], packing by cost [165], and so on. Such problems have arisen in resource allocation with an increasing frequency over the past years, such as edge computing [166], cloud storage [167], parcel delivery [168], aircraft maintenance task allocation [169] and product transportation [170]. For the online version, there are a diverse set of online algorithms designed for BPP. Single-class algorithms consist of next fit [171], next-k-fit [172], first-fit [173], best-fit [174] and worst-fit [175]. Refined algorithms include harmonic-k [176] and refined-harmonic [177]. On the other side, the offline algorithm is able to observe all the items before beginning to place them into bins. Multiplicative approximation is the simplest approach utilized by the offline algorithm. The members in this family are first-fit-decreasing [178], next-fit-decreasing [179] and modified first-fit-decreasing [178]. Additive approximation [180] and exact algorithms [181] are also widely mentioned in solving BPPs.

2.2. Deep Learning

Deep learning (DL) is a member of ML methods, where multiple layers of artificial neurons or neural networks (NNs) are structured to learn the representation of data [55]. The structures of NNs can be quite fruitful, such as deep neural networks (DNNs), convolutional neural networks (CNNs), recurrent neural networks (RNNs), attention mechanism, graph neural networks (GNNs), etc. In this work, we mainly concentrate on attention mechanism and GNNs in consideration of the natural connection between COPs and these two techniques.

2.2.1. Attention Mechanism

Some COPs are typical sequence-to-sequence (Seq2Seq) [56] problems, (for example, TSP), which require an encoder to map the input sequence into a d-dimensional space, and then use a decoder to map it to the output sequence, and the sequences are usually long (due to large problem size) and variable in length. With the rapid development of DNNs in the ML approach over the past years, people have started to realize that conventional encoders, such as RNNs, fail to utilize the information of previous segments in a relatively long sequence. Thus, a novel technique that mimics cognitive attention, namely attention mechanism, was proposed afterwards.

The idea behind is that the network should devote more attention to those smaller but more important parts of the input data. It was first used to deal with the Seq2Seq model, especially for neural machine translation [57]. Then the encoder–decoder architecture was widely known. The decoder utilizes the most relevant segments of the input sequence, which are encoded as a fixed-length vector by the encoder in a flexible manner to obtain the variable-length target consequence [58].

Nevertheless, the decoder here can just partially obtain the information from the input sequence. This would probably cause trouble, especially for those long or complicated sequences in which the dimensionality of the representation ingredient is constrained to be the same as for shorter or simpler sequences. Afterward, the proposal of self-attention and transformer models [59] revolutionized the implementation of attention by dispensing

recurrence and convolutions used in RNNs [60] and CNNs [61], respectively, and relying solely on a self-attention mechanism alternatively.

(1) Additive Attention

Additive attention, proposed by Dzmitry Bahdanau [57] and therefore regarded as Bahdanau attention, is the first type of attention. The objective of this type of attention is to enhance the performance of the Seq2Seq model especially in machine translation tasks through aligning the decoder with the relevant inputs and correspondingly employing the attention mechanism. The method in this work, on the other hand, replaces the fixed-length vector with a variable-length one, to improve the translation performance of the basic encoder–decoder model. With time passing by, this method has been implemented more and more often due to its high performance, such as automatic music transcription [182], Fastformer [183], classification [184] and image search [185].

(2) Pointer Networks

Considering those problems, such as sequence sorting, where the sequence size is variable, and diverse COPs cannot be figured out by conventional Seq2Seq and neural turing machines [186], the pointer network was proposed by Vinyals et al. [76] to learn the conditional probability of an output sequence and hence to resolve variable-sized output problems. The attention is utilized as the pointer for choosing a member of the input sequence as the output. Pointer networks can be widely employed in dependency parsing [187], code completion [188], dialogue context generation [189], etc.

(3) Multiplicative Attention

Multiplicative attention is also referred to as Luong attention, proposed by Thang Luong [190]. This attention mechanism was established based on the additive attention mechanism. The differences are as follows: First, the way that the alignment score is calculated; And secondly, the position at which the attention mechanism is introduced in the decoder. The overall architecture of the attention decoder is distinguishable from multiplicative attention since the context vector is leveraged only after the RNN has produced the output for that time step. There are various applications that can be supported by this attention mechanism, such as aspect category and sentiment detection [191], time series forecasting [192], and the prediction of air pollutant concentration [193]. As COPs usually have variable sequence sizes (e.g., when solving TSP formulated as a sequence-to-sequence structure, the number of cites to go is variable, forming a variable input size), pointer networks are used most frequently.

2.2.2. Graphic Neural Networks

Graph structures are common in COPs, so we need to encode graphs reasonably. In addition, the graph-structured representation is more suitable for some COPs than sequence representation. For instance, the permutation invariant property in the TSP cannot be characterized in the sequence representation. GNNs belong to the category of NNs and are widely utilized in DL methods [55], especially in those scenarios where the data are expressed by graph structures. Considering the representation superiority of graph data, people have begun to leverage the characteristics of graphs to design various types of ingenious GNN structures through integrating other techniques, such as graph convolutional networks [194], graph attention networks [195], and graph recurrent networks [196] over different applications, which range from node-type, edge-type, to graph-type predictive assignments. In general, GNNs are customarily employed to resolve those issues in which the graph-structured data are represented. Five typical problem categories are listed as follows.

(1) Graph Classification Problem

Fundamentally, the goal of graph classification is to divide the whole graph into domains with disparate features. In other words, the entire graph is required to be separated

into different segments in this classification scenario. There are numerous applications of this classification, such as document categorizing [197], social recommendation [198], protein function prediction [199], etc.

(2) Graph Grouping Problem

Graph grouping is a class of grouping methods, the task of which is to put together graph data that have similar features [200]. There are also two sub-fields of graph grouping. The first is node clustering, which is based on density. High edge weight or long edge distance corresponds to high density. On the other hand, the graphs are regarded as objects and, hence, grouped based on similarity.

(3) Node Classification Problem

For node classification cases, the nodes in the graph are used to represent the label of samples. The objective of this task is to decide which label each node belongs to via checking their neighbor labels. With each node being tagged a specific label, all nodes in this graph system can be grouped into different classifications. The training approach in this type of case is a semi-supervised learning method [201], where only a portion of nodes in the graph are tagged.

(4) Link Prediction Problem

To predict whether or not two nodes in a graph exist a link is the research objective of this scenario. This function is widely investigated in a recommendation system consisting of friend recommendation [202], movie recommendation [203] and commerce recommendation [204]. Some COPs are well structured and can be reflected in these typical problems, e.g., MIS can be regarded as a node classification problem with two categories representing whether a node is included in MIS. For more COPs, abundant works are reshaping the graph to using GNNs.

2.3. Reinforcement Learning

Many COP environments involve more than one node or executor and can therefore be considered for inclusion in the game theory framework due to their advantage in figuring out multi-player problems. Meanwhile, RL is a reward-oriented approach to help improve the performance of the player or agent. At this point, the two aforementioned techniques are naturally integrated to address COPs.

2.3.1. Single-Agent Reinforcement Learning

RL, especially single-agent RL, is a human-like learning method in which the agent acts like a human and is trained to obtain the expected objective through a series of positive or negative rewards across time [36]. During the learning phase, the RL agent is required to continuously observe the environment and take actions based on its perceptions and rewards offered by the environment. This learning mode makes the agent gradually collect experience and perform better by reinforcing those good behaviors while abandoning the bad ones. In general, RL methods are grouped into two categories, i.e., the model-based RL [62] and the model-free one [63].

In model-based learning approach, the system is able to utilize the predictive model, which is not available in a model-free algorithm, and execute sequential decision making [64]. In contrast to the large amounts of interaction required in the model-free learning, a transparent advantage of model-based learning is the usage of predictive comprehension over the environment model [65].

On the other hand, the model-free algorithm is more flexible since it does not need any prior information about the environment system. All the agents supposed to do is to explore and learn. The model-free algorithm consists of value-based learning and policy-based learning.

Furthermore, Sarsa [66] and Q-learning [67] are the most fundamental online and offline policy learning approaches in value-based learning, respectively. Policy-based learning includes the actor–critic method [68] and the trust region policy optimization (TRPO) method [205], where the policy gradient is implemented, and a continuous strategy is available.

2.3.2. Multi-Agent Reinforcement Learning

Multi-agent RL (MARL) is a sub-field of RL that is becoming increasingly relevant and is extremely impressive. Considering that there exist more than one agent in the systems over enormous applications, such as communication networks [206,207], cyber–physical systems [208,209], financial activities [210,211], and social communities [212], research individuals and groups have started to transfer single-agent RL (SARL) models to the MARL ones. An obvious distinction between SARL and MARL is the number of agents in the environment. Hence, the research objective on MARL is to investigate how multiple agents are going to interact with each other in a common system and how this system evolves [213]. Specifically, the goal of each agent in this multi-agent system is, by treating other agents as part of the environment, to constantly optimize its policy for maximizing the expected long-term rewards.

According to the types of agents, MARL can be generally divided into homogeneous MARL [214], where all agents in the system are homogeneous and able to play an interchangeable role, and heterogeneous MARL [215], in which agents possess different action space and state space.

2.3.3. Multi-Agent Reinforcement Learning with Game Theory

Since the agents in MARL are continuously interacting with the environment and other agents, some interaction modes can be identified and classified, such as cooperative, competitive, or a mix of both. From this moment, researchers have started to integrate MARL with game theory in consideration that game theory is such a study of mathematical models of strategic interactions among rational agents.

(1) Cooperative MARL [216]

In cooperative scenarios, agents are designated as collaborators to achieve the goal of the common system while interacting with the environment. A series of coordination settings, such as sharing sensing information, sharing the same reward function or sharing the policies, are widely investigated and studied [217]. The corresponding applications include rescue operations [218], cooperative exploration [219], resource allocation [220], etc.

(2) Competitive MARL [221]

In those situations with competitive settings, the interests of agents are in conflict. In other words, the more that one or a portion of agents earn, the more others lose, which is also known as a zero-sum game [222]. There are numerous MARL cases with competitive settings that have been studied over the past years, such as Atari games [223], pricing strategies in electronic market [224], sports game [225], etc. Notably, the computational complexities for figuring out two-player and multi-player zero-sum games are quite distinguishable.

(3) Mixed MARL [226]

Mixed MARL is a combination of fully cooperative and fully competitive situations. That is, both cooperative and competitive behaviors are going to appear in this setting, which therefore makes this type of learning architecture notoriously challenging and burdensome to handle [213]. Compared to the above two scenarios, mixed MARL seems not to be so popular due to its complicated optimization procedure and computational process in finding stationary Nash or a related equilibrium. Multi-player poker game [227] and multi-player online battle arenas, such as DOTA 2 [228] and StarCraft 2 [229], are

several classical applications using mixed settings, in which human-level or superhuman-level is achieved.

3. Learning to Solve COPs

Recent works for solving COPs with ML methods are presented systematically in this section. Firstly, they are categorized based on the ML approach: supervised learning, reinforcement learning, and game theoretic methods, basically presented according to the years that they were proposed. Then, the categorization is based on fundamental problems in COPs, including integer programming (ILP), MIS, maximum clique (MC), MVC, and TSP. Figure 2 shows an overview of how ML methods are applied to solve all kinds of fundamental COPs.

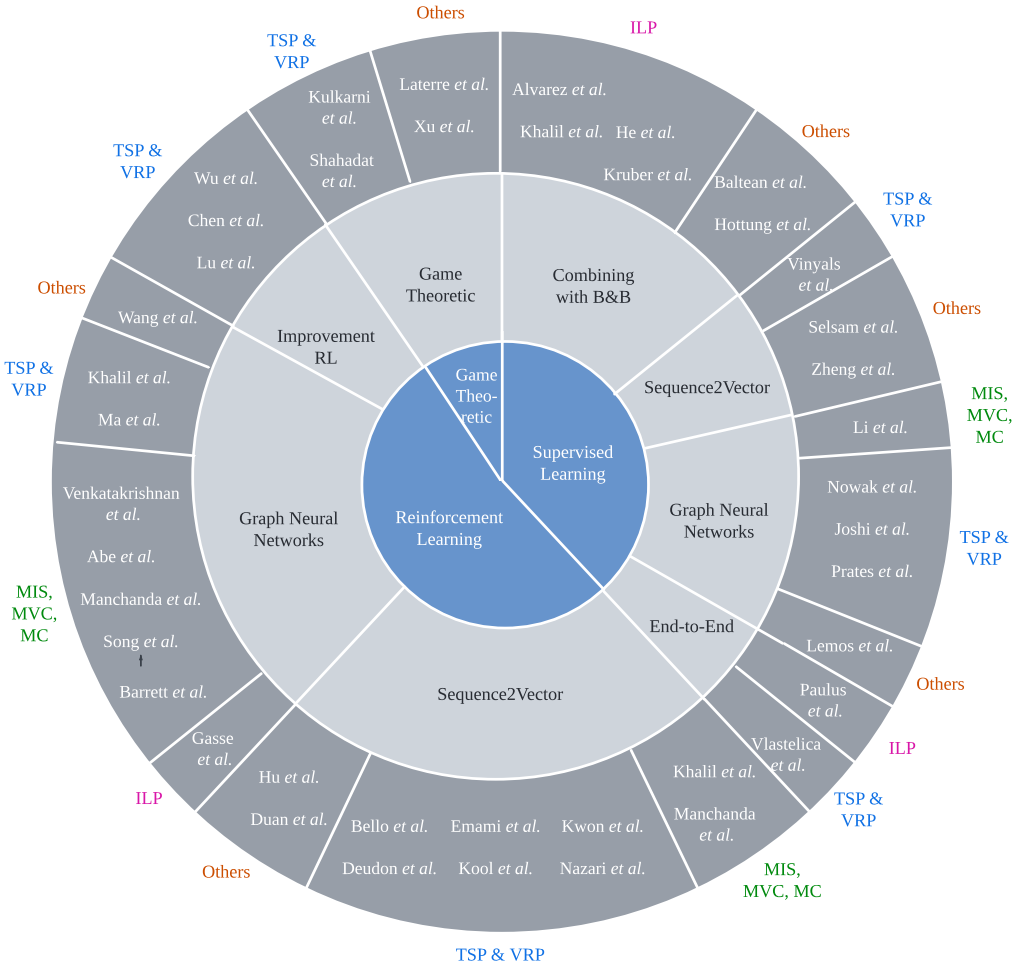

Figure 2. ML methods for COPs. Recent works are categorized firstly based on the ML approach and then the fundamental problems in COPs.

3.1. Methods

3.1.1. Supervised Learning

Combining with Branch and Bound

Traditional approaches for COPs have the same challenges: high computational complexity and dependence on expert knowledge. There have been rich works studying how to replace some components of traditional approaches with supervised learning methods to respond to those challenges. Most of them focus on a classical algorithm called branch and bound (B&B) [69], which is broadly applied for ILP-formed COPs. Table 1 summarizes them.

Alvarez et al. [70] proposed to approximate the scoring strategy of strong branching (SB) with supervised learning for solving mixed ILP problems. Similarly, He et al. [71] proposed to speed up the traditional B&B method with imitation learning. He et al. clearly formulated B&B search as a sequential decision-making process and learned the node selection (prioritizing) policy as well as a node-pruning policy using DAgger. Compared to traditional approaches, this method learns problem-solving patterns from experience, which speeds up the future process. Meanwhile, the sequential decision-making process takes the effects on the future into account when choosing actions. It may be also treated as a simple reinforcement learning approach since it involves the concept of learning from demonstration, but essentially, it directly fits the policy with collected data, which is more like a kind of supervised learning.

Moreover, Khali et al. [72] proposed to solve the scoring problem of SB as a learning-to-rank problem as opposed to a regression or classification problem, and enabled an on-the-fly solver, where the transition from traditional to ML approaches is seamless. That is, instead of learning a policy offline as [70,71], Khali et al. proposed a three-phased online method for a given mixed ILP problem. Firstly, SB is used as the branching strategy for a limited number of nodes. Then, the dataset collected in phase iis fed into a learning-to-rank algorithm. At last, the learned model replaces SB for branching until termination.

Table 1. Supervised learning approaches combining with traditional algorithms, such as branch and bound.

Reference	Year	Advantage or Novelty
Alvarez et al. [70]	2014	approximated SB with supervised learning
He et al. [71]	2014	formulated B&B as sequential decision-making process and learned it with imitation learning
Khali et al. [72]	2016	solved the scoring problem of SB with a learning-to-rank algorithm, online algorithm
Baltean et al. [73]	2018	used supervised learning to select quadratic semi-definite outer approximation of cutting planes
Hottung et al. [74]	2020	learned the container pre-marshaling problem (CPMP)

Baltean et al. [73] applied the above idea to the selection of quadratic semi-definite outer approximation of cutting planes. They used NNs to learn cut selection offline, thereby reducing the computational complexity during testing. Hottung et al. [74] applied the idea for the container pre-marshaling problem (CPMP) (the Container Pre-Marshaling Problem is concerned with the re-ordering of containers in container terminals during off-peak times so that containers can be quickly retrieved when the port is busy), using DNNs to learn the branching strategy and predict bounding bounds.

Sequence2Vector

Sequence-to-sequence is a typical structure of COPs. To deal with those problems, encoders are needed to obtain the vector-form embeddings. Early encoders are RNNs, such as LSTM [75], which can cope with the relationship between an element in the sequence and the previous elements, but RNNs tend to "forget" the information of previous elements when sequences are long. Bahdanau et al. [57] proposed an attention mechanism to solve the problem of long-range dependencies. By calculating the similarity between

two elements, the importance of the previous elements is decided, and the mixed information of the sequence is obtained. That is, the decoder can obtain the information of all states from the encoder instead of the current state. Based on the attention mechanism, Vinyals et al. [76] proposed pointer network (Ptr-Net), which solved the problem that the length of the output sequence depends on the input sequence and became a milestone breakthrough in solving COPs. The main idea of Ptr-Net is to point to each element of the input sequence instead of the mixed information and to obtain the probability distribution over the elements of the input sequence.

Vinyals et al. [76] applied Ptr-Net for the convex hull problem, Delaunay Triangulation, and small-scale planar TSP problems, obtaining higher accuracy than LSTM and LSTM with attention.

Zheng et al. [77] then applied Ptr-Net for the permutation flow shop scheduling problem (PFSP) (the Permutation Flow-shop Scheduling Problem involves the determination of the order of processing of n jobs on m machines). They showed that the average relative percentage deviation (ARPD) of Ptr-Net is better than the LSTM and LSTM with attention. They also found that the element order of the input sequence has some impact on the results.

Graph Neural Networks

Another typical form of COPs is graph based, which characterized the permutation-invariant property of nodes. GNNs [78–80] are a commonly used architecture to encode the input of the graph structure. The input is a vector representing a node or an edge, then the information of nodes and edges is integrated according to the local neighbor structure of the graph, which is used for updating the embedding after. Works related to this approach are summarized in Table 2.

Table 2. Supervised learning methods using graph neural networks.

Reference	Year	Advantage or Novelty
Nowak et al. [81]	2017	encode source and target graphs
Joshi et al. [82]	2017	just encode source graph
Selsam et al. [230]	2018	model SAT as an undirected graph
Li et al. [231]	2018	solve SAT, MIS, MVC and MC with GNNs
Lemos et al. [232]	2019	solve graph coloring problem (GCP) with GNNs
Prates et al. [233]	2019	solve problems involving numerical information

Nowak et al. [81] discussed how to apply GNN [78] to the quadratic assignment problem (QAP) (examples of matching and TSP are given). For two graphs, A and B with the same number of nodes, GNN is used to encode them into two normalized adjacency matrices, E_1 and E_2, respectively, and then the loss related to E_1 and E_2 is calculated for autoregressive learning. For the TSP problem, A and B are the input graph and the ground truth TSP cycle. Finally, the solution is generated by the adjacency matrix obtained from the input graph with beam search.

Instead of using the one-shot autoregressive method as [81], Joshi et al. [82] only used Graph ConvNet to obtain the adjacency probability matrix from the input TSP graph, and then directly calculated the loss related to ground-truth TSP cycle. During the test, Joshi et al. tried three kinds of search methods and compared them: greedy search (greedily selecting the next node from its neighbors with the highest probability), beam search (expanding the b most probable edge connections among the node's neighbors), and beam search with shortest tour heuristic (selecting the shortest tour among the set of b complete tours as the final solution at the end of beam search). Experiments show that

for TSP problems with a fixed number of nodes, the quality of the solution outperforms previous DL approaches.

Selsam et al. [230] solved the satisfiability (SAT) problem with GNNs [78,234]. Due to the permutation invariance and negation invariance properties of SAT, Selsam et al. modeled SAT as an undirected graph with one node for every literal and clause, one edge between each literal and clause, and one edge between literal and complementary literal. Then, the message-passing method [235] is used to update the embedding of each node, and the resulting embeddings are fed into MLP to obtain the literals' votes, which are then used to calculate the loss with the true label. The model parameters of [230] are independent of the problem size due to two aggregation operators: to form the incoming message by summing the messages of one node's neighbors, and to form the logit probability by calculating the mean of literal votes.

Similar to [230], Lemos et al. [232] solved the graph coloring problem (GCP) (the graph coloring problem (GCP): does a graph G accept a C-coloring?) with GNNs [78,80,232]. They took the vertex-to-color adjacency matrix as input, and finally used MLP to obtain the logit probability and calculated the loss with true labels.

Selsam et al. [230] and Lemos et al. [232] successfully solved some NP-hard problems involving symbolic relationships (e.g., whether an edge is connected), and Prates et al. [233] further applied GNNs [79] to solve the NP-hard problems involving numerical information (in this paper, they focused on a variant of TSP: does graph G admit a Hamiltonian path with cost $< C$?). In order to encode the weights of edges, Prates et al. used a vertex-edge adjacency matrix as input, and represented an edge using the concatenation of weight and C, which is a 2-d vector $\in R^2$. Then, similar to [230,232], the refined edge embeddings are fed into an MLP to compute the logit probability.

Li et al. [231] solved four NP-hard problems with GCN [236,237]: SAT, MIS, MVC and MC. Taking MIS as an example, the graph is presented by an $N \times N$ binary adjacency matrix, and then mapped to an N-dimensional real-valued vector through GCN, indicating how likely each vertex is to belong to the MIS. Finally, the solution is obtained from this real-valued vector. A basic method to obtain it is greedy search using the vector as the heuristic function. However, Li et al. pointed out that this search method can become confused when there are multiple optimal solutions for the same graph. Therefore, a further improvement is using GCN to generate multiple probability maps, which is integrated into the tree search procedure (breadth-first account for the diversity of solutions).

End-to-End Architecture

Some recent works focus on how to integrate combinatorial building blocks into neural network architectures as layers so that combinatorial optimization solvers can be embedded into NNs as black boxes, enabling end-to-end optimization.

Vlastelica et al. [83] and Paulus et al. [84] proposed such approaches to integrate ILP solvers. The NN-based solver takes the parameters of the ILP (objective function weight c, constraint parameters A and b) as network weights, inputs an integer solution y, outputs the value of the objective function, and computes loss with the given optimal value. Paulus et al. [84] addressed two main difficulties in achieving the end-to-end optimization: (1) the value of objective function in integer programming is piece-wise and thus non-differentiable; (2) the constraints of ILP are not necessarily active. Vlastelica et al. [83] used the continuous interpolation method to solve the differentiating problem of the objective function weight c and applied this solver for three graph-structure problems: shortest path problem, min-cost problem and TSP. Paulus et al. [84] implemented the differentiation of c as well as A and b using gradient surrogate methods, and conducted experiments on the weighted set cover problem, KNAPSACK, and keypoint matching problem.

3.1.2. Reinforcement Learning

Parameterization of Policy Network

As pointed out by Bello et al. [85], training strategies for COPs with supervised learning require a large number of labels, while the optimal labels for many COPs are not easy to obtain. However, it is relatively easy to evaluate the quality of a given solution. Therefore, Bello et al. proposed to use evaluation as a reward feedback in an RL paradigm to solve the labeling problem, which is another milestone work. More other works follow the same structure and shape the parameterization of the policy network as shown in Table 3, which is the similar concept of an encoder in supervised learning.

Table 3. Works on policy network of reinforcement learning.

Reference	Year	Advantage or Novelty
Bello et al. [85]	2016	used evaluation as reward feedback in RL paradigm
Hu et al. [238]	2017	solved the 3D BPP problem using RL
Nazari et al. [239]	2018	the properties of the input sequence is not static
Khalil et al. [95]	2017	formulated partial solutions as decision sequence
Venkatakrishnan et al. [240]	2018	improved scalability on larger testing graph than training set
Manchanda et al. [241]	2020	billion-sized graphs
Song	2020	combined information from both graph-based representation and ILP-form representation

Bello et al. [85] modeled the 2D Euclidean TSP problem as a sequential decision problem. At each step, a distribution over the next city to visit in the tour is output by the policy, which forms the travel strategy by the chain rule at the terminal step, and the objective is to shorten the travel distance. The policy model is based on Ptr-Net proposed by Vinyals et al. [76] without glimpse, which reads the state s (the city sequence, one city at a time), encodes it as a d-dimensional embedding with an LSTM, and decodes it with an LSTM to obtain the action (the distribution of the next city). Bello et al. used the travel distance to indicate the reward and train the policy with an actor–critic method.

Hu et al. [238] applied [85] to the 3D BPP problem to learn the order of packing items into bins, and proved that it outperforms the well-designed heuristic approach.

A major limitation of [85] is that it assumes the problem is static over time, thus it cannot be used to solve problems, such as VRP, where the properties of the input sequence keep changing. To solve this problem, Nazari et al. [239] proposed a new policy model which leaves out the encoder RNN. They dealt with the static elements and dynamic elements in the input separately, that is, the embeddings of the static elements are input to the RNN decoder and then input to an attention mechanism with glimpses similar to Ptr-Net together with the embeddings of the dynamic elements. Finally, the probability vector of the next destination is output.

For another typical form of CO problem-graph structure, Khalil et al. [95] proposed a training method combining GNNs with reinforcement learning to remove the dependence on sample labels, called S2V-DQN. Khalil et al. presented a partial solution as an ordered list S (a decision-making sequence). At each time step, a node is greedily selected to join S so that the evaluation value Q is maximized. The embedding of each node in list S is obtained by Structure2Vec [242] according to its graph structure (the properties of the node and its neighbor), and Q depends on the embeddings of the candidate nodes and the embedding of the whole graph. Parameters are trained using n-step Q-learning

and fitted Q-iteration. Khalil et al. tested S2V-DQN on MVC, MC, and 2-D TSP, showing its applicability to multiple different problems, graph types, and graph sizes.

Venkatakrishnan et al. [240] focused on the scalability of methods on graph structure problem: can a graph neural network trained at a certain scale perform well on orders-of-magnitude larger graphs [240]? To address this problem, Venkatakrishnan et al. proposed G2S-RNN. Instead of mapping the graph structure to a fixed-length vector to obtain the embeddings of the vertices, G2S-RNN uses a discrete-time dynamical system to process the graph structure. The nodes' embeddings are updated by multiple rounds of convolution, and finally, the entire time-series vector is used as the embedding of the graph, which makes the embeddings' length variable (related to the number of convolution layers). Then, Seq2Vec takes the embedding as input to obtain the vector representation of each node, and Q-learning is used to update the policy.

Manchanda et al. [241] focused on larger graph structures. They tested [95] on MC and show that S2V-DQN fails on graphs beyond 2000 nodes. Based on the observation that, although the graph may be large, only a small percentage of the nodes are likely to contribute to the solution set [241], Manchanda et al. proposed a two-stage method called GCOMB, which uses supervised learning to prune poor nodes and learns the node's embedding with GCN, and searches for solutions on good nodes with Q-learning. The experiment results show that the efficiency is successfully improved on billion-sized graphs.

Song et al. [243] proposed CoPiEr which uses information from both graph-based representation and ILP-form representation. The trajectories are generated by algorithms using two kinds of representations, respectively, following an exchange between two trajectories. Specifically, two trajectories are evaluated by the reward function, and the one with a higher reward is used as a demonstration for the algorithm based on the other trajectory.

Reinforcing Methods

Some works pay more attention to improving the optimization process rather than designing the policy network, which is shown in Table 4.

Table 4. Works on reinforcing methods of reinforcement learning.

Reference	Year	Advantage or Novelty
Deudon et al. [86]	2018	enhanced the framework with 2-opt
Emami et al. [87]	2018	learned permutation instead of decision sequences point by point
Kool et al. [88]	2018	re-designed the baseline of REINFORCE algorithm as the cost of a solution from the policy defined by the best model
Ma et al. [244]	2019	solved COPs with constraints
Abe et al. [245]	2019	used CombOpt inspired by AlphaGo Zero to replace Q-learning
Barrett	2020	explored the solution space during test time
Kwon et al. [246]	2020	handled equally optimal solutions

Based on Bello et al. [85], Deudon et al. [86] proposed a policy network with only attention mechanism and enhanced the framework with 2-opt. Instead of using LSTM, the decoder maps three last sampled actions (visited cities) to a vector, and 2-opt is used to improve the output solution.

The previous idea of using reinforcement learning to solve COPs is usually to generate decision sequences point by point, which is a greedy heuristic. On the contrary, Emami et al. [87] proposed the Sinkhorn policy gradient (SPG) algorithm to learn permutation so that in addition to TSP-type problems, it can also solve other issues, for example, maximum weight matching. The state of SPG is a permutation instance of size N, and the action is an $N \times N$ permutation matrix. To use a deterministic policy gradient method, such as DDPG, Emami et al. used Sinkhorn layers to approximate the permutation matrix and added a term to the critic loss to de-bias the policy gradient.

Kool et al. [88] improved [85] on two aspects: (1) They used an attention mechanism instead of LSTM for encoding, which introduces invariance to the node input order and enables parallel computation. (2) They designed the baseline of REINFORCE algorithm as the cost of a solution from the policy defined by the best model, which is similar to self-play that improves itself by comparing with the previous solutions.

Ma et al. [244] focused on COPs with constraints, for example, TSP with time window (TSPTW), and proposed to solve them using hierarchical RL. Ma et al. compared two hierarchical structures: one is that the lower layer is to make the solution satisfy the feasible region constraints, and the higher layer is to optimize the objective function; the other is that the lower layer is to optimize the solution of an unconstrained TSP, and the higher layer is to give an instance of TSP with constraints. They proved that in the TSP experiment the first structure is better. Ma et al. also proposed graph pointer network (GPN), which uses LSTM to encode a graph with context provided by GNN to obtain more transferable representations.

Abe et al. [245] addressed the problem that previous methods obtained poor empirical performance on some graphs due to the limited exploration space of the Q-learning method. They proposed CombOpt inspired by AlphaGo Zero to replace Q-learning. While the original AlphaGo Zero is designed for two-player games with a terminal state of win/lose/draw, Abe et al. modeled the state value of AlphaGo by "how likely the player is going to win", and extended it to COPs by a reward normalization technique relating to "how good it is compared to random actions".

Barrett et al. [247] addressed the necessity of exploration of the solution space during test time since many COPs are too complex to learn only through training. They designed a reward structure to motivate the agent to find better solutions when testing by adding or removing a node. This method is complementary to many training methods.

Kwon et al. [246] addressed the problem that for some COPs with equally optimal solutions, previous works always learn one of them, which is a biased strategy. They pointed out that previous works are sensitive to the selection of the first node, thus they used the idea of entropy maximization to improve exploration. In addition, they proposed a new baseline for policy gradient to avoid local minima.

Improvement RL

The previous works usually use NNs to generate COP solutions in one shot. Despite the efficiency, there is always a certain gap from the optimal in such methods. Another type of work learns the improvement heuristic, which improves an initial solution iteratively.

Chen et al. [97] proposed a method of making incremental improvement until convergence by locally rewriting feasible strategies, called NeuRewriter. Chen et al. modeled the rewriting process as MDP, where the state is the partial solution, the action is the desired local region and the associated rewriting rule. They trained the policy with the advantaged actor–critic algorithm and applied NeuRewriter to three problems—expression simplification, online job scheduling, and vehicle routing problems (VRP)—where NeuRewriter performs better than strong heuristics.

Wu et al. [248] proposed to use deep RL to automatically improve an initial solution based on neighborhood search. The architecture is based on self-attention, where the action is a node pair for pairwise local operators. Learning with the actor–critic algorithm, the model generates a probability matrix for each pair.

Lu et al. [249] proposed a new framework to improve heuristic methods with reinforcement learning for capacitated vehicle routing problems (CVRP). They divided heuristic operators into two classes, improvement operators and perturbation operators. According to each state, a certain class of operators is selected first, and then a specific operator is selected within the class. Specifically, given recent solutions, the method for selecting a class is a threshold-based rule. If a local minimum is reached, the perturbation operator is activated to generate a new solution, otherwise, the improvement operator is activated. They used reinforcement learning to select the specific improvement operator, where the problem, solution, and running history are used as the features of the state, and different operators are used as actions. Lu et al. designed two kinds of rewards: one is +1 if the solution is improved, otherwise -1, and the other is to take the total distance achieved by the problem instance during the first improvement iteration as a baseline.

3.1.3. Game Theoretic Methods

Single-Player Game

Some COPs are modeled as reinforcement learning problems, in which the idea of single-player games and self-play can be applied.

Laterre et al. [89] used the concept of self-play on 2D and 3D BPP problems and proposed the ranked reward algorithm. When applying MCTs to solve a BPP problem modeled as MDP, the reward for all non-terminal states is usually set to 0 and for the terminal state, is obtained according to the quality of the solution. Laterre et al. reshaped the reward function by storing the agent's recent performance against which new solutions are compared, earning the reward related to whether or not it outperformed itself. In this approach, ranked reward reproduces the benefits of self-play for single-player games, removes the requirement of training data, and obtains a well-suited adversary. Experiments show that it outperforms ranked-free strategies on BPP problems.

Xu et al. [250] proposed a method to transform certain COPs into Zermelo games to use self-play-based neural MCTS. The Zermelo game is a two-player, finite information game with perfect information. During the game, two players alternate actions, and finally one wins and one loses. Xu et al. modeled a type of COP (the highest safe rung (HSR) problem (Highest safe rung (HSR): consider throwing jars from a specific rung of a ladder where the jars could either break or not. If a jar is unbroken during a test, it can be used next time. The highest safe rung is a rung that for any test performed above it, the jar will break.)) as a Zermelo game, mainly using its recursive property. The state–space convergence and solution quality of the algorithm prove to be good, but it faces the common time-consuming problem of the neural MCTS algorithm.

Drori et al. [90] modeled some COPs of choosing edges over graphs as a single-player game, represented by a decision tree. Drori et al. used the graph attention network to encode the line-graph variant of the original graph, and then used the attention mechanism to select actions (add/remove an edge). Finally, they updated the network weights through the evaluation oracle of the leaf nodes, where perfect information game theory proved that a similar mechanism could approximately converge to an optimal policy. Drori et al. conducted experiments on four problems of minimum spanning tree, shortest path, TSP, and VRP, and demonstrated linear running times with good optimality gaps.

Competitive Game

Shahadat et al. [91] regarded each strategy of the TSP problem as a player and iteratively updated them through competition and reinforcement to obtain the optimal strategy. Specifically, Shahadat et al. first randomly initialized some strategies, each of which has a payoff equal to 0. In each round of iteration, for the player X_i, an opponent player X_j is chosen to compare; if X_j wins, then X_i imitates the strategy of X_j (copy some sequence from the opponent) to update itself, and rewards the winning player according to probabilistic utility theory, otherwise, X_i updates itself according to the current global best player and the local best player (the strongest strategy that X_i has explored).

If the update does not exceed its previous performance, the strategy of X_i is updated again with a local search strategy 2-opt.

Cooperative Game

In tasks involving cooperation among multiple agents, the concept of equilibrium and communication in game theory tends to lead to good solution quality.

Kulkarni et al. [92] proposed to use probability collectives to solve the small-scale multi-depot multiple traveling salesmen problem (MDMTSP). This method is a kind of cooperative approach, and the main idea is to treat each salesman as an agent, and each route as a strategy that forms an agent's strategy set. In each iteration, each agent independently updates the probability distribution over its own strategy set to select a specific action that has the highest probability of optimizing its own utility (private utility) until the probability distribution can no longer be changed (to reach Nash equilibrium). As a cooperative approach, the private utility is allocated by the global utility, so the global objective is optimized in each iteration, too. In order to deal with the problem of node-repeated-selection in TSP, Kulkarni et al. also used heuristic methods, including node insertion and elimination, and neighboring and node swapping. This method solves the MDMTSP of 3 depots, 3 vehicles, and 15 nodes in a short time, but Kulkarni et al. also pointed out that it can only solve small-scale problems with few constraints.

3.2. Problems

In this section, the historical solution process of classical COPs are concerned, focusing on the change of methods and the data scale of the experiments, so as to guide application to practical problems. It can also reflect how the ML algorithms proposed in recent years move the field forward. Some problem-specific solutions not mentioned in Section 3.1 are added.

3.2.1. Integer Linear Programming

ILP is a general form for many discrete COPs and is of great practical significance. Table 5 presents ML methods proposed in recent years to solve the ILPs.

To solve MILP problems, B&B is a common approach, which builds an optimization tree and selects variables for branching at each step, and prunes according to the results of LP-relaxation. In the branching stage, strong branching (SB) gives the score of candidate nodes to guide the selection. Alvarez et al. [70] proposed to treat the scoring stage of SB as a regression problem and used supervised learning for fitting to solve binary MILP problems. The method effectively improves operation efficiency. He et al. [71] formally modeled the decision process of B&B as MDP, and treated node selection and node pruning as classification problems. They set the label of expand and prune according to whether the node is optimal, and then used the imitation learning method to deal with the problem of imperfect information. On larger-scale problems, the method of He et al. further reduces the computational time and the gap with the optimal solution on MILP problems. Khalil et al. [72] further regarded the SB scoring problem as a learning-to-rank problem where nodes are labeled by binary ranking, that is, nodes with scores close to the highest score are ranked higher than others. The ranking strategy is learned by supervised learning methods. In addition, the method proposed by Khalil et al. seamlessly transfers the branching strategy from the SB- to ML-based method when applied to a given problem, instead of learning the strategy offline. Gasse et al. [93] also formulated B&B as MDP and learned the policy by behavioral cloning. They treated the binary tree as a bipartite graph and used GCNN to encode the state to improve the generalization ability of the algorithm on test instances.

The work of Kruber et al. [251] concerned the issue of the decomposition of MIPs. A MIP problem can be decomposed in many ways, and some of them can well reflect the structure of the original model for which the MIP can be reformulated to an easier form to solve, while some decompositions are not suitable. Kruber et al. proposed to use supervised learning to train a classifier to decide whether a given decomposition of MIP is

worth continuing to be reformulated. Kruber et al. tested the classifier on both structured instances (such as set covering, BPP, VRP, etc.) and unstructured instances with a total of 1619 decompositions.

Table 5. Methods for ILPs.

Method	Reference	Year	Description	Scale
SL	Alvarez [70]	2014	learning SB scoring decisions on binary MILP	hundreds variables, 100 constraints
	He [71]	2014	using Dagger to learn binary branching and pruning policy on MILP	200–1000 variables, 100–500 constraints
	Khalil [72]	2016	learning ranking policy derived by SB, on-the-fly learning	50,000 and 500,000 nodes
	Kruber [251]	2017	to decide whether a decomposition of MIP is suitable	–
	Gasse [93]	2019	using GCNN to encode the bipartite graph	–
	Paulus [84]	2021	an end-to-end trainable architecture to learn constraints and cost terms of ILP	1–8 variables, 2–16 constraints
RL	Tang [94]	2020	learning to cut plane with RL on IP	variables × constraints = 200, 1000, 5000

Paulus et al. [84] directly used the mathematical gradient substitution method to enable the loss between the objective function and the optimal objective function to be passed to the parameters c of cost function and the parameters A and b of the constraints, thereby realizing an end-to-end black-box solver. Paulus et al. showed high accuracy of this method on very small-scale ILP problems.

Tang et al. [94] used reinforcement learning to improve the cutting plane method (specifically, Gomory's method [252]) for solving IPs. Tang et al. modeled the process of selecting cutting planes as an MDP, where the state is the current feasible region and the reward is the change in the value of the objective function between episodes. In order to deal with the problem that the policy is independent of the order of the constraint parameters and the input length is variable, Tang et al. used the attention network and LSTM, respectively. In the experiment, Tang et al. considered problems of three scales with the number of variables, and the number of constraints being about 200, 1000, and 5000.

3.2.2. MIS, MVC, MC

MIS, MC, and MIS problems are COPs over graphs and are equivalent problems mathematically, thus, the methods for these problems are similar. Table 6 represents the surveyed works on this problem.

Table 6. Methods for MIS, MC, MVC.

Method	Reference	Year	Problem	Description	Scale (Vertice)
SL	Li [231]	2018	MIS, MC, MVC	using GCN to generate multiple probability maps and doing tree-search on them	1000–100,000
RL	Khalil [95]	2017	MC, MVC	using S2V-DQN to select desired node once a time	50–2000
	Venka-takrish-nan [240]	2018	MIS, MC, MVC	using Graph2Seq to handle variable graph size trained by Q-learning	25–3200
	Abe [245]	2019	MC, MVC	using CombOpt to solve the problem of limited exploration space	100–5000
	Barrett [247]	2020	MC	using ECO-DQN to improve the solution during test time	20–500
	Song [243]	2020	MVC	co-training algorithms with graph-based and ILP-based representa-tions	100–500
	Manch-anda [241]	2020	MC, MVC	learning to prune poor nodes in order to generalize to larger graph	50 K–65 M
	Karal-ias [96]	2020	MC	traing GNN in unsuper-vised way by constructing a differen-tiable loss function	up to 1500

With the development of deep networks, GNNs are widely used to solve problems based on graph structures, including COPs. Khalil et al. [95] used GNNs to solve MC and MVC firstly. As described in Section 3.1.2, they proposed to select the desired node once a time, using the Structre2Vector network to embed nodes that are learned by Q-learning, and tested this method on graphs with 50–2000 nodes. Li et al. [231] regarded the output of GCN as an n-dimension real-valued probability map. Instead of converting the probability map to the discrete solution directly, they proposed to generate multiple maps and used a breadth-first tree search to obtain the complete solution. They showed that this method generalized well on graphs with 1000 nodes to 100,000 nodes and 10 million edges. Venkatakrishnan et al. [240] proposed Graph2Seq-RNN based on [95], which uses a discrete-time dynamical system to deal with the graph and enables the network to perform well on graphs with different sizes from the trained one. Experiments are performed on graphs with 25–3200 vertices and various edge probabilities, notably, the policy is trained on graphs of size 15 and edge probability 0.15. Abe et al. [245] proposed CombOpt to solve the problem of limited exploration space in the Q-learning method, which uses a reward normalization technique relating to comparison with random actions. The training set includes graphs with 40–50 nodes for MC and 80–100 nodes for other problems, and the testing set includes graphs of various sizes from 100 to 5000. Manchanda et al. [241] extended the method of [95,231] to graphs with 50 K to 65 M nodes with up to 1.8 B edges by learning to prune poor nodes that have little contribution to the solution set. Karalias et al. [96] proposed a principle method to construct the differentiable loss function from discrete problems, and trained a GNN to minimize this loss function in an unsupervised way, following selecting a node according to the vector output by GNN.

Rather than improve the training process, Barrett et al. [247] focused on improving the solution during test time. They proposed ECO-DQN with a reward structure to guide exploration, that is, adding or removing a node from the solution. They tested this method on graphs with 20–500 vertices and showed that it beat S2V-DQN in most cases.

Song et al. [243] used both graph-based and ILP-based representations of COPs. They solved a problem with two algorithms based on two kinds of representation separately and exchanged the information between two solutions. Experiments were performed on graphs of sizes 100–500 and showed good generalizing ability and solution quality.

3.2.3. Traveling Salesman Problems

Due to the natural structure of the TSP, it is easy to treat the strategy as a decision sequence, where nodes are selected one by one and form a route. There are two ways to deal with the input structure: a sequence or a graph. Generally, the graph representation characterizes the structure better since the order of depots in input has the property of being permutation invariant. Recent works on this problem are presented in Table 7.

The first attempt to apply the sequence-to-sequence model into COPs is made by Vinyals et al. [76]. They proposed Ptr-Net based on the attention mechanism, which enables a variable length of input and is trained with supervised learning. Nodes are selected according to the probability distribution over candidate nodes set as output by Ptr-Net. They tested it on small-scale TSPs with fewer than 50 nodes and found that Ptr-Net seems to break for 40 nodes and beyond. Bello et al. [85] improved [76] by learning the probability distribution with REINFORCE, removing the reliance on training data. Active search was used to construct the final solution. Experiments were performed on TSP20, 50, and 100. Refs. [86–88,239,246] used similar architecture based on S2S. Emami et al. [87] proposed to learn the permutation matrix of nodes instead of the probability sequence using the Sinkhorn policy gradient and test it on TSP-20, obtaining better performance. Kwon et al. [246] proposed POMO to solve the problem that equivalent optimal solutions are unequally selected by previous methods.

To solve problems such as VRP where the input status changes over time, Nazari et al. [239] proposed a variant of [85] which embeds the static elements and dynamic elements in the input separately. Experiments suggest this method can obtain better results than classical heuristics.

Nowak et al. [81] proposed to apply GNN to COPs. It is trained in a supervised such way that the loss between the bedded origin graph and target graph with the same number of nodes is calculated. Nowak et al. tested the method on TSP20 and mentioned a performance gap with Ptr-Net [76]. Khalil et al. [95] further proposed S2V-DQN to remove training data, as described in Section 3.1.2. Joshi et al. [253] improved [81] by replacing the self-regression approach and compared different search methods. Prates et al. [233] focused on COPs with numerical information (in this case, TSP with cost constraints) and proposed to embed edges with weights and constraints and trained the GNNs with labels. Ma et al. [244] also focused on COPs with constraints and proposed to use the graph pointer network and hierarchical RL architecture, where two levels aim at optimization and making solutions feasible, respectively.

Table 7. Methods for TSP.

Problem	Method	Reference	Year	Description	Scale
TSP	S2V	Vinyals [76]	2015	Ptr-Net	<50
		Bello [85]	2016	Ptr-Net + REINFORCE	20, 50, 100
		Deudon [86]	2018	attention+REINFORCE+2-opt	20, 50, 100
		Emami [87]	2018	attention+Sinkhorn Policy Gradient	20
		Kool [88]	2019	Transformer+REINFORCE with baseline relating to the best model so far	20, 50, 100
		Kwon [246]	2020	attention+REINFORCE	20, 50, 100
	G2V	Nowak [81]	2017	GNNs	20
		Khalil [95]	2017	GNNs + Q-learning, S2V-DQN	50–2000
		Joshi [253]	2019	GCNs + beam search	20, 50, 100
		Prates [233]	2019	GNNs, message passing	20–40
		Ma [244]	2019	Graph pointer Network+hierarchical RL	250, 500, 750, 1000
	improve-ment RL	Wu [248]	2021	transformer+AC	20, 50, 100
	end-to-end	Vlaste-lica [83]	2020	an end-to-end trainable architecture to learn cost terms of ILP	5, 10, 20, 40
	game	Kulkar-ni [92]	2009	Probability Collectives	3 depots, 3 vehicles
		Shaha-dat [91]	2021	competitive game among strategies	-

Table 7. *Cont.*

Problem	Method	Reference	Year	Description	Scale
VRP	S2V	Nazari [239]	2018	RNN + attention + policy gradient	10, 20, 50, 100 customers, 20, 30, 40, 50 vehicle capacity
		Kwon [246]	2020	attention + REIN-FORCE	20, 50, 100
	improve-ment RL	Chen [97]	2019	using NeuRewriter to find a local region and the associated rewriting rule	20, 50, 100
		Lu [249]	2020	use RL to choose operators	20, 50, 100
		Wu [248]	2021	transformer + AC	20, 50, 100

Some RL improvement methods are proposed to solve TSP. Chen et al. [97] proposed NeuRewriter to learn region-picking and rule-picking policies and improved a feasible solution by locally rewriting. They tested NeuRewriter on Capacitated VPR of sizes 20, 50, and 100. Lu et al. [249] proposed to use RL to choose heuristic improvement operators, and the other class of operators, namely perturbation operators, was used to avoid the local optimum. Wu et al. [248] proposed an attention mechanism-based architecture to choose a node pair for pairwise local operators without human guidance, which outperforms other deep models on TSP and CVRP instances.

Game theoretic methods have also been applied in TSPs. Kulkarni et al. [92] proposed to solve MTSPs using probability collectives, where agents try to maximize their own utility, which is assigned by a global utility independently. They obtained good solutions in a short time on small-scale MTSPs. Shahadat et al. [91] created a competitive game among strategies over TSPs. At each step, two strategies were compared, and the winner was used as a demonstration to the loser.

4. Applications in Energy Field

This section mainly focuses on applications in problems in the energy field [254]. Applications in several aspects were surveyed, including petroleum supply chain, steel-making, electric power system, and wind power. The results of the review are reported, respectively, as follows.

4.1. Petroleum Supply Chain

Three scenarios of the petroleum supply chain were focused on: refinery production planning, crude oil scheduling, and oil transportation, each of which plays a significant role in the supply chain by a sequence of operation processes. Then a brief review of progress on application to solutions of the aforementioned problems in the petroleum supply chain was presented, for these processes are major concerns of worldwide petroleum supply systems. Table 8 gives a brief summary of applications surveyed.

Table 8. Application in petroleum supply chain.

Senario	Problem	Reference	Year	Approach
Refinery Production Planning	refinery product profits	[99]	2007	stochastic algorithm
	strategic refinery production planning	[98]	2017	Cournot oligopoly-type game
	refinery operation problem	[100]	2017	Cournot oligopoly model
	gasoline industry investment	[101]	2020	three-phase Stackelberg game
Refinery Scheduling	refinery production and operation	[105]	2009	DP + mixed genetic
	crude oil scheduling	[104]	2010	fuzzy and chance-constrained programming
	refinery planning and crude oil operation scheduling	[255]	2011	Lagrangian decomposition
	scheduling refinery problem	[256]	2011	logic-expressed heuristic rules
	oil-refinery scheduling	[102]	2015	heuristic algorithm
	refinery crude oil scheduling	[257]	2020	line-up competition algorithm
	crude oil operation scheduling	[258]	2020	NSGA-III
	crude oil supply problem	[259]	2020	MILP clustering
	tank blending and scheduling	[260]	2020	discretization-based algorithm
	oil blending and processing optimization	[261]	2020	discrete-time-presented multi-periodic MILP model
	crude oil refinery operation	[262]	2020	unit-specific event-based time representation
Oil Transportation	refinery product profits	[99]	2007	stochastic algorithm
	long-term multi-product pipeline scheduling	[10]	2014	MILP-based continuous-time approach
	multi-product treelike pipeline scheduling	[12]	2015	continuous-time MILP
	long-distance pipeline transportation	[263]	2015	outer-approximation-based iterative algorithm
	crude oil pipeline scheduling	[264]	2016	two-stage stochastic algorithm
	pipeline scheduling	[11]	2017	SM + ACO
	fuel replenishment problem	[107]	2020	adaptive large neighborhood search
	refined oil pipeline transportation	[265]	2020	parallel computation + heuristic rules + adaptive search
	refined oil transportation	[266]	2021	improved variable neighborhood search

4.1.1. Refinery Production Planning

Refinery production planning is the precondition of all the operations, as it defines the number of end products to be produced and predicts the total profits of the whole production process. The ML model based on game theory was investigated to deal with refinery production planning problems.

Tominac et al. [98] presented a structure based on game theory according to a Cournot oligopoly-type game to solve strategic refinery production planning problem, as the model involves multiple refineries and markets. The results show rational and robust decisions and are mutual best responses in the competitive planning game. Ravinger [99] built a Cournot oligopoly model embedding a stochastic algorithm to cope with product profits problem for refineries. This model with an oligopolistic market structure takes into account key characteristics of refineries and yields a solution concerning investment decisions under demand shock. Tominac [100] also constructed a Cournot oligopoly model for refinery operation problems in multiple markets competition, which is an optimal strategic production planning problem. The model takes into account both economic objectives and process constraints to tackle such MILP. On the aspect of economy, Babaei et al. [101] used a three-phase Stackelberg game theory approach based on a multi-agent method to analyze the gasoline industry investment by the government under massive consumption and shortage of production in developing countries. The results show that profit is maximized for investors under the management of production volume.

From producing point of view, the ML approach based on game theory regards each refinery as a single rational agent, where each considers both their profits and integrated profits in the game to achieve Nash equilibrium. Therefore, with cooperation, i.e., transporting products to each other refinery, the whole supply system maximizes profit. However, transportation costs cannot be easily considered in this game model since none of the agents would be willing to take the risk. Therefore, dealing with transportation costs in the ML approach based on game theory would be another challenge.

4.1.2. Refinery Scheduling

Refinery scheduling is a significant production operation in the upstream petroleum supply chain in which many complex chemical sub-operations are involved, including crude oil unloading, crude oil mixing, production unit operation, products blending, and refined oil distribution [102]. These scheduling optimization problems are stochastically uncertain and have multiple constraints and objective functions [103]. It is difficult to optimize the model with such complexity, not to mention large-scale scheduling problems. Therefore, different models and algorithms with built-in ML mechanisms were surveyed, where each approach either simplifies the model or reduces the time of solving.

Crude oil scheduling sometimes can involve uncertain conditions, especially when dealing with product demands or ship arrival time at the terminal. To solve crude oil scheduling problems under uncertain conditions, Wang et al. [104] developed a two-stage robust model which transforms fuzzy programming and chance-constrained programming into deterministic counterpart problems at the first stage and proposes the second stage based on the scenario. The experiment shows that the feasible rate raised from 32% to 97% compared to the approach proposed by Cal et al. Li et al. [105] suggested a hybrid mechanism combining DP and mixed genetics to improve the general DP algorithm, which is computationally expensive to solve the model with inequality constraints. The algorithm successfully solves multi-stage production and operation DP problems in refineries under uncertain market demand and yields an adaptive and effective solution.

Among all the literature works investigated, many applications of heuristic aim to simplify the model and thus make the algorithm computationally cheaper to obtain the optimization. Shah et al. [102] proposed a heuristic algorithm based on a decomposed network to solve the oil-refinery problem. The authors decomposed the problem into two separate scheduling problems and generated multiple integer cuts at the end of each iteration, which significantly reduces the computational time due to fewer iterations being needed.

Similarly, A MIP model integrating logic-expressed heuristic rules was reported by Li [256] to simplify the model of the scheduling refinery problem as well as to improve the efficiency of the solution without losing optimization guaranteed. Yue et al. [257] first proposed a heuristic rule of the oil transport sequence. Then an asynchronous continuous-time CO model based on a line-up competition algorithm was built to transform the original refinery crude oil scheduling problem (RCSP), a MINLP, into a CO problem that makes it easier to obtain the optimal solution, and then solve it within a relatively short computational time. The total cost of scheduling, the objective function, is proved to reduce by 2.1% compared to other approaches mentioned. To solve the crude oil operation scheduling problem for refineries, which is a multi-objective optimization problem, an NSGA-III based method was proposed by Hou et al. [258]. As a meta-heuristic algorithm, NSGA-III improves population diversity by adaptively updating reference points compared to NSGA-II.

There are some sources involving the use of other techniques. Assis et al. [259] presented a MILP clustering mechanism to cope with the operational management of the crude oil supply (OMCOS) problem consisting of the scheduling of vessel traveling between a terminal and platforms. The authors used the clustering solution as a pre-step to simplify operations and reduce the total number of vessel routes. Beach et al. [260] proposed a discretization-based algorithm which can approximate non-convex mixed-integer quadratically constrained programming (MIQCP) as a MILP. To solve the tank blending and scheduling problem, the authors combined the algorithm with a rolling horizon approach, which is evaluated to be supportive of using industrial datasets. Li [261] reported a discrete-time-presented multi-periodic MILP model for oil blending and processing optimization problem. Shown by numerical results, this formulation is computationally effective, as it reduces solving time. Mouret et al. [255] proposed an approach involving Lagrangian decomposition to solve refinery planning and crude oil operation scheduling integration, which is a large-scale MINLP. The authors introduced a hybrid dual problem to update Lagrange multipliers and then solve each problem separately. Bayu et al. [262] reported a unit-specific event-based time representation based on the state task network (STN), an extension of a previous benchmark given by Yadav and Shaik, to solve crude oil refinery operation involving desalting. The proposed model makes it possible to inform the decision maker of resources consumed, such as wash water and sludge.

To conclude this section, the built-in ML algorithm could help simplify the COP model or reduce the solving time with certain architecture constructed. Yet not every part of the production process was considered for refinery scheduling, i.e., the omitted elements, such as dependence on uncertain parameters and uncertainties under non-linear constraints, would cause the results to be different.

4.1.3. Oil Transportation

Now it is time to focus on the intermediate operation that connects upstream and downstream of the petroleum supply chain: oil transportation. The oil transportation system consists of several methods among crude oil fields, refineries, storage terminals, and sales companies: pipeline, trucks, railway, and tankers [106]. The determination is made with respect to different factors, such as distance, oil type, cost, etc. In this section, applications to problems concerning pipeline scheduling were mainly investigated.

Oil transportation through pipeline usually involves multi-product scheduling problems. Mostafaei et al. [10] introduced a MILP-based continuous-time approach for the long-term multi-product pipeline scheduling problem, which is a MINLP. Compared to work performed by Cafaro et al., the formulation proposed by Mostafaei et al. allows 25% less total operation cost. Focusing on the same topic, Mostafaei et al. [12] also presented their work of a continuous-time MILP to address the scheduling of multi-product treelike pipeline scheduling consisting of a single refinery and several downstream depots. The proposed model is proved to be significantly more computationally efficient. Like Mostafaei et al., who focused on pipeline scheduling between a single refinery and multiple terminals, Zhang et al. [11] also concentrated on multiple pump stations, which

is a crucial structure for long-distance pipelines. The authors presented a hybrid computational approach embedding SM into ACO. Therefore, with such an architecture, two sub-models are more interactive than other multi-step models in previous literary works.

Now, the reader may look at other approaches. Oliveira et al. [264] presented a two-stage stochastic MILP model to determine the scheduling of pipeline-transported oil and sequence of ships at the terminal, which is a system with supply uncertainty. Zhang et al. [263] developed a continuous-time scheduling model embedding an outer-approximation-based iterative algorithm to tackle the long-distance pipeline transportation problem. Gao et al. [266] proposed an improved variable neighborhood search algorithm (IVNS) for scheduling refined oil transportation with multiple trips and due times. To improve local optimization ability, a greedy strategy is built based on the initial solution obtained by the forwarding insertion heuristic. Therefore, the results converge faster without losing the quality of the solution. Wang et al. [107] constructed an adaptive large neighborhood search (ALNS) heuristic to cope with the fuel-replenishment problem (FRP), a VRP with multiple deliveries, trips, and compartments, where different products are involved. Wang [265] proposed an improved simulated annealing (ISA) algorithm incorporating parallel computation, heuristic rules, and adaptive search to tackle refined oil pipeline transportation problems in order to optimize computation and special constraints. To simplify the structure, the author presented an approach that separates one pipeline transportation scheduling problem into initial-station input scheduling and sub-problems of distribution scheduling at each station along the pipeline. The resulting scheduling plan was proved to be efficient and effective applying to a real instance. Based on previous research of a petroleum products distribution system, where an object-oriented Petri nets (OOPN) framework is proposed, Li et al. [267] presented a queue theory to improve the safety of the existing model, for the OOPN model only shows boundedness and no deadlock without evidence of safety. The improved model was shown to be effective.

So far, the references surveyed in this section were all focused on treelike pipelines with multiple destinations, i.e., the models were built on a single refinery concerning several terminals, while in reality, the pipeline for oil transportation would be more like a complicated network. With such complexity, whether the current algorithm would still be competent is worth questioning.

4.2. Steel-Making

The scheduling problem in steel making is another complicated combinatorial problem in the industrial field. Like petroleum refinery scheduling, steel-making scheduling also involves numerous chemical operations and materials blending. Table 9 gives a brief summary of the applications surveyed.

To deal with dynamic uncertainty involved in steel-making workshops, Lin et al. [108] introduced a deep RL-based algorithm to the crane scheduling problem. Then a DQN algorithm was built into the crane action value network model. The resulting scheduling showed that the task completes faster using this model, thus improving efficiency. Zhou [109] proposed an improved gray wolf optimization algorithm based on a deep deterministic strategy gradient algorithm (DDPG-GWO) to deal with problems such as the local optimum and poor stability of the solution without using the gray wolf algorithm. DDPG can help train the agents, thus avoiding the above problems. It is proven that DDPG-GWO can obtain a more accurate solution in less computational time. Jia [110] aimed at solving batch machine scheduling problems and presented a batch optimization algorithm based on Ptr-Net, which is trained by RL. To improve the performance of Ptr-Net, a hybrid genetic algorithm is introduced, due to its capability for global search. As a crucial link during the steel-making process, the operation and scheduling problem of steel-making and continuous casting has complicated constraints with uncertainty. To realize intelligent optimization, Ma et al. [111] discussed an integrated framework based on ML with rolling optimization algorithm and rule mining.

There were not enough clues to show the application of the algorithm based on game theory that was used in steel making. Most of the references surveyed were relevant to the ML-based algorithm with modification to approaches to solve COPs.

Table 9. Application in steel making.

Problem	Reference	Year	Approach
crane scheduling problem	[108]	2021	deep RL-based algorithm + DQN
scheduling of steelmaking and continuous casting	[109]	2021	DDPG-GWO
batch machine scheduling	[110]	2021	Ptr-Net
steel-making operation and scheduling	[111]	2022	integrated framework + ML

4.3. Electric Power System

The scheduling problem in electric power systems is another industrial field that draws attention. Similar to the petroleum supply chain, the electric system is also a multivariate non-linear system with high uncertainty and complexity. Nevertheless, unlike liquid or gas-formed petroleum products, this power cannot be stored due to the special physical feature of electricity. Therefore, solving such a system requires more consideration and is thus more complicated [268]. Table 10 gives a brief summary of the applications surveyed.

ML-based approaches were first surveyed to solve CO scheduling problems in the electric system. Yan et al. [112] proposed an improved Ptr-Net combining the deep RL algorithm to obtain end-to-end self-learning calculation for the topology control strategy of distribution network fault recovery. Dong et al. [113] constructed an optimization dispatch model based on the RL framework using the Markov decision process. Then the model was trained by the AC algorithm and deep deterministic policy gradient algorithm. Then the system was divided into multiple agents concerning data interaction, and the optimization model was transformed into an MARL model. Compared to the SARL algorithm, the training process can converge more stably. Aiming at transient voltage stability in energy internet (EI), Cao et al. [114] proposed a deep RL algorithm based on CNN to improve decision-making optimization to balance the power supply–demand. This algorithm has a more accurate prediction compared to conventional ML algorithms, and it also satisfies the expectation of system stability. Zhang et al. [115] focused on hybrid energy coordinated control optimization problems for hybrid energy storage systems. The authors designed a deep RL framework embedding a neural network model to solve the formulated decision-making problem. Li et al. [116] focused on the scheduling problem of charging stations to predict the power supply capacity required and conduct a neural network mapping model based on DL. Experiments show the capability of dealing with the sizing problem of station charging capacity. Huang et al. [117] suggested a hybrid approach based on the original differential evolution (DE) algorithm, combining the ant system to tackle the optimal reactive power dispatch (ORPD) problem. Having been tested on a real system, the approach can achieve lower power losses during transmission with better results and performance compared to previous methods. Yuce et al. [118] aimed at optimizing the smart scheduling of energy-consuming devices, utilizing artificial neural network/genetic algorithm (ANN-GA). The best performance was found in a Levenverg–Marquardt-based learning algorithm with which the maximized use of renewable sources and reduced energy demand were achieved. With a great interest in generation expansion planning, Huang et al. [119] presented an investment planning model embedding a double-layer optimization construction and three types of agents built with the Q-learning algorithm

and GA, considering a change in electricity price and generation cost. The model is effective and was practicably proved by two real-world instances.

Now, research on models that involve game theory are concentrated. This ML approach considers the perspective of the economic market and treats each decision maker as an individual agent. The objective is usually the total profit. To improve the efficiency of the energy system, Sheikhi et al. [269] proposed a model based on cloud computing (CC) framework that modifies the classic energy hub (EH) to support managing communication between utility companies and smart energy hubs (S.E. hub). The approach is applied for users to obtain demand side management (DSM) while the Nash equilibrium is achieved by a subgradient optimization algorithm. Fan et al. [270] drew their interest in energy hub and studied the multi-neighbor cooperative economic scheduling problem, where EHs compose a community to exchange energy with each other to minimize operational cost. The authors built a model of a bargaining, cooperative game for this management problem and found the Nash equilibrium to ensure optimal operations. Peng et al. [271] built a bottom-up inter-regional transaction model for electricity pricing mechanism. The results quantitatively showed the relationship between the retailers' behavior and benefit and cost of electricity under the certain economic assumption. Chen et al. [272] proposed a mechanism based on the Stachelberg game for demand response (DR) scheduling optimization problems with load uncertainty. The authors took into account the users' optimal consumption as well as the price of electricity to model the interaction between the service provider and users for the smart grid. The results reached a balanced demand versus supply. For the same interest, Li et al. [273] also reported an optimization framework based on the Stachelberg game to deal with scheduling of the DR energy system, setting profits of the integrated energy operator (IEO) as the objective. The game was built into a MINLP formulation and solved with an introduced sequence operation theory. The results were verified to be applicable by real-world instances.

More widely used ML approaches are shown in this section on the aspect of management, where each energy hub is considered an agent bargaining with each other. Unlike the petroleum supply chain, the electric power system has a shorter supply chain, and the problem is closer to the market.

Table 10. Application in electric power system.

Problem	Reference	Year	Approach
energy-consuming device optimization	[118]	2000	ANN-GA
optimal reactive power dispatch	[117]	2012	DE + ant system
DR scheduling optimization	[272]	2012	Stachelberg-ML mechanism
energy system efficiency	[269]	2015	CC
generation expansion planning	[119]	2016	Q-learning + GA

Table 10. *Cont.*

Problem	Reference	Year	Approach
charging stations scheduling	[116]	2017	NN + DL
multi-neighbor cooperative economic scheduling	[270]	2018	bargaining cooperative game
electricity pricing	[271]	2018	bottom-up inter-regional transaction model
power supply-demand	[114]	2019	RL + CNN
hybrid energy storage	[115]	2019	RL + NN
distribution network fault recovery	[112]	2021	improved Ptr-Net
optimization dispatch	[113]	2021	AC + deep deterministic policy gradient algorithm
DR energy system	[273]	2021	Stachelberg-ML optimization

4.4. Wind Power

The scheduling problem in wind power is another issue that people are concerned about in the energy field that involves the application of game theory to solve combinatorial problems. Table 11 gives a brief summary of applications surveyed.

Marden et al. [120] focused on energy production optimization problems in wind farms and suggested a model-free distributed learning strategy, an algorithm without building a model of interaction between wind turbines based on game theory and cooperative control. This learning strategy was demonstrated to achieve maximum energy production. Quan et al. [121] proposed a non-parametric neural network-based prediction intervals (PIs) with a built-in Monte Carlo simulation method for wind power prediction problems. The stochastic security-constrained unit commitment (SCUC) model was then solved by a heuristic genetic algorithm and the model was presented to be robust. Mei et al. [122] proposed a min-max game model for the static reserve capacity planning problem with large-scale integration of wind power. The authors then introduced a two-stage relaxation algorithm to cope with the min-max game. The application showed robustness and efficiency. Liu et al. [123] analyzed the capacity planning model based on cooperative game theory in a low-carbon economy and presented an improved strategy for the union of wind-farm and grid companies. The model built a balanced and reasonable profit separating mechanism.

Not many references were surveyed for wind power field. The presented results showed an application of ML approaches with game theory, considering wind turbines as independent agents, similar to that in an electric power system.

Table 11. Application in wind power.

Problem	Reference	Year	Approach
energy production optimization	[120]	2013	model-free distributed learning
wind power prediction	[121]	2014	non-parametric NN
static reserve capacity planning	[122]	2014	two-stage relaxation
capacity planning	[123]	2015	cooperative game theory

5. Challenge

5.1. Developing Game Theoretic Learning Methods

There are three main challenges to solving COPs: (1) Accuracy—the policy space of a COP is always huge and hard to explore. For large-scale COPs, the optimal solution is even unknown. Therefore, it is hard to find the best optimization direction, escape from the local optimal, and bound the optimization gap. (2) Efficiency—solving a COP has been proved to be NP-hard, and all algorithms seek to solve it in considerable time. (3) Scalability—despite having the same structure, when the data or scale of a COP is changed, the solution distribution can be totally different.

Game theory has been widely used to solve complex optimization problems, and been integrated into ML, especially RL. Single-player algorithms (e.g., self-play) can improve the policy by beating an opponent, which is a useful way to improve the accuracy, and multi-player algorithms (e.g., centralized training decentralized execution methods) realize cooperation when agents make decisions independently, which may be used to solve complex COPs that include several independent components whose policies are changeable, improving scalability. However, as summarized in Section 3, more than half of the current methods use RL to update the policy, while few use the idea of game theory.

5.2. Challenge of Application in Energy Field

Given three scenarios in the petroleum supply chain to illustrate the application of the learning approach for COPs, it is undeniable that COPs in the petroleum supply chain are significantly complicated, where each scenario requires a different approach.

Refinery production planning involves strategic decisions of each refinery, considering both productions of other refineries and the demand of the market. This scenario is considered a multi-player game, and ML approaches based on game theory are widely used, where each refinery is seen as a single agent (player) competing or cooperating in the game, considering both their own profits and integrated profits in the game to achieve Nash equilibrium. Yet most of the ML approaches using game theory are applied to the aspect of market taking profits as the objective, although cases may differ in the real production process. Hence, more ML methods on the aspect of industrial production are to be investigated. Refinery scheduling is a more micro scenario which focuses on operations inside a single refinery where complex chemical sub-operations are involved. Since no player is involved in this scenario, no learning approach based on game theory is used when solving the problems. Rather, traditional COP frameworks, such as MILP and MINLP with machine learning improvement, such as heuristic algorithm, are more often utilized. Oil transportation including crude oil transportation and refined oil transportation is a classic programming problem that many industrial fields are facing. Problems of transporting through pipeline are mainly surveyed. Again, no learning approaches based on game theory are used due to lack of players in this scenario, and the traditional COP frameworks' embedding approaches, such as the stochastic algorithm, are introduced.

Steel making has a similar production operational process to refinery scheduling, and hence they utilize the same scheduling problems and approaches. Traditional COP frameworks are widely used with the RL-based algorithm introduced for improvement.

Through investigation, two points of view and corresponding methods to solve scheduling problems in electric power system were discovered. ML-based learning approaches aim at solving CO scheduling problems concentrating on operation, which is similar to approaches for refinery scheduling. On the other hand, ML methods using game theory consider each energy hub as an agent deciding with decentralization. Neither approach provides the decision maker with solutions as the blueprint for the whole system, i.e., the methods emphasize the single stage or partial process in the operation instead of having a macro view. Plus, ML methods based on game theory build the model on the aspect of the economic market rather than industrial production, which may generate application problems.

Not many applications for wind power were surveyed. The existing studies build models concerning power generated from and stored in the wind farm. Further investigation in this field is needed, and more surveys of the wind power system are to be conducted.

Many applications of learning methods are utilized to simplify the model from MINLP to MIP, thus reducing the computational cost. However, such simplification may yield problems, such as poor generalization, low accuracy, limited scalability, expensive computation, etc. For instance, an easily computed algorithm achieving a lower gap to the objective may not be able to establish accurate solutions [71], and an adaptive approach for a specific large-scale problem may propose too strong assumptions a priori and thus have poor generalization [10]. With the growing complexity and scale of problems in the energy field, more attention should be paid to uncertainty, which makes the simplification of the models more difficult.

5.3. Application Gap

How to apply new methods in reality is always a big challenge. As investigated in Section 4, plenty of problems in the real world are formulated as ILP or MIP, which are traditionally solved by B&B and other heuristic algorithms. However, from Figure 2, we can see that general or specific ML-based algorithms for ILP lack attention.

We believe there are several reasons for this phenomenon. Firstly, ILP is more difficult to be solved using machine learning methods than structured problems, such as TSP and MIS. In machine learning areas, natural language processing and computer vision have developed rapidly in recent years, whose structures are the sequence and the graph, respectively. Therefore, strong machine learning algorithms are proposed to solve problems with those two structures and can be applied in COPs conveniently. ILP, as a traditional optimization problem with constraints, is more about logic and mathematics than recognition and neural. Secondly, ILP is well solved by some traditional solvers (e.g., Gurobi, and SCIP), and thus is hard to beat. Although traditional solvers are stable, they depend on large amounts of data and experience and are not very useful for new situations in the application. Last but not least, there are few real-world datasets for researchers since large-scale COPs are usually faced by businesses that are private or state owned.

6. Conclusions

This paper investigates ML methods based on game theory to solve COPs in energy fields.

Section 2 is the introduction of the background of combinatorial optimization problems and several popular approaches of ML in addressing this kind of issue over the past years. COPs can be stated as minimization or maximization problems associated with the given objective function(s). A few classical COPs are mentioned, including the traveling salesman problem, minimum spanning tree problem, and knapsack problem, following which several methods are presented: attention mechanisms can handle problems of fixed-length vectors with limited access to input information in neural networks; GNNs which belong to DL methods are designed for graph-expressed data structure; RL can be applied for models involving strategic interactions among rational agents.

Sub-fields of COPs and more details of these ML methods are further discussed in Section 3. The approaches can cover the shortage of traditional methods for COPs: SL improves the computational performance of traditional approaches; reinforcement learning evaluates the quality of a given solution to avoid the difficulty of obtaining labels in SL; and game theory can be applied to problems modeled as RL. The applications on corresponding COPs are demonstrated chronologically.

Section 3 concentrates on some specific applications in energy field and investigates how the ML algorithms are applied to these issues. Although algorithms based on game theory have been applied in energy fields, there still exists space for further exploration. Among three scenarios of the petroleum supply chain, only refinery production planning involves the application of approaches with game theory, while no clue of game is found in the background of refinery scheduling and oil transportation. The review of steel-making also shows no discovery of application on game theory, as the operation is rather similar to that of petroleum production. There are more widely used ML approaches in the electric system and wind power on the aspect of management, where each energy hub or wind turbine is considered as an agent bargaining with each other.

Section 5 is the summary of the challenges over these energy application in which the ML methods are deployed. Consequently, the existing COPs in energy fields can be solved by applying ML algorithm under certain conditions, and there still exists space for learning methods to develop and apply to a wider range of problems.

Future Work

According to the investigation, great efforts have been made by introducing ML approaches to overcome the high computational complexity of traditional approaches for COPs. In the energy field, there exist large-scale COPs that can be studied by ML methods. However, current researchers still face the challenge of balancing poor generalization, low accuracy, limited scalability, expensive computation, etc. To overcome these challenges, some possible ideas for future working direction are as follows.

1. Further investigation of the application of approaches based on game theory to tackle systematic problems is of vital significance. Currently, there are limited applications in petroleum supply chain. Refineries can be modeled as rational independent agents, while it is possible to model oil fields, pipelines, or transportation tasks as agents as well. Building such a system by approaches based on game theory in which each scene in the petroleum supply chain is a game dependent on each other is worth expecting.
2. Studies on a systematic framework to handle the increasing uncertainty in CO scheduling problems in the energy field are worth investigating. With the development of complexity caused by uncertainty in real-world problems, the algorithms also need to consider adapting to the trend.
3. It is worth studying the application of ML approaches to COPs without the framework of a traditional COP solver. Subtle modification may not be able to fully demonstrate the strengths of ML algorithms. By resolving the problem of complexity due to the large scale, ML algorithms may have better performance on COPs.

Author Contributions: Conceptualization, N.M., N.Y. and H.L.; Data curation, X.Y.; Formal analysis, X.Y. and Z.W.; Funding acquisition, H.L. and L.Y.; Investigation, X.Y., Z.W. and H.Z. (Hengxi Zhang); Methodology, N.Y.; Project administration, N.M. and N.Y.; Resources, N.M., H.L. and L.Y.; Software, H.Z. (Haifeng Zhang); Supervision, N.M., N.Y., H.L., H.Z. (Haifeng Zhang) and L.Y.; Validation, X.Y.; Visualization, X.Y. and H.Z. (Hengxi Zhang); Writing—original draft, X.Y., Z.W. and H.Z. (Hengxi Zhang); Writing—review & editing, N.M., N.Y and H.Z. (Haifeng Zhang) All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

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

References

1. Korte, B.H.; Vygen, J.; Korte, B.; Vygen, J. *Combinatorial Optimization*; Springer: Berlin/Heidelberg, Germany, 2011; Volume 1.
2. Schrijver, A. On the history of combinatorial optimization (till 1960). *Handbooks Oper. Res. Manag. Sci.* **2005**, *12*, 1–68.
3. Papadimitriou, C.H.; Steiglitz, K. *Combinatorial Optimization: Algorithms and Complexity*; Courier Corporation: North Chelmsford, MA, USA, 1998.
4. Schrijver, A. *Combinatorial Optimization: Polyhedra and Efficiency*; Springer: Berlin/Heidelberg, Germany, 2003; Volume 24.
5. Panda, D.; Ramteke, M. Preventive crude oil scheduling under demand uncertainty using structure adapted genetic algorithm. *Appl. Energy* **2019**, *235*, 68–82. [CrossRef]
6. Yang, H.; Bernal, D.E.; Franzoi, R.E.; Engineer, F.G.; Kwon, K.; Lee, S.; Grossmann, I.E. Integration of crude-oil scheduling and refinery planning by Lagrangean Decomposition. *Comput. Chem. Eng.* **2020**, *138*, 106812. [CrossRef]
7. Qin, H.; Han, Z. Crude-oil scheduling network in smart field under cyber-physical system. *IEEE Access* **2019**, *7*, 91703–91719. [CrossRef]
8. Panda, D.; Ramteke, M. Dynamic hybrid scheduling of crude oil using structure adapted genetic algorithm for uncertainty of tank unavailability. *Chem. Eng. Res. Des.* **2020**, *159*, 78–91. [CrossRef]
9. Fragkogios, A.; Saharidis, G.K. Modeling and solution approaches for crude oil scheduling in a refinery. In *Energy Management—Collective and Computational Intelligence with Theory and Applicat*; Springer: Berlin/Heidelberg, Germany, 2018; pp. 251–275.
10. Mostafaei, H.; Ghaffari Hadigheh, A. A general modeling framework for the long-term scheduling of multiproduct pipelines with delivery constraints. *Ind. Eng. Chem. Res.* **2014**, *53*, 7029–7042. [CrossRef]
11. Zhang, H.; Liang, Y.; Liao, Q.; Wu, M.; Yan, X. A hybrid computational approach for detailed scheduling of products in a pipeline with multiple pump stations. *Energy* **2017**, *119*, 612–628. [CrossRef]
12. Mostafaei, H.; Castro, P.M.; Ghaffari-Hadigheh, A. A novel monolithic MILP framework for lot-sizing and scheduling of multi-product treelike pipeline networks. *Ind. Eng. Chem. Res.* **2015**, *54*, 9202–9221. [CrossRef]
13. Li, S.; Yang, J.; Song, W.; Chen, A. A real-time electricity scheduling for residential home energy management. *IEEE Internet Things J.* **2018**, *6*, 2602–2611. [CrossRef]
14. Yang, J.; Liu, J.; Fang, Z.; Liu, W. Electricity scheduling strategy for home energy management system with renewable energy and battery storage: A case study. *IET Renew. Power Gener.* **2018**, *12*, 639–648. [CrossRef]
15. Li, S.; Yang, J.; Fang, J.; Liu, Z.; Zhang, H. Electricity scheduling optimisation based on energy cloud for residential microgrids. *IET Renew. Power Gener.* **2019**, *13*, 1105–1114. [CrossRef]
16. Deng, Z.; Liu, C.; Zhu, Z. Inter-hours rolling scheduling of behind-the-meter storage operating systems using electricity price forecasting based on deep convolutional neural network. *Int. Elect. Power Energy Syst.* **2021**, *125*, 106499. [CrossRef]
17. Griffin, P.W.; Hammond, G.P. The prospects for green steel making in a net-zero economy: A UK perspective. *Glob. Transit.* **2021**, *3*, 72–86. [CrossRef]
18. Song, G.W.; Tama, B.A.; Park, J.; Hwang, J.Y.; Bang, J.; Park, S.J.; Lee, S. Temperature Control Optimization in a Steel-Making Continuous Casting Process Using a Multimodal Deep Learning Approach. *Steel Res. Int.* **2019**, *90*, 1900321. [CrossRef]
19. Adetunji, O.; Seidu, S.O. Simulation and Techno-Economic Performance of a Novel Charge Calculation and Melt Optimization Planning Model for Steel Making. *J. Miner. Mater. Charact. Eng.* **2020**, *8*, 277–300. [CrossRef]
20. Ren, L.; Zhou, S.; Peng, T.; Ou, X. A review of CO_2 emissions reduction technologies and low-carbon development in the iron and steel industry focusing on China. *Renew. Sustain. Energy Rev.* **2021**, *143*, 110846. [CrossRef]
21. Osborne, M.J.; Rubinstein, A. *A Course in Game Theory*; MIT Press: Cambridge, MA, USA. 1994.
22. Zhu, P.; Wang, X.; Jia, D.; Guo, Y.; Li, S.; Chu, C. Investigating the co-evolution of node reputation and edge-strategy in prisoner's dilemma game. *Appl. Math. Comput.* **2020**, *386*, 125474. [CrossRef]
23. Shen, C.; Chu, C.; Shi, L.; Perc, M.; Wang, Z. Aspiration-based coevolution of link weight promotes cooperation in the spatial prisoner's dilemma game. *R. Soc. Open Sci.* **2018**, *5*, 180199. [CrossRef]
24. Bian, J.; Lai, K.K.; Hua, Z.; Zhao, X.; Zhou, G. Bertrand vs. Cournot competition in distribution channels with upstream collusion. *Int. J. Prod. Econ.* **2018**, *204*, 278–289. [CrossRef]
25. Lundin, E.; Tangerås, T.P. Cournot competition in wholesale electricity markets: The Nordic power exchange, Nord Pool. *Int. J. Ind. Organ.* **2020**, *68*, 102536. [CrossRef]
26. Dyson, B.J.; Musgrave, C.; Rowe, C.; Sandhur, R. Behavioural and neural interactions between objective and subjective performance in a Matching Pennies game. *Int. J. Psychophysiol.* **2020**, *147*, 128–136. [CrossRef] [PubMed]
27. Dolgova, T.; Bartsev, S. Neural networks playing "matching pennies" with each other: Reproducibility of game dynamics. In *IOP Conference Series: Materials Science and Engineering*; IOP Publishing: Bristol, UK, 2019; Volume 537, p. 42002.
28. Fiez, T.; Chasnov, B.; Ratliff, L. Implicit learning dynamics in stackelberg games: Equilibria characterization, convergence analysis, and empirical study. In Proceedings of the International Conference on Machine Learning, Virtual, 13–18 July 2020; pp. 3133–3144.

29. Jacobsen, H.J.; Jensen, M.; Sloth, B. Evolutionary learning in signalling games. *Games Econ. Behav.* **2001**, *34*, 34–63. [CrossRef]
30. Allen, F.; Morris, S. Game theory models in finance. In *Game Theory and Business Applications*; Springer: Berlin/Heidelberg, Germany, 2014; pp. 17–41.
31. Jordan, M.I.; Mitchell, T.M. Machine learning: Trends, perspectives, and prospects. *Science* **2015**, *349*, 255–260. [CrossRef]
32. Cunningham, P.; Cord, M.; Delany, S.J. Supervised learning. In *Machine Learning Techniques for Multimedia*; Springer: Berlin/Heidelberg, Germany, 2008; pp. 21–49.
33. Barlow, H.B. Unsupervised learning. *Neural Comput.* **1989**, *1*, 295–311. [CrossRef]
34. Ghahramani, Z. Unsupervised learning. In *Summer School on Machine Learning*; Springer: Berlin/Heidelberg, Germany, 2003; pp. 72–112.
35. Hastie, T.; Tibshirani, R.; Friedman, J. Unsupervised learning. In *The Elements of Statistical Learning*; Springer: Berlin/Heidelberg, Germany, 2009; pp. 485–585.
36. Kaelbling, L.P.; Littman, M.L.; Moore, A.W. Reinforcement learning: A survey. *J. Artif. Intell. Res.* **1996**, *4*, 237–285. [CrossRef]
37. Wiering, M.A.; Van Otterlo, M. Reinforcement learning. *Adapt. Learn. Optim.* **2012**, *12*, 729.
38. Li, Y. Deep reinforcement learning: An overview. *arXiv* **2017**, arXiv:1701.07274.
39. Mnih, V.; Kavukcuoglu, K.; Silver, D.; Graves, A.; Antonoglou, I.; Wierstra, D.; Riedmiller, M. Playing atari with deep reinforcement learning. *arXiv* **2013**, arXiv:1312.5602.
40. Sengupta, S.; Kambhampati, S. Multi-agent reinforcement learning in bayesian stackelberg markov games for adaptive moving target defense. *arXiv* **2020**, arXiv:2007.10457.
41. Zheng, L.; Fiez, T.; Alumbaugh, Z.; Chasnov, B.; Ratliff, L.J. Stackelberg actor-critic: Game-theoretic reinforcement learning algorithms. *arXiv* **2021**, arXiv:2109.12286.
42. Shi, D.; Li, L.; Ohtsuki, T.; Pan, M.; Han, Z.; Poor, V. Make Smart Decisions Faster: Deciding D2D Resource Allocation via Stackelberg Game Guided Multi-Agent Deep Reinforcement Learning. *IEEE Trans. Mob. Comput.* **2021**. [CrossRef]
43. Han, C.; Huo, L.; Tong, X.; Wang, H.; Liu, X. Spatial anti-jamming scheme for internet of satellites based on the deep reinforcement learning and Stackelberg game. *IEEE Trans. Veh. Technol.* **2020**, *69*, 5331–5342. [CrossRef]
44. Modares, A.; Somhom, S.; Enkawa, T. A self-organizing neural network approach for multiple traveling salesman and vehicle routing problems. *Int. Trans. Oper. Res.* **1999**, *6*, 591–606. [CrossRef]
45. Fagerholt, K.; Christiansen, M. A travelling salesman problem with allocation, time window and precedence constraints—An application to ship scheduling. *Int. Trans. Oper. Res.* **2000**, *7*, 231–244. [CrossRef]
46. Debnath, D.; Hawary, A. Adapting travelling salesmen problem for real-time UAS path planning using genetic algorithm. In *Intelligent Manufacturing and Mechatronics*; Springer: Berlin/Heidelberg, Germany, 2021; pp. 151–163.
47. Filip, E.; Otakar, M. The travelling salesman problem and its application in logistic practice. *WSEAS Trans. Bus. Econ.* **2011**, *8*, 163–173.
48. Wang, P.; Sanin, C.; Szczerbicki, E. Evolutionary algorithm and decisional DNA for multiple travelling salesman problem. *Neurocomputing* **2015**, *150*, 50–57. [CrossRef]
49. Uchida, A.; Ito, Y.; Nakano, K. Accelerating ant colony optimisation for the travelling salesman problem on the GPU. *Int. J. Parallel Emergent Distrib. Syst.* **2014**, *29*, 401–420. [CrossRef]
50. Zhang, R.; Prokhorchuk, A.; Dauwels, J. Deep reinforcement learning for traveling salesman problem with time windows and rejections. In Proceedings of the 2020 International Joint Conference on Neural Networks (IJCNN), Glasgow, UK, 19–24 July 2020; pp. 1–8.
51. Zhang, Y.; Han, X.; Dong, Y.; Xie, J.; Xie, G.; Xu, X. A novel state transition simulated annealing algorithm for the multiple traveling salesmen problem. *J. Supercomput.* **2021**, *77*, 11827–11852. [CrossRef]
52. Larranaga, P.; Kuijpers, C.M.H.; Murga, R.H.; Inza, I.; Dizdarevic, S. Genetic algorithms for the travelling salesman problem: A review of representations and operators. *Artif. Intell. Rev.* **1999**, *13*, 129–170. [CrossRef]
53. Dorigo, M.; Gambardella, L.M. Ant colonies for the travelling salesman problem. *Biosystems* **1997**, *43*, 73–81. [CrossRef]
54. Gendreau, M.; Laporte, G.; Semet, F. A tabu search heuristic for the undirected selective travelling salesman problem. *Eur. J. Oper. Res.* **1998**, *106*, 539–545. [CrossRef]
55. LeCun, Y.; Bengio, Y.; Hinton, G. Deep learning. *Nature* **2015**, *521*, 436–444. [CrossRef] [PubMed]
56. Sutskever, I.; Vinyals, O.; Le, Q.V. Sequence to sequence learning with neural networks. *Adv. Neural Inform. Process. Syst.* **2014**, *27*, 1–9.
57. Bahdanau, D.; Cho, K.; Bengio, Y. Neural machine translation by jointly learning to align and translate. *arXiv* **2014**, arXiv:1409.0473.
58. Cho, K.; Van Merrienboer, B.; Bahdanau, D.; Bengio, Y. On the properties of neural machine translation: Encoder-decoder approaches. *arXiv* **2014**, arXiv:1409.1259.
59. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Kaiser, Ł.; Polosukhin, I. Attention is all you need. *Adv. Neural Inform. Process. Syst.* **2017**, *30*, 1–15.
60. Zaremba, W.; Sutskever, I.; Vinyals, O. Recurrent neural network regularization. *arXiv* **2014**, arXiv:1409.2329.
61. Yamashita, R.; Nishio, M.; Do, R.K.G.; Togashi, K. Convolutional neural networks: An overview and application in radiology. *Insights Imaging* **2018**, *9*, 611–629. [CrossRef]
62. Deisenroth, M.; Rasmussen, C.E. PILCO: A model-based and data-efficient approach to policy search. In Proceedings of the 28th International Conference on Machine Learning (ICML-11), Bellevue, WA, USA, 28 June–2 July 2011; pp. 465–472.

63. Çalışır, S.; Pehlivanoğlu, M.K. Model-free reinforcement learning algorithms: A survey. In Proceedings of the 2019 27th Signal Processing and Communications Applications Conference (SIU), Sivas, Turkey, 24–26 April 2019; pp. 1–4.
64. Moerland, T.M.; Broekens, J.; Jonker, C.M. Model-based reinforcement learning: A survey. *arXiv* **2020**, arXiv:2006.16712.
65. Kaiser, L.; Babaeizadeh, M.; Milos, P.; Osinski, B.; Campbell, R.H.; Czechowski, K.; Erhan, D.; Finn, C.; Kozakowski, P.; Levine, S.; et al. Model-based reinforcement learning for atari. *arXiv* **2019**, arXiv:1903.00374.
66. Zhao, D.; Wang, H.; Shao, K.; Zhu, Y. Deep reinforcement learning with experience replay based on SARSA. In Proceedings of the 2016 IEEE Symposium Series on Computational Intelligence (SSCI), Athens, Greece, 6–9 December 2016; pp. 1–6.
67. Watkins, C.J.; Dayan, P. Q-learning. *Mach. Learn.* **1992**, *8*, 279–292. [CrossRef]
68. Konda, V.; Tsitsiklis, J. Actor-critic algorithms. In Proceedings of the Advances in Neural Information Processing Systems 12 (NIPS 1999), Denver, CO, USA, 29 November–4 December 1999.
69. Land, A.H.; Doig, A.G. An automatic method for solving discrete programming problems. In *50 Years of Integer Programming 1958–2008*; Springer: Berlin/Heidelberg, Germany, 2010; pp. 105–132.
70. Alvarez, A.M.; Louveaux, Q.; Wehenkel, L. A Supervised Machine Learning Approach to Variable Branching in Branch-and-Bound. In ecml. 2014. Available online: https://orbi.uliege.be/handle/2268/167559 (accessed on 1 May 2022).
71. He, H.; Daume, H., III; Eisner, J.M. Learning to search in branch and bound algorithms. In Proceedings of the Advances in Neural Information Processing Systems 27 (NIPS 2014), Montreal, QC, Canada, 8–13 December 2014.
72. Khalil, E.; Le Bodic, P.; Song, L.; Nemhauser, G.; Dilkina, B. Learning to branch in mixed integer programming. In Proceedings of the AAAI Conference on Artificial Intelligence, Phoenix, AZ, USA, 12–17 February 2016; Volume 30.
73. Baltean-Lugojan, R.; Bonami, P.; Misener, R.; Tramontani, A. *Selecting Cutting Planes for Quadratic Semidefinite Outer-Approximation via Trained Neural Networks*; Technical Report; CPLEX Optimization, IBM: New York, NY, USA, 2018.
74. Hottung, A.; Tanaka, S.; Tierney, K. Deep learning assisted heuristic tree search for the container pre-marshalling problem. *Comput. Oper. Res.* **2020**, *113*, 104781. [CrossRef]
75. Hochreiter, S.; Schmidhuber, J. Long short-term memory. *Neural Comput.* **1997**, *9*, 1735–1780. [CrossRef] [PubMed]
76. Vinyals, O.; Fortunato, M.; Jaitly, N. Pointer networks. In Proceedings of the Advances in Neural Information Processing Systems 28 (NIPS 2015), Montreal, QC, Canada, 7–12 December 2015.
77. Zheng, P.; Zuo, L.; Wang, J.; Zhang, J. Pointer networks for solving the permutation flow shop scheduling problem. In Proceedings of the 48th International Conference on Computers & Industrial Engineering (CIE48), Auckland, New Zealand, 2–5 December 2018; pp. 2–5.
78. Scarselli, F.; Gori, M.; Tsoi, A.C.; Hagenbuchner, M.; Monfardini, G. The graph neural network model. *IEEE Trans. Neural Netw.* **2008**, *20*, 61–80. [CrossRef] [PubMed]
79. Battaglia, P.W.; Hamrick, J.B.; Bapst, V.; Sanchez-Gonzalez, A.; Zambaldi, V.; Malinowski, M.; Tacchetti, A.; Raposo, D.; Santoro, A.; Faulkner, R.; et al. Relational inductive biases, deep learning, and graph networks. *arXiv* **2018**, arXiv:1806.01261.
80. Gori, M.; Monfardini, G.; Scarselli, F. A new model for learning in graph domains. In Proceedings of the 2005 IEEE International Joint Conference on Neural Networks, Montreal, QC, Canada, 31 July–4 August 2005; Volume 2, pp. 729–734.
81. Nowak, A.; Villar, S.; Bandeira, A.S.; Bruna, J. A note on learning algorithms for quadratic assignment with graph neural networks. In Proceedings of the 34th International Conference on Machine Learning (ICML), Sydney, Australia, 6–11 August 2017; Volume 1050, p. 22.
82. Bresson, X.; Laurent, T. Residual gated graph convnets. *arXiv* **2017**, arXiv:1711.07553.
83. Vlastelica, M.; Paulus, A.; Musil, V.; Martius, G.; Rolínek, M. Differentiation of blackbox combinatorial solvers. *arXiv* **2019**, arXiv:1912.02175.
84. Paulus, A.; Rolínek, M.; Musil, V.; Amos, B.; Martius, G. Comboptnet: Fit the right np-hard problem by learning integer programming constraints. In Proceedings of the International Conference on Machine Learning, Virtual, 18–24 July 2021; pp. 8443–8453.
85. Bello, I.; Pham, H.; Le, Q.V.; Norouzi, M.; Bengio, S. Neural combinatorial optimization with reinforcement learning. *arXiv* **2016**, arXiv:1611.09940.
86. Deudon, M.; Cournut, P.; Lacoste, A.; Adulyasak, Y.; Rousseau, L.M. Learning heuristics for the tsp by policy gradient. In Proceedings of the International Conference on the Integration of Constraint Programming, Artificial Intelligence, and Operations Research, Vienna, Austria, 21–24 September 2018; Springer: Berlin/Heidelberg, Germany, 2018; pp. 170–181.
87. Emami, P.; Ranka, S. Learning permutations with sinkhorn policy gradient. *arXiv* **2018**, arXiv:1805.07010.
88. Kool, W.; Van Hoof, H.; Welling, M. Attention, learn to solve routing problems! *arXiv* **2018**, arXiv:1803.08475.
89. Laterre, A.; Fu, Y.; Jabri, M.K.; Cohen, A.S.; Kas, D.; Hajjar, K.; Dahl, T.S.; Kerkeni, A.; Beguir, K. Ranked reward: Enabling self-play reinforcement learning for combinatorial optimization. *arXiv* **2018**, arXiv:1807.01672.
90. Drori, I.; Kharkar, A.; Sickinger, W.R.; Kates, B.; Ma, Q.; Ge, S.; Dolev, E.; Dietrich, B.; Williamson, D.P.; Udell, M. Learning to solve combinatorial optimization problems on real-world graphs in linear time. In Proceedings of the 2020 19th IEEE International Conference on Machine Learning and Applications (ICMLA), Virtual, 14–17 December 2020; pp. 19–24.
91. Shahadat, A.S.B.; Ayon, S.I.; Khatun, M.R. SSGTA: A novel swap sequence based Ggame theory algorithm for traveling salesman problem. In Proceedings of the 2021 24th International Conference on Computer and Information Technology (ICCIT), Dalian, China, 5–7 May 2021; pp. 1–5.

92. Kulkarni, A.J.; Tai, K. Probability collectives: A multi-agent approach for solving combinatorial optimization problems. *Appl. Soft Comput.* **2010**, *10*, 759–771. [CrossRef]
93. Gasse, M.; Chételat, D.; Ferroni, N.; Charlin, L.; Lodi, A. Exact combinatorial optimization with graph convolutional neural networks. In Proceedings of the Advances in Neural Information Processing Systems 32, Vancouver, BC, Canada, 8–14 December 2019.
94. Tang, Y.; Agrawal, S.; Faenza, Y. Reinforcement learning for integer programming: Learning to cut. In Proceedings of the International Conference on Machine Learning, Virtual, 13–18 July 2020; pp. 9367–9376.
95. Khalil, E.; Dai, H.; Zhang, Y.; Dilkina, B.; Song, L. Learning combinatorial optimization algorithms over graphs. In Proceedings of the Advances in Neural Information Processing Systems 30, Long Beach, CA, USA, 4–9 December 2017.
96. Karalias, N.; Loukas, A. Erdos goes neural: An unsupervised learning framework for combinatorial optimization on graphs. *Adv. Neural Inform. Process. Syst.* **2020**, *33*, 6659–6672.
97. Chen, X.; Tian, Y. Learning to perform local rewriting for combinatorial optimization. In Proceedings of the Advances in Neural Information Processing Systems 32, Vancouver, BC, Canada, 8–14 December 2019.
98. Tominac, P.; Mahalec, V. A game theoretic framework for petroleum refinery strategic production planning. *Aiche J.* **2017**, *63*, 2751–2763. [CrossRef]
99. Ravinger, F. Analyzing Oil Refinery Investment Decisions: A Game Theoretic Approach. Ph.D. Thesis, Central European University, Vienna, Austria, 2007.
100. Tominac, P.A. Game Theoretic Approaches to Petroleum Refinery Production Planning—A Justification for the Enterprise Level Optimization of Production Planning. Ph.D. Thesis, McMaster University, Hamilton, ON, Canada, 2017.
101. Babaei, M.; Asgarian, F.; Jamali, M.B.; Rasti-Barzoki, M.; Piran, M.J. A game theoretic approach for pricing petroleum and determining investors' production volume with the consideration of government and intermediate producers. *Sustain. Energy Technol. Assess.* **2020**, *42*, 100825. [CrossRef]
102. Shah, N.K.; Sahay, N.; Ierapetritou, M.G. Efficient decomposition approach for large-scale refinery scheduling. *Ind. Eng. Chem. Res.* **2015**, *54*, 9964–9991. [CrossRef]
103. Liang, Y.; Liao, Q.; Zhang, H.; Nie, S.; Yan, X.; Ma, J. Research progress on production scheduling optimization of refi nery. *Oil Gas Storage Transp.* **2017**, *36*, 646–650.
104. Wang, J.; Rong, G. Robust Optimization Model for Crude Oil Scheduling under Uncertainty. *Ind. Eng. Chem. Res.* **2010**, *49*, 1737–1748. [CrossRef]
105. Li, S.; Su, D. Establishment and solution of refinery multi-stage production scheduling model based on dynamic programming. *Control Inst. Chem. Ind.* **2009**, *36*, 6.
106. Cheng, L.; Duran, M.A. Logistics for world-wide crude oil transportation using discrete event simulation and optimal control. *Comput. Chem. Eng.* **2004**, *28*, 897–911. [CrossRef]
107. Wang, L.; Kinable, J.; Woensel, T.V. The fuel replenishment problem: A split-delivery multi-compartment vehicle routing problem with multiple trips. *Comput. Oper. Res.* **2020**, *118*, 104904. [CrossRef]
108. Lin, S.; Xu, A.; Liu, C.; Kai, F.; Li, J. Crane scheduling method in steelmaking workshop based on deep reinforcement learning. *China Metall.* **2021**, *31*, 7.
109. Zhou, Y. Application Research of Improved Gray Wolf Optimization Algorithm in Optimal Scheduling of Ateelmaking and Continuous Casting. Ph.D. Thesis, Qingdao University of Science and Technology, Qingdao, China, 2021.
110. Jia, C. Deep Reinforcement Learning for Batch Machine Scheduling Problem with Non-Identical Job Sizes. Ph.D. Thesis, Hefei University of Technology, Heifei, China, 2021.
111. Ma, Y.; Liu, X.; Jiang, S. Machine Learning-Based Scheduling Approach for Steelmaking-Continuous Casting Production. *Metall. Ind. Autom.* **2022**, *46*, 2.
112. Yan, D.; Peng, G.; Gao, H.; Chen, S.; Zhou, Y. Research on distribution network topology control based on deep reinforcement learning combinatorial optimization. *Power Syst. Technol.* **2021**, 1–9. [CrossRef]
113. Dong, L.; Liu, Y.; Qiao, J.; Wang, X.; Wang, C.; Pu, T. Optimal dispatch of combined heat and power system based on multi-agent deep reinforcement learning. *Power Syst. Technol.* **2021**, *45*, 9.
114. Cao, J.; Zhang, W.; Xiao, Z.; Hua, H. Reactive Power Optimization for Transient Voltage Stability in Energy Internet via Deep Reinforcement Learning Approach. *Energies* **2019**, *12*, 1556. [CrossRef]
115. Zhang, Z.; Qiu, C.; Zhang, D.; Xu, S.; He, X. A Coordinated Control Method for Hybrid Energy Storage System in Microgrid Based on Deep Reinforcement Learning. *Power Syst. Technol.* **2019**, *6*, 1914–1921. [CrossRef]
116. Li, Z.; Hou, X.; Liu, Y.; Sun, H.; Zhu, Z.; Long, Y.; Xu, T. A capacity planning method of charging station based on depth learning. *Power Syst. Prot. Control* **2017**, *45*, 67–73.
117. Huang, C.M.; Huang, Y.C. Combined Differential Evolution Algorithm and Ant System for Optimal Reactive Power Dispatch. *Energy Procedia* **2012**, *14*, 1238–1243. [CrossRef]
118. Yuce, B.; Rezgui, Y.; Mourshed, M. ANN-GA smart appliance scheduling for optimised energy management in the domestic sector. *Energy Build.* **2016**, *111*, 311–325. [CrossRef]
119. Huang, X.; Guo, R. A multi-agent model of generation expansion planning in electricity market. *Power Syst. Prot. Control* **2016**. Available online: http://en.cnki.com.cn/Article_en/CJFDTOTAL-JDQW201624001.htm (accessed on 1 May 2022).

120. Marden, J.R.; Ruben, S.D.; Pao, L.Y. A model-free approach to wind farm control using game theoretic methods. *IEEE Trans. Control Syst. Technol.* **2013**, *21*, 1207–1214. [CrossRef]
121. Quan, H.; Srinivasan, D.; Khosravi, A. Incorporating wind power forecast uncertainties into stochastic unit commitment using neural network-based prediction intervals. *IEEE Trans. Neural Netw. Learn. Syst.* **2014**, *26*, 2123–2135. [CrossRef] [PubMed]
122. Mei, S.; Zhang, D.; Wang, Y.; Liu, F.; Wei, W. Robust optimization of static reserve planning with large-scale integration of wind power: A game theoretic approach. *IEEE Trans. Sustain. Energy* **2014**, *5*, 535–545. [CrossRef]
123. Liu, W.; Ling, Y.; Zhao, T. Cooperative game based capacity planning model for wind power in low-carbon economy. *Autom. Electr. Power Syst.* **2015**, *39*, 68–74.
124. Kuçukoglu, I.; Dewil, R.; Cattrysse, D. Hybrid simulated annealing and tabu search method for the electric travelling salesman problem with time windows and mixed charging rates. *Expert Syst. Appl.* **2019**, *134*, 279–303. [CrossRef]
125. Zhou, A.H.; Zhu, L.P.; Hu, B.; Deng, S.; Song, Y.; Qiu, H.; Pan, S. Traveling-salesman-problem algorithm based on simulated annealing and gene-expression programming. *Information* **2018**, *10*, 7. [CrossRef]
126. Dewantoro R.W.; Sihombing, P. The combination of ant colony optimization (ACO) and tabu search (TS) algorithm to solve the traveling salesman problem (TSP). In Proceedings of the 2019 3rd International Conference on Electrical, Telecommunication and Computer Engineering (ELTICOM), Medan, Indonesia, 16–17 September 2019; pp. 160–164.
127. Arapoglu, O.; Akram, V.K.; Dagdeviren, O. An energy-efficient, self-stabilizing and distributed algorithm for maximal independent set construction in wireless sensor networks. *Comput. Stand. Interfaces* **2019**, *62*, 32–42. [CrossRef]
128. Kose, A.; Ozbek, B. Resource allocation for underlaying device-to-device communications using maximal independent sets and knapsack algorithm. In Proceedings of the 2018 IEEE 29th Annual International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC), Bologna, Italy, 9–12 September 2018; pp. 1–5.
129. López-Ramírez, C.; Gómez, J.E.G.; Luna, G.D.I. Building a Maximal Independent Set for the Vertex-coloring Problem on Planar Graphs. *Electron. Notes Theor. Comput. Sci.* **2020**, *354*, 75–89. [CrossRef]
130. Tarjan, R.E.; Trojanowski, A.E. Finding a maximum independent set. *SIAM J. Comput.* **1977**, *6*, 537–546. [CrossRef]
131. Feo, T.A.; Resende, M.G.; Smith, S.H. A greedy randomized adaptive search procedure for maximum independent set. *Oper. Res.* **1994**, *42*, 860–878. [CrossRef]
132. Berman, P.; Furer, M. Approximating maximum independent set. In Proceedings of the Fifth Annual ACM-SIAM Symposium on Discrete Algorithms, Arlington, VA, USA, 23–25 January 1994; Volume 70, p. 365.
133. Reba, K.; Guid, M.; Rozman, K.; Janežič, D.; Konc, J. Exact Maximum Clique Algorithm for Different Graph Types Using Machine Learning. *Mathematics* **2021**, *10*, 97. [CrossRef]
134. Ghaffari, M. Distributed maximal independent set using small messages. In Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms, San Diego, CA, USA, 6–9 January 2019; pp. 805–820.
135. Andrade, D.V.; Resende, M.G.; Werneck, R.F. Fast local search for the maximum independent set problem. *J. Heuristics* **2012**, *18*, 525–547. [CrossRef]
136. Xiao, M.; Nagamochi, H. Exact algorithms for maximum independent set. *Inform. Comput.* **2017**, *255*, 126–146. [CrossRef]
137. Zetina, C.A.; Contreras, I.; Fernandez, E.; Luna-Mota, C. Solving the optimum communication spanning tree problem. *Eur. Oper. Res.* **2019**, *273*, 108–117. [CrossRef]
138. Akpan, N.; Iwok, I. A minimum spanning tree approach of solving a transportation problem. *Int. J. Math. Stat. Invent.* **2017**, *5*, 9–18.
139. Bartolín, H.; Martínez, F.; Cortés, J.A. Topological GIS-based analysis of a water distribution network model. Applications of the minimum spanning tree. In Proceedings of the Computing and Control for the Water Industry, Exeter, UK, 5–7 September 2005.
140. Liao, X.Q.; Su, T.; Ma, L. Application of neutrosophic minimum spanning tree in electrical power distribution network. *CAAI Trans. Intell. Technol.* **2020**, *5*, 99–105. [CrossRef]
141. Dong, M.; Zhang, X.; Yang, K.; Liu, R.; Chen, P. Forecasting the COVID-19 transmission in Italy based on the minimum spanning tree of dynamic region network. *PeerJ* **2021**, *9*, e11603. [CrossRef]
142. Kireyeu, V. Cluster dynamics studied with the phase-space minimum spanning tree approach. *Phys. Rev. C* **2021**, *103*, 54905. [CrossRef]
143. Mashreghi, A.; King, V. Broadcast and minimum spanning tree with o (m) messages in the asynchronous CONGEST model. *Distrib. Comput.* **2021**, *34*, 283–299. [CrossRef]
144. Jin, Y.; Zhao, H.; Gu, F.; Bu, P.; Na, M. A spatial minimum spanning tree filter. *Meas. Sci. Technol.* **2020**, *32*, 15204. [CrossRef]
145. Ochs, K.; Michaelis, D.; Solan, E. Wave digital emulation of a memristive circuit to find the minimum spanning tree. In Proceedings of the 2019 IEEE 62nd International Midwest Symposium on Circuits and Systems (MWSCAS), Dallas, TX, USA, 4–7 August 2019; pp. 351–354.
146. Farashi, S.; Khosrowabadi, R. EEG based emotion recognition using minimum spanning tree. *Phys. Eng. Sci. Med.* **2020**, *43*, 985–996. [CrossRef] [PubMed]
147. Dong, W.; Wang, J.; Wang, C.; Qi, Z.; Ding, Z. Graphical Minimax Game and Off-Policy Reinforcement Learning for Heterogeneous MASs with Spanning Tree Condition. *Guid. Navig. Control* **2021**, *1*, 2150011. [CrossRef]
148. Dey, A.; Son, L.H.; Pal, A.; Long, H.V. Fuzzy minimum spanning tree with interval type 2 fuzzy arc length: Formulation and a new genetic algorithm. *Soft Comput.* **2020**, *24*, 3963–3974. [CrossRef]

149. Wang, Y.; Yu, S.; Gu, Y.; Shun, J. Fast parallel algorithms for euclidean minimum spanning tree and hierarchical spatial clustering. In Proceedings of the 2021 International Conference on Management of Data, Virtual, 20–25 June 2021; pp. 1982–1995.
150. Gyoten, H.; Hiromoto, M.; Sato, T. Area efficient annealing processor for ising model without random number generator. *IEICE Trans. Inform. Syst.* **2018**, *101*, 314–323. [CrossRef]
151. Cook, C.; Zhao, H.; Sato, T.; Hiromoto, M.; Tan, S.X.D. GPU-based ising computing for solving max-cut combinatorial optimization problems. *Integration* **2019**, *69*, 335–344. [CrossRef]
152. Patel, S.; Chen, L.; Canoza, P.; Salahuddin, S. Ising model optimization problems on a FPGA accelerated restricted Boltzmann machine. *arXiv* **2020**, arXiv:2008.04436.
153. Tu, K.N. Recent advances on electromigration in very-large-scale-integration of interconnects. *J. Appl. Phys.* **2003**, *94*, 5451–5473. [CrossRef]
154. Wang, J.; Paesani, S.; Ding, Y.; Santagati, R.; Skrzypczyk, P.; Salavrakos, A.; Tura, J.; Augusiak, R.; Mančinska, L.; Bacco, D.; et al. Multidimensional quantum entanglement with large-scale integrated optics. *Science* **2018**, *360*, 285–291. [CrossRef]
155. Noé, F.; De Fabritiis, G.; Clementi, C. Machine learning for protein folding and dynamics. *Curr. Opin. Struct. Biol.* **2020**, *60*, 77–84. [CrossRef]
156. Rodriguez, A.; Wright, G.; Emrich, S.; Clark, P.L. %MinMax: A versatile tool for calculating and comparing synonymous codon usage and its impact on protein folding. *Protein Sci.* **2018**, *27*, 356–362. [CrossRef]
157. Poljak, S.; Rendl, F. Solving the max-cut problem using eigenvalues. *Discrete Appl. Math.* **1995**, *62*, 249–278. [CrossRef]
158. Festa, P.; Pardalos, P.M.; Resende, M.G.; Ribeiro, C.C. Randomized heuristics for the MAX-CUT problem. *Optim. Methods Softw.* **2002**, *17*, 1033–1058. [CrossRef]
159. Kim, Y.H.; Yoon, Y.; Geem, Z.W. A comparison study of harmony search and genetic algorithm for the max-cut problem. *Swarm Evol. Comput.* **2019**, *44*, 130–135. [CrossRef]
160. Martí, R.; Duarte, A.; Laguna, M. Advanced scatter search for the max-cut problem. *INFORMS J. Comput.* **2009**, *21*, 26–38. [CrossRef]
161. Blum, C.; Schmid, V. Solving the 2D bin packing problem by means of a hybrid evolutionary algorithm. *Procedia Comput. Sci.* **2013**, *18*, 899–908. [CrossRef]
162. Hifi, M.; Kacem, I.; Nègre, S.; Wu, L. A linear programming approach for the three-dimensional bin-packing problem. *Electron. Notes Discret. Math.* **2010**, *36*, 993–1000. [CrossRef]
163. Yu, A.B.; Zou, R.; Standish, N. Modifying the linear packing model for predicting the porosity of nonspherical particle mixtures. *Ind. Eng. Chem. Res.* **1996**, *35*, 3730–3741. [CrossRef]
164. Biro, P.; Manlove, D.F.; Rizzi, R. Maximum weight cycle packing in directed graphs, with application to kidney exchange programs. *Discret. Math. Algorithms Appl.* **2009**, *1*, 499–517. [CrossRef]
165. Epstein, L.; Levin, A. Bin packing with general cost structures. *Math. Program.* **2012**, *132*, 355–391. [CrossRef]
166. Lai, P.; He, Q.; Abdelrazek, M.; Chen, F.; Hosking, J.; Grundy, J.; Yang, Y. Optimal edge user allocation in edge computing with variable sized vector bin packing. In Proceedings of the International Conference on Service-Oriented Computing, Hangzhou, China, 12–15 November 2018; Springer: Berlin/Heidelberg, Germany, 2018; pp. 230–245.
167. Mohiuddin, I.; Almogren, A.; Al Qurishi, M.; Hassan, M.M.; Al Rassan, I.; Fortino, G. Secure distributed adaptive bin packing algorithm for cloud storage. *Future Gener. Comput. Syst.* **2019**, *90*, 307–316. [CrossRef]
168. Baldi, M.M.; Manerba, D.; Perboli, G.; Tadei, R. A generalized bin packing problem for parcel delivery in last-mile logistics. *Eur. J. Oper. Res.* **2019**, *274*, 990–999. [CrossRef]
169. Witteman, M.; Deng, Q.; Santos, B.F. A bin packing approach to solve the aircraft maintenance task allocation problem. *Eur. J. Oper. Res.* **2021**, *294*, 365–376. [CrossRef]
170. Qomi, T.; Hamedi, M.; Tavakkoli-Moghaddam, R. Optimization of Crude Oil Transportation using a Variable Cost and Size Bin Packing Problem (VCSBPP). 2021. Available online: https://www.trijournal.ir/article_121959.html?lang=en (accessed on 1 May 2022).
171. Frenk, J.G.; Galambos, G. Hybrid next-fit algorithm for the two-dimensional rectangle bin-packing problem. *Computing* **1987**, *39*, 201–217. [CrossRef]
172. Mao, W. Tight worst-case performance bounds for next-k-fit bin packing. *SIAM J. Comput.* **1993**, *22*, 46–56. [CrossRef]
173. Xia, B.; Tan, Z. Tighter bounds of the First Fit algorithm for the bin-packing problem. *Discrete Appl. Math.* **2010**, *158*, 1668–1675. [CrossRef]
174. El-Ashmawi, W.H.; Abd Elminaam, D.S. A modified squirrel search algorithm based on improved best fit heuristic and operator strategy for bin packing problem. *Appl. Soft Comput.* **2019**, *82*, 105565. [CrossRef]
175. Dhahbi, S.; Berrima, M.; Al-Yarimi, F.A. Load balancing in cloud computing using worst-fit bin-stretching. *Clust. Comput.* **2021**, *24*, 2867–2881. [CrossRef]
176. Bansal, N.; Han, X.; Iwama, K.; Sviridenko, M.; Zhang, G. A harmonic algorithm for the 3D strip packing problem. *SIAM J. Comput.* **2013**, *42*, 579–592. [CrossRef]
177. Gu, X.; Chen, G.; Xu, Y. Deep performance analysis of refined harmonic bin packing algorithm. *J. Comput. Sci. Technol.* **2002**, *17*, 213–218. [CrossRef]

178. Raj, P.H.; Kumar, P.R.; Jelciana, P.; Rajagopalan, S. Modified first fit decreasing method for load balancing in mobile clouds. In Proceedings of the 2020 4th International Conference on Intelligent Computing and Control Systems (ICICCS), Madurai, India, 13–15 May 2020; pp. 1107–1110.
179. Jin, S.; Liu, M.; Wu, Y.; Xu, Y.; Zhang, J.; Xu, Y. Research of message scheduling for in-vehicle FlexRay network static segment based on next fit decreasing (NFD) algorithm. *Appl. Sci.* **2018**, *8*, 2071. [CrossRef]
180. Coffman, E.G.; Garey, M.R.; Johnson, D.S. Approximation algorithms for bin-packing—An updated survey. In *Algorithm Design for Computer System Design*; Springer: Berlin/Heidelberg, Germany, 1984; pp. 49–106.
181. Labbé, M.; Laporte, G.; Martello, S. An exact algorithm for the dual bin packing problem. *Oper. Res. Lett.* **1995**, *17*, 9–18. [CrossRef]
182. Cheuk, K.W.; Luo, Y.J.; Benetos, E.; Herremans, D. Revisiting the onsets and frames model with additive attention. In Proceedings of the 2021 International Joint Conference on Neural Networks (IJCNN), Shenzhen, China, 18–22 July 2021; pp. 1–8.
183. Wu, C.; Wu, F.; Qi, T.; Huang, Y.; Xie, X. Fastformer: Additive attention can be all you need. *arXiv* **2021**, arXiv:2108.09084.
184. Li, X.; Xu, Q.; Chen, X.; Li, C. Additive Attention for CNN-based Classification. In Proceedings of the 2021 IEEE International Conference on Mechatronics and Automation (ICMA), Takamatsu, Japan, 8–11 August 2021; pp. 55–59.
185. Tian, Y.; Newsam, S.; Boakye, K. Image Search with Text Feedback by Additive Attention Compositional Learning. *arXiv* **2022**, arXiv:2203.03809.
186. Graves, A.; Wayne, G.; Danihelka, I. Neural turing machines. *arXiv* **2014**, arXiv:1410.5401.
187. Ma, X.; Hu, Z.; Liu, J.; Peng, N.; Neubig, G.; Hovy, E. Stack-pointer networks for dependency parsing. *arXiv* **2018**, arXiv:1805.01087.
188. Li, J.; Wang, Y.; Lyu, M.R.; King, I. Code completion with neural attention and pointer networks. *arXiv* **2017**, arXiv:1711.09573.
189. Yavuz, S.; Rastogi, A.; Chao, G.L.; Hakkani-Tur, D. Deepcopy: Grounded response generation with hierarchical pointer networks. *arXiv* **2019**, arXiv:1908.10731.
190. Luong, M.T.; Pham, H.; Manning, C.D. Effective approaches to attention-based neural machine translation. *arXiv* **2015**, arXiv:1508.04025.
191. Trueman, T.E.; Cambria, E. A convolutional stacked bidirectional lstm with a multiplicative attention mechanism for aspect category and sentiment detection. *Cogn. Comput.* **2021**, *13*, 1423–1432.
192. Cui, R.; Wang, J.; Wang, Z. Multiplicative attention mechanism for multi-horizon time series forecasting. In Proceedings of the 2021 International Joint Conference on Neural Networks (IJCNN), Shenzhen, China, 18–22 July 2021; pp. 1–6.
193. Gan, J.; Liu, H.; He, T. Prediction of air pollutant concentration based on luong attention mechanism Seq2Seq model. In Proceedings of the 2021 7th Annual International Conference on Network and Information Systems for Computers (ICNISC), Guiyang, China, 23–25 July 2021; pp. 321–325.
194. Zhang, S.; Tong, H.; Xu, J.; Maciejewski, R. Graph convolutional networks: A comprehensive review. *Comput. Soc. Netw.* **2019**, *6*, 1–23. [CrossRef]
195. Veličković, P.; Cucurull, G.; Casanova, A.; Romero, A.; Lio, P.; Bengio, Y. Graph attention networks. *arXiv* **2017**, arXiv:1710.10903.
196. Bai, L.; Yao, L.; Li, C.; Wang, X.; Wang, C. Adaptive graph convolutional recurrent network for traffic forecasting. *Adv. Neural Inform. Process. Syst.* **2020**, *33*, 17804–17815.
197. Wu, L.; Chen, Y.; Shen, K.; Guo, X.; Gao, H.; Li, S.; Pei, J.; Long, B. Graph neural networks for natural language processing: A survey. *arXiv* **2021**, arXiv:2106.06090.
198. Fan, W.; Ma, Y.; Li, Q.; He, Y.; Zhao, E.; Tang, J.; Yin, D. Graph neural networks for social recommendation. In Proceedings of the World Wide Web Conference, San Francisco, CA, USA, 13–17 May 2019; pp. 417–426.
199. Ioannidis, V.N.; Marques, A.G.; Giannakis, G.B. Graph neural networks for predicting protein functions. In Proceedings of the 2019 IEEE 8th International Workshop on Computational Advances in Multi-Sensor Adaptive Processing (CAMSAP), Guadeloupe, France, 15–18 December 2019; pp. 221–225.
200. Tsitsulin, A.; Palowitch, J.; Perozzi, B.; Muller, E. Graph clustering with graph neural networks. *arXiv* **2020**, arXiv:2006.16904.
201. Abu-El-Haija, S.; Kapoor, A.; Perozzi, B.; Lee, J. N-gcn: Multi-scale graph convolution for semi-supervised node classification. In Uncertainty in Artificial Intelligence. 2020. pp. 841–851. Available online: http://proceedings.mlr.press/v115/abu-el-haija20a.html (accessed on 1 May 2022).
202. Adamic, L.A.; Adar, E. Friends and neighbors on the web. *Soc. Netw.* **2003**, *25*, 211–230. [CrossRef]
203. Koren, Y.; Bell, R.; Volinsky, C. Matrix factorization techniques for recommender systems. *Computer* **2009**, *42*, 30–37. [CrossRef]
204. Schafer, J.B.; Konstan, J.; Riedl, J. Recommender systems in e-commerce. In Proceedings of the 1st ACM Conference on Electronic Commerce, Denver, CO, USA, 3–5 November 1999; pp. 158–166.
205. Schulman, J.; Levine, S.; Abbeel, P.; Jordan, M.; Moritz, P. Trust region policy optimization. In Proceedings of the International Conference on Machine Learning, Lille, France, 6–11 July 2015; pp. 1889–1897.
206. Cortes, J.; Martinez, S.; Karatas, T.; Bullo, F. Coverage control for mobile sensing networks. *IEEE Trans. Robot. Autom.* **2004**, *20*, 243–255. [CrossRef]
207. Choi, J.; Oh, S.; Horowitz, R. Distributed learning and cooperative control for multi-agent systems. *Automatica* **2009**, *45*, 2802–2814. [CrossRef]
208. Adler, J.L.; Blue, V.J. A cooperative multi-agent transportation management and route guidance system. *Transp. Res. Part Emerg. Technol.* **2002**, *10*, 433–454. [CrossRef]

209. Wang, S.; Wan, J.; Zhang, D.; Li, D.; Zhang, C. Towards smart factory for industry 4.0: A self-organized multi-agent system with big data based feedback and coordination. *Comput. Netw.* **2016**, *101*, 158–168. [CrossRef]
210. Lee, J.W.; Zhang, B.T. Stock trading system using reinforcement learning with cooperative agents. In Proceedings of the Nineteenth International Conference on Machine Learning, San Francisco, CA, USA, 8–12 July 2002; pp. 451–458.
211. Lee, J.W.; Park, J.; Jangmin, O.; Lee, J.; Hong, E. A multiagent approach to *Q*-learning for daily stock trading. *IEEE Trans. Syst. Man Cybern. Part A Syst. Hum.* **2007**, *37*, 864–877. [CrossRef]
212. Castelfranchi, C. The theory of social functions: Challenges for computational social science and multi-agent learning. *Cogn. Syst. Res.* **2001**, *2*, 5–38. [CrossRef]
213. Zhang, K.; Yang, Z.; Başar, T. Multi-agent reinforcement learning: A selective overview of theories and algorithms. In *Handbook of Reinforcement Learning and Control*; Springer: Berlin/Heidelberg, Germany, 2021; pp. 321–384.
214. Rădulescu, R.; Legrand, M.; Efthymiadis, K.; Roijers, D.M.; Nowé, A. Deep multi-agent reinforcement learning in a homogeneous open population. In *Proceedings of the Benelux Conference on Artificial Intelligence, 2018*; Springer: Berlin/Heidelberg, Germany, 2018; pp. 90–105.
215. Yu, Y.; Wang, T.; Liew, S.C. Deep-reinforcement learning multiple access for heterogeneous wireless networks. *IEEE J. Sel. Areas Commun.* **2019**, *37*, 1277–1290. [CrossRef]
216. Gupta, J.K.; Egorov, M.; Kochenderfer, M. Cooperative multi-agent control using deep reinforcement learning. In *Proceedings of the International Conference on Autonomous Agents and Multiagent Systems, 2017*; Springer: Berlin/Heidelberg, Germany, 2017; pp. 66–83.
217. OroojlooyJadid, A.; Hajinezhad, D. A review of cooperative multi-agent deep reinforcement learning. *arXiv* **2019**, arXiv:1908.03963.
218. Liu, Y.; Nejat, G. Multirobot cooperative learning for semiautonomous control in urban search and rescue applications. *J. Field Robot.* **2016**, *33*, 512–536. [CrossRef]
219. Liu, I.J.; Jain, U.; Yeh, R.A.; Schwing, A. Cooperative exploration for multi-agent deep reinforcement learning. In Proceedings of the International Conference on Machine Learning, Online, 18–24 July 2021; pp. 6826–6836.
220. Khan, M.I.; Alam, M.M.; Moullec, Y.L.; Yaacoub, E. Throughput-aware cooperative reinforcement learning for adaptive resource allocation in device-to-device communication. *Future Internet*. **2017**, *9*, 72. [CrossRef]
221. Abramson, M.; Wechsler, H. Competitive reinforcement learning for combinatorial problems. In Proceedings of the International Joint Conference on Neural Networks. Proceedings (Cat. No. 01CH37222), Washington, DC, USA, 15–19 July 2001; Volume 4, pp. 2333–2338.
222. Crawford, V.P. Learning the optimal strategy in a zero-sum game. *Econom. J. Econom. Soc.* **1974**, *42*, 885–891. [CrossRef]
223. McKenzie, M.; Loxley, P.; Billingsley, W.; Wong, S. Competitive reinforcement learning in atari games. In *Proceedings of the Australasian Joint Conference on Artificial Intelligence, 2017*; Springer: Berlin/Heidelberg, Germany, 2017; pp. 14–26.
224. Kutschinski, E.; Uthmann, T.; Polani, D. Learning competitive pricing strategies by multi-agent reinforcement learning. *J. Econ. Dyn. Control* **2003**, *27*, 2207–2218. [CrossRef]
225. Movahedi, Z.; Bastanfard, A. Toward competitive multi-agents in Polo game based on reinforcement learning. *Multimed. Tools Appl.* **2021**, *80*, 26773–26793. [CrossRef]
226. McKee, K.R.; Gemp, I.; McWilliams, B.; Duéñez-Guzmán, E.A.; Hughes, E.; Leibo, J.Z. Social diversity and social preferences in mixed-motive reinforcement learning. *arXiv* **2020**, arXiv:2002.02325.
227. Brown, N.; Sandholm, T. Superhuman AI for multiplayer poker. *Science* **2019**, *365*, 885–890. [CrossRef] [PubMed]
228. Ye, D.; Chen, G.; Zhang, W.; Chen, S.; Yuan, B.; Liu, B.; Chen, J.; Liu, Z.; Qiu, F.; Yu, H.; et al. Towards playing full moba games with deep reinforcement learning. *Adv. Neural Inf. Process. Syst.* **2020**, *33*, 621–632.
229. Vinyals, O.; Babuschkin, I.; Czarnecki, W.M.; Mathieu, M.; Dudzik, A.; Chung, J.; Choi, D.H.; Powell, R.; Ewalds, T.; Georgiev, P.; et al. Grandmaster level in StarCraft II using multi-agent reinforcement learning. *Nature* **2019**, *575*, 350–354. [CrossRef]
230. Selsam, D.; Lamm, M.; B$\grave{}$unz, B.; Liang, P.; de Moura, L.; Dill, D.L. Learning a SAT solver from single-bit supervision. *arXiv* **2018**, arXiv:1802.03685.
231. Li, Z.; Chen, Q.; Koltun, V. Combinatorial optimization with graph convolutional networks and guided tree search. *Adv. Neural Inform. Process. Syst.* **2018**, arXiv:1810.10659.
232. Lemos, H.; Prates, M.; Avelar, P.; Lamb, L. Graph colouring meets deep learning: Effective graph neural network models for combinatorial problems. In Proceedings of the 2019 IEEE 31st International Conference on Tools with Artificial Intelligence (ICTAI), Portland, OR, USA, 4–6 November 2019; pp. 879–885.
233. Prates, M.; Avelar, P.H.; Lemos, H.; Lamb, L.C.; Vardi, M.Y. Learning to solve np-complete problems: A graph neural network for decision tsp. In Proceedings of the AAAI Conference on Artificial Intelligence, Honolulu, HI, USA, 27 January–1 February 2019; Volume 33, pp. 4731–4738.
234. Li, Y.; Tarlow, D.; Brockschmidt, M.; Zemel, R. Gated graph sequence neural networks. *arXiv* **2015**, arXiv:1511.05493.
235. Gilmer, J.; Schoenholz, S.S.; Riley, P.F.; Vinyals, O.; Dahl, G.E. Neural message passing for quantum chemistry. In Proceedings of the International Conference on Machine Learning, Sydney, Australia, 6–11 August 2017; pp. 1263–1272.
236. Defferrard, M.; Bresson, X.; Vandergheynst, P. Convolutional neural networks on graphs with fast localized spectral filtering. *Adv. Neural Inform. Process. Syst.* **2016**, arXiv:1606.09375.
237. Kipf, T.N.; Welling, M. Semi-supervised classification with graph convolutional networks. *arXiv* **2016**, arXiv:1609.02907.

238. Hu, H.; Zhang, X.; Yan, X.; Wang, L.; Xu, Y. Solving a new 3d bin packing problem with deep reinforcement learning method. *arXiv* **2017**, arXiv:1708.05930.
239. Nazari, M.; Oroojlooy, A.; Snyder, L.; Takác, M. Reinforcement learning for solving the vehicle routing problem. *Adv. Neural Inform. Process. Syst.* **2018**, arXiv:1802.04240.
240. Venkatakrishnan, S.B.; Alizadeh, M.; Viswanath, P. Graph2seq: Scalable learning dynamics for graphs. *arXiv* **2018**, arXiv:1802.04948.
241. Manchanda, S.; Mittal, A.; Dhawan, A.; Medya, S.; Ranu, S.; Singh, A. Gcomb: Learning budget-constrained combinatorial algorithms over billion-sized graphs. *Adv. Neural Inform. Process. Syst.* **2020**, *33*, 20000–20011.
242. Dai, H.; Dai, B.; Song, L. Discriminative embeddings of latent variable models for structured data. In Proceedings of the International Conference on Machine Learning, New York, NY, USA, 24–26 June 2016; pp. 2702–2711.
243. Song, J.; Lanka, R.; Yue, Y.; Ono, M. Co-Training for Policy Learning. Uncertainty in Artificial Intelligence. 2020; pp. 1191–1201. Available online: https://arxiv.org/abs/1907.04484 (accessed on 1 May 2022).
244. Ma, Q.; Ge, S.; He, D.; Thaker, D.; Drori, I. Combinatorial optimization by graph pointer networks and hierarchical reinforcement learning. *arXiv* **2019**, arXiv:1911.04936.
245. Abe, K.; Xu, Z.; Sato, I.; Sugiyama, M. Solving np-hard problems on graphs with extended alphago zero. *arXiv* **2019**, arXiv:1905.11623.
246. Kwon, Y.D.; Choo, J.; Kim, B.; Yoon, I.; Gwon, Y.; Min, S. Pomo: Policy optimization with multiple optima for reinforcement learning. *Adv. Neural Inform. Process. Syst.* **2020**. *33*, 21188–21198.
247. Barrett, T.; Clements, W.; Foerster, J.; Lvovsky, A. Exploratory combinatorial optimization with reinforcement learning. In Proceedings of the AAAI Conference on Artificial Intelligence, New York, NY, USA, 7–12 February 2020; Volume 34, pp. 3243–3250.
248. Wu, Y.; Song, W.; Cao, Z.; Zhang, J.; Lim, A. Learning Improvement Heuristics for Solving Routing Problems. *IEEE Trans. Neural Netw. Learn. Syst.* **2021**. [CrossRef]
249. Lu, H.; Zhang, X.; Yang, S. A learning-based iterative method for solving vehicle routing problems. In Proceedings of the International Conference on Learning Representations, New Orleans, LA, USA, 6–9 May 2019.
250. Xu, R.; Lieberherr, K. Learning self-game-play agents for combinatorial optimization problems. *arXiv* **2019**, arXiv:1903.03674.
251. Kruber, M.; Lubbecke, M.E.; Parmentier, A. Learning when to use a decomposition. In Proceedings of the International Conference on AI and OR Techniques in Constraint Programming for Combinatorial Optimization Problems, Padova, Italy, 5–8 June 2017; Springer: Berlin/Heidelberg, Germany, 2017; pp. 202–210.
252. Gomory, R. *An Algorithm for the Mixed Integer Problem*; Technical Report; RAND Corp.: Santa Monica, CA, USA, 1960.
253. Joshi, C.K.; Laurent, T.; Bresson, X. An efficient graph convolutional network technique for the travelling salesman problem. *arXiv* **2019**, arXiv:1906.01227.
254. Golmohamadi, H. Operational scheduling of responsive prosumer farms for day-ahead peak shaving by agricultural demand response aggregators. *Int. J. Energy Res.* **2020**, *45*, 938–960. [CrossRef]
255. Mouret, S.; Grossmann, I.E.; Pestiaux, P. A new Lagrangian decomposition approach applied to the integration of refinery planning and crude-oil scheduling. *Comput. Chem. Eng.* **2011**, *35*, 2750–2766. [CrossRef]
256. Li, M. Modeling Method Research on Refinery Process Production Shceduling. Ph.D. Thesis, Shandong University, Jinan, China, 2011.
257. Yue, X.; Wang, H.; Zhang, Y.; Shi, B. Optimization of refinery crude oil scheduling based on heuristic rules. *Comput. Appl. Chem.* **2020**, *2*, 147–154.
258. Hou, Y.; Wu, N.Q.; Li, Z.W.; Zhang, Y.; Zhu, Q.H. Many-Objective Optimization for Scheduling of Crude Oil Operations based on NSGA-III with Consideration of Energy Efficiency. *Swarm Evol. Comput.* **2020**, *57*, 100714. [CrossRef]
259. Assis, L.S.; Camponogara, E.; Grossmann, I.E. A MILP-based clustering strategy for integrating the operational management of crude oil supply. *Comput. Chem. Eng.* **2020**, *145*, 107161. [CrossRef]
260. Beach, B.; Hildebrand, R.; Ellis, K.; Lebreton, B. An Approximate Method for the Optimization of Long-Horizon Tank Blending and Scheduling Operations. *Comput. Chem. Eng.* **2020**, *141*, 106839. [CrossRef]
261. Li, M. Refinery Operations Optimization Integrated Production Process and Gasoline Blending. *J. Phys. Conf. Ser.* **2020**, *1626*, 12111. [CrossRef]
262. Bayu, F.; Shaik, M.A.; Ramteke, M. Scheduling of crude oil refinery operation with desalting as a separate task. *Asia-Pac. J. Chem. Eng.* **2020**, *15*, e2539. [CrossRef]
263. Zhang, S.; Xu, Q. Refinery continuous-time crude scheduling with consideration of long-distance pipeline transportation. *Comput. Chem. Eng.* **2015**, *75*, 74–94. [CrossRef]
264. Oliveira, F.; Nunes, P.M.; Blajberg, R.; Hamacher, S. A framework for crude oil scheduling in an integrated terminal-refinery system under supply uncertainty. *Eur. J. Oper. Res.* **2016**, *252*, 635–645. [CrossRef]
265. Wang, H. Batch Optimization Combined with AI Ideas for Refinery Oil Pipeline Networks. Ph.D. Thesis, China University of Petroleum, Beijing, China, 2020.
266. Gao, H.; Xie, Y.; Ma, J.; Zhang, B. Optimization of refined oil distribution with multiple trips and multiple due time. *Control Decis.* **2021**, 1–10. [CrossRef]
267. Li, M.; Huang, Q.; Zhou, L.; Ni, S. Research on modeling of petroleum products distribution system based on object-oriented Petri nets. *Comput. Eng. Appl.* **2015**, *51*, 55–61.

268. Li, Z.; Su, T.; Zhang, L. Application Analysis and Prospect of Artificial Intelligence Technology in Smart Grid. *Telecom Power Technol.* **2020**, *37*, 2.
269. Sheikhi, A.; Rayati, M.; Bahrami, S.; Ranjbar, A.M.; Sattari, S. A cloud computing framework on demand side management game in smart energy hubs. *Int. J. Elect. Power Energy Syst.* **2015**, *64*, 1007–1016. [CrossRef]
270. Fan, S.; Li, Z.; Wang, J.; Longjian, P.; Ai, Q. Cooperative Economic Scheduling for Multiple Energy Hubs: A Bargaining Game Theoretic Perspective. *IEEE Access* **2018**, *6*, 27777–27789. [CrossRef]
271. Peng, X.; Tao, X. Cooperative game of electricity retailers in China's spot electricity market. *Energy* **2018**, *145*, 152–170. [CrossRef]
272. Chen, J.; Bo, Y.; Guan, X. Optimal demand response scheduling with Stackelberg game approach under load uncertainty for smart grid. In Proceedings of the IEEE Third International Conference on Smart Grid Communications, Tainan, Taiwan, 5–8 November 2012.
273. Li, Y.; Wang, C.; Li, G.; Chen, C. Optimal Scheduling of Integrated Demand Response-Enabled Integrated Energy Systems with Uncertain Renewable Generations: A Stackelberg Game Approach. *Energy Convers. Manag.* **2021**, *235*, 113996. [CrossRef]