smart cities

MDPI

Article

A Survey of Adaptive Multi-Agent Networks and Their Applications in Smart Cities

Nasim Nezamoddini [1,*] and Amirhosein Gholami [2]

[1] Industrial and Systems Engineering Department, Oakland University, Rochester, MI 48309, USA
[2] Systems Science and Industrial Engineering Department, Binghamton University, Binghamton, NY 13902, USA; agholam1@binghamton.edu
* Correspondence: nezamoddini@oakland.edu; Tel.: +1-(248)-370-2989

Abstract: The world is moving toward a new connected world in which millions of intelligent processing devices communicate with each other to provide services in transportation, telecommunication, and power grids in the future's smart cities. Distributed computing is considered one of the efficient platforms for processing and management of massive amounts of data collected by smart devices. This can be implemented by utilizing multi-agent systems (MASs) with multiple autonomous computational entities by memory and computation capabilities and the possibility of message-passing between them. These systems provide a dynamic and self-adaptive platform for managing distributed large-scale systems, such as the Internet-of-Things (IoTs). Despite, the potential applicability of MASs in smart cities, very few practical systems have been deployed using agent-oriented systems. This research surveys the existing techniques presented in the literature that can be utilized for implementing adaptive multi-agent networks in smart cities. The related literature is categorized based on the steps of designing and controlling these adaptive systems. These steps cover the techniques required to define, monitor, plan, and evaluate the performance of an autonomous MAS. At the end, the challenges and barriers for the utilization of these systems in current smart cities, and insights and directions for future research in this domain, are presented.

Keywords: MAS; adaptive systems; network systems; smart city; systems of systems

Citation: Nezamoddini, N.; Gholami, A. A Survey of Adaptive Multi-Agent Networks and Their Applications in Smart Cities. *Smart Cities* **2022**, *5*, 318–347. https://doi.org/10.3390/smartcities5010019

Academic Editor: Zhixiang Fang

Received: 11 January 2022
Accepted: 7 March 2022
Published: 9 March 2022

1. Introduction

The smart city is a new notion that has rapidly gained ground in the agendas of city authorities all over the world. An increasing population concentration in urban areas and subsequent arising challenges have highlighted the need for intelligent ways to facilitate citizen's lives, deliver services, and mitigate against disasters [1]. One of the solutions was to introduce new urban areas equipped with a city with an advanced metering infrastructure and smart objects with ubiquitous sensing and embedded intelligence [2]. Each one of the smart objects collects the data from their environment, communicates with other objects, process information, and in some cases, autonomously react to dynamic internal and external changes [3]. Wireless sensor networks (WSN), radio frequency identification (RFID), near-field communications (NFC) tags, unique/universal/ubiquitous identifiers (UID), actuators, smartphones, and smart appliances are examples of these smart devices. The devices are connected through a platform called the Internet-of-Things (IoTs) that allows technologies to access and interchange data through wireless and sired internet networks [4].

Although adopting the IoTs opens new possibilities and opportunities to change our society to a connected world, at the same time, it brings its own problems and risks. Integrating a diverse range of devices with different functionalities, computation capabilities, and data streams is extremely challenging [5]. Scalability is another major issue for controlling IoTs systems, considering the highly dynamic and distributed nature of these networked systems [6]. Each of the sensing devices and other end users may join or

leave the system or change their locations at any time, and that makes the topology of the system uncertain and subject to unexpected changes. The control mechanisms with fixed configurations are not efficient for the heterogeneous and dynamic nature of IoTs systems. The distributed nature of the IoTs also makes it more vulnerable to possible cyber-attacks and failures. Failures in part of the system, especially in a central leader, may cascade to other connected nodes and eventually result in the collapse of the whole system. These systems need to be designed and controlled in a more robust and resilient way, promoting the efficiency of the system in delivering services and achieving its predefined targets and goals.

The distributed processing and control of multi-agent systems (MAS) or agent-oriented programming (AOP) are some of the main technological paradigms for the efficient deployment of smart devices and services in the smart city [7]. These techniques are considered the best abstraction approaches for modeling the operations and functionalities of IoTs systems in all three layers of perception, network, and application [8]. They also proved their effectiveness in supporting autonomous networks of the objects in the IoTs with self-adaptive and self-organizing properties [9]. There is a large number of instantaneous communications between devices in the IoT. In MAS, each device is mapped to an agent with a predefined range of features and capabilities. This mapping provides a suitable high-performance infrastructure for testing and implementing large-scale data acquisition and offers a scalable platform for the distributed computing of the received data [10]. Moreover, simulating objects as smart-reactive agents enables the real-time tuning of control parameters in complex interconnected networks, such as the IoT. The autonomous agents in the MAS can be programmed and function without human intervention. They are also able to interact with other agents and reflect modular functionalities and coordination. The agents in the MAS also reflect goal-oriented behavior, which is one of the main requirements for managing connected devices in smart cities [11]. These advantages make them suitable platforms for implementing different domains of a smart city, including mobility, environment, governance, energy, economy, health, technology, and education [12]. Figure 1 summarizes some of the related literature for the application of multi-agent systems in different applications in a smart city.

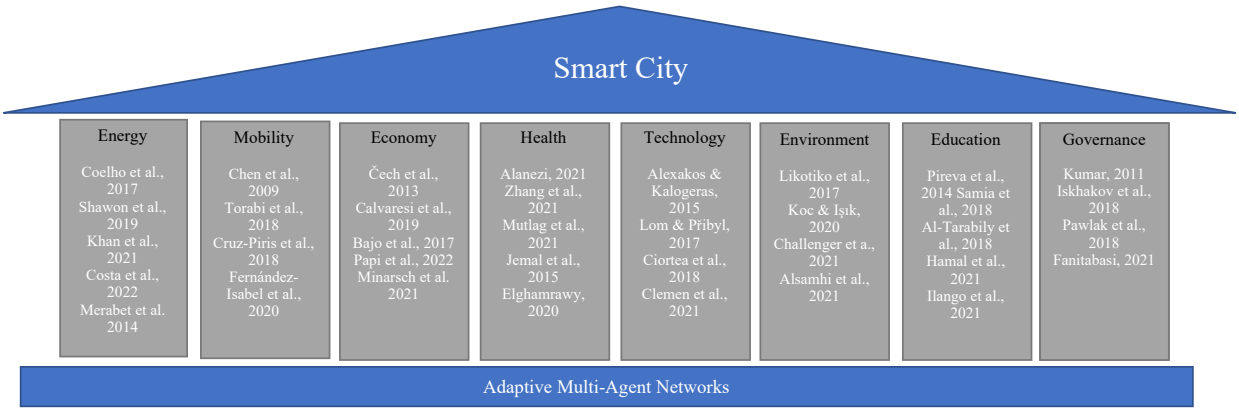

Figure 1. Multi-agent systems in smart city applications.

Self-adaptiveness is one of the main requirements in designing pervasive interconnected systems, such as the IoTs and smart cities [13]. The cooperative nature of MAS-based systems and their learning capabilities enable the designing of robust solutions that can adapt their configuration and operations when facing unexpected variations and disruptions [14]. To achieve these abilities, it is necessary to learn from previous events, such as abrupt environmental changes and internal misfunctions, and adapt the controlling parameters and strategies continuously. The overview of a self-adaptive system is presented in Figure 2. A self-adaptive MAS constantly receives feedback from internal units and the external environment. The received data is monitored and analyzed, and perceptions are fed into

the planning module to decide the next action and response of the system. Control commands are defined considering system's main goal and the units' roles and responsibilities in achieving the desired level. Then, actions are sent back to the actuating units and outcomes are collected for updating the next actions and eventually system evaluation and modifications.

Existing survey articles in the literature cover relatively narrow topics of multi-agent systems and fail to provide a broad understanding of these systems and their utilization in IoTs platforms. Some researchers only focused on particular applications of these systems [8,15–18] , while others reviewed variations of a certain control technique [19–21]. Successful applications of MASs in real-world smart cities require a more comprehensive systematic perspective, integrating different steps of system definition, monitoring, planning, and evaluation. This paper enhances the previously published papers related to MAS systems with a systematic survey of the steps required to realize future IoTs platforms and smart cities. Specifically, we focus on the techniques that enable MAS to perceive system status and environment dynamics and react autonomously to overcome unexpected changes and disruptions. The rest of the paper is organized as follows: In Section 2, all factors required for defining MAS frameworks are reviewed. Correspondingly, Section 3 investigates the state of the art of the models that can be applied for data monitoring, and the status measurement of these systems. The concepts and techniques for planning and controlling MASs are listed in Section 4. The common platforms for validating and evaluating these systems are presented in Section 5. The last section provides insights to existing gaps and open issues that can be addressed by this research community to achieve fully operative autonomous IoTs in future smart cities.

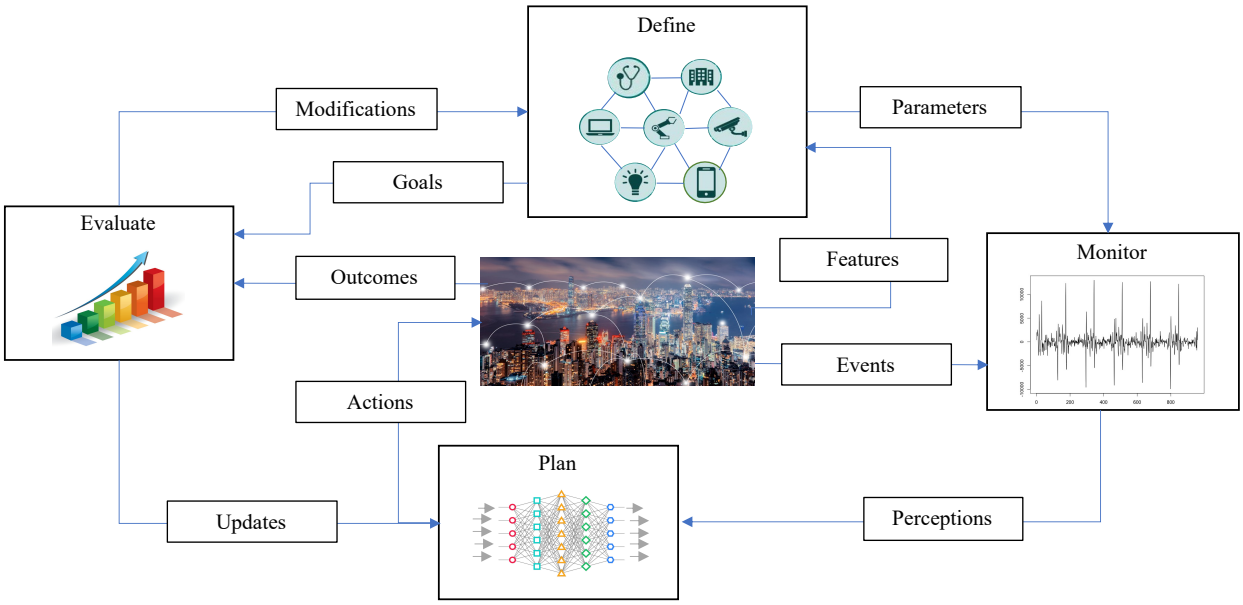

Figure 2. Overview of a self-adaptive MAS.

2. Definition Frameworks

The first step for designing an adaptive multi-agent network is to define the key features and requirements of the system. These systems can be identified based on the characteristics of their entities, operations and actions, information flow, and the external environment that they operate in [8]. Summary of the influencing factors in defining MASs is presented in Figure 3.

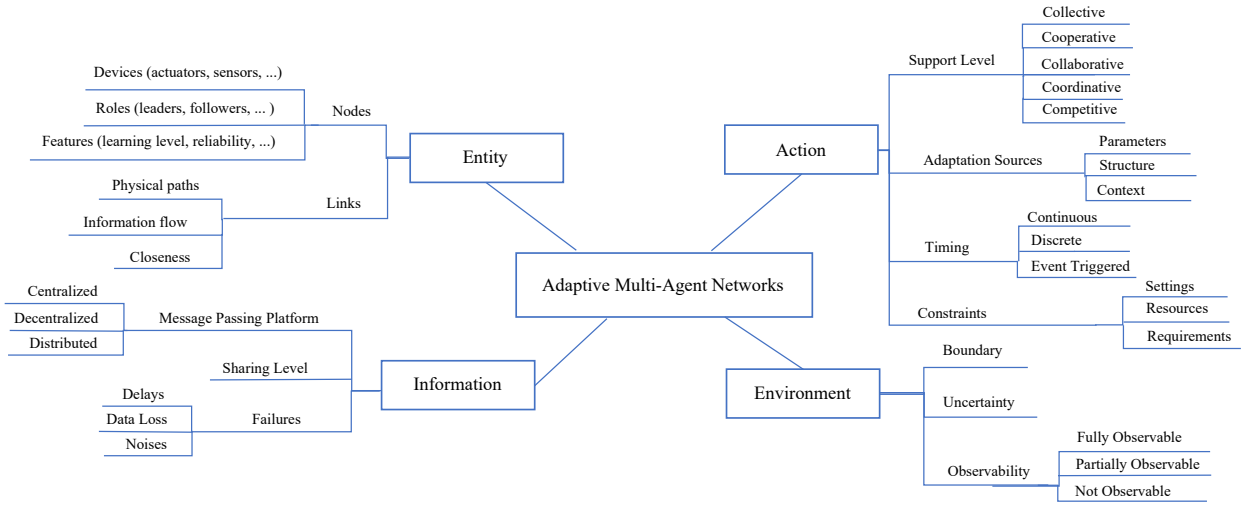

Figure 3. Contributing factors in defining adaptive MASs.

2.1. Entities

A MAS system is a network of multiple entities that may or may not interact with each other. In most of the MAS literature, these systems are mapped into a graph $G(v, \epsilon)$ with a non-empty set of nodes or agents $v = \{v_1, ..., v_N\}$ and edges ϵ connecting these agents [22]. Entity defines the type of these agents and their communication links that can be same or different in a MAS platform. Nodes in adaptive multi-agent networks varied in their assigned devices, roles, features, and dynamics. They can be defined as agents reflecting either software or hardware devices. In the IoTs platforms, a node can be a sensor to collect the data, an actuator to perform commands, or a combination of both [23]. Nodes are also defined based on the role/roles that they play in the network. For example, the agents may play leader or follower roles in MASs. The leader is the main decision maker in these systems, and has access to the target settings [24]. Other agents called "followers" just mimic and minimize their distances from the leader. Some agents are listeners that can only observe their relative position and environment, while other speaker agents are able to produce a communication output with other nodes and agents [25]. In the literature, there are more complex roles, such as meta-agents, that can perform high-level responsibilities, such as reasoning for other special-purpose agents [26]. Nodes may also play the role of external observer or resource, either collecting status data or feeding other operating agents [27].

In real-world heterogeneous MASs, agents/nodes also differ based on their features and settings, including accessibility, energy usage, authority level, and learning and information processing capabilities [28]. Sometimes agents are identified based on their reliability and performance quality in the system. Reliability is the degree of reliance that a system can place on the agent and its information and services. For example, in swarm optimization, agents are categorized and distinguished based on the quality of their solutions in the previous iteration [29]. In the IoTs systems, some agents and elements may be self-interested, with a lack of a global perspective of the system [30], and the potential to inject unreliable and misleading information into the system, which needs extra considerations during modeling and evaluation states [31].

The agents also differ based on their dynamics and mathematical descriptions, explaining the movement and evolution of the agents over time [32]. These changes are the result of injecting a control input in the agent. These dynamics can be expressed as linear or nonlinear models [33]. In linear models, we may be faced with a first-order single integrator that all agents converge to constant values, or double-integrator differential equations, which agents may converge to multiple final states [34]. These models are special forms of general linear models, where the agents are influenced with state and input matrix parameters. Nonlinear agent models also include Lagrangian systems, unicycle

models, and the attitude dynamics of rigid bodies [32]. In the majority of nonlinear models, it is assumed that agents' nonlinear functions are either unknown or contain unknown parameters [35], and this uncertainty needs to be addressed by either limiting conditions or approximation techniques, such as fuzzy logic systems [36] and neural networks [37]. Each one of these models use different sets of equations to explain the dynamics of individual agents. Depending on the number of agreement metrics, agents' dynamics can be also categorized as first-, second-, and higher-order models [8].

Other than individual nodes, edges and joint connections between the agents play an important role in the controllability of MASs [38,39]. These interactions enable cooperation between agents and reflect communication paths, such as wires, plans, and routes in the network. They are defined to investigate the physical entity flow or information sharing between agents [40]. Agents in MAS platforms exchange their local states and control commands with other neighboring nodes or influencing nodes. The communication platform is shown as an undirected graph in cases that both agents can communication with each other. In the cases that there is a one-way information flow between agents, directed graphs are used to reflect these interactions [41]. The links and their weights can be also applied to show the similarities and closeness of agent pairs in the system.

2.2. Actions

Multi-agent systems can be also investigated from an action standpoint and distinguished based on their support level, requirements, adaptation sources, timing, and constraints that they may face during their operations. Actions or decisions can be defined in terms of policies that determine a set of actions or probabilities for selecting actions in each state of the system [42]. These policies are either deterministic or stochastic policies. In deterministic policies, the optimal fixed decisions are determined for each state of the agent, while in stochastic policies, the probability distribution of the actions is defined. In nonstationary systems, the decisions vary with time as well [43].

Support level refers to the agents' supports and reactions versus other agents' actions and decisions. For example, they may follow the same or different goals. Minimizing operation errors, maintaining a certain status, or maximizing system performance over time are examples of these objectives [44]. There are also cases in which agents follow more than one goal, which require more complex reward structures [43]. In systems that agents have common goals, depending on on their awareness, they may fall into one of the collective or cooperative MASs [45]. In collective MASs such as robot formations, the agents have the same goal, but agents independently perform their own tasks and explore their possible contributions to accomplish the system's main goal [46]. Swarm intelligence is also an example of the collective systems in which agents optimize the main objective function, and agents' actions are guided by the partial information sharing of successful agents with other exploring agents [47]. In cooperative MASs, agents are aware of other agents and share their local information to help them achieve their common goals. Consensus and rescue agents are two well-known examples of such systems [21]. In systems with nonidentical goals, we may experience negative or positive interactions between agents. Negative interactions refer to the agents competing for shared resources or agents with conflicting individual goals [48]. In systems with positive interactions such as path planning, the agents only focus on their own operations while minimizing interference with other agents [40]. In collaborative MASs such as machine learning techniques, although agents have different goals, they help other agents by sharing their experience and knowledge from their environment and rewarding system. There are also examples of multi-agent systems with combined positive cooperative intra-group and negative competitive inter-group interactions [49].

It should be noted that not all goals in real-world problems are functional and some are defined to meet certain requirements, such as reliability and the system's tracking error [50]. It was shown that the individual contribution of the agents on the global solution can be quantified and specified by a control mechanism [51]. Requirement constraints

usually reflect the desired output defined by the decision maker. Synchronization, tracking, and estimation errors are examples of such constraints in the system [24,52]. Deadlines for accomplishing tasks can be also considered another form of the constraints for the MAS [53].

Actions are also categorized based on the source of their adaptations. Adaptation techniques are defined in three main groups of parameters, structure, and context [13]. Similarly in MASs, adaptations are achieved by changing a system's decisions and behaviors through parameters and network structures. These parameters can be defined at macro-level as system parameters or at micro-level in terms of agents' actions and decisions. The environment or context can be also altered using the decisions of the actuator agents in the system. In some MASs such as formation and tracking controls, the agents only focus on their locating decisions [19]. In other applications such as distributed computing, their parameter decisions may represent their beliefs and estimations for certain variables and system settings [54]. Settings for the level of resource or service that agents provide for other agents or borrow from other agents are other examples of parameter settings [55]. Adaptation can be also achieved by changing the structure and topology of the system. Underlying topology can be fixed or switched over time because of unreliable transmission or limitations in the communication and sensing range of the agents. At the same time, new agents may join the system and start to create new connections and change the neighborhood map of the previous agents. Sometimes communications may fail due to link failures between two agents [56].

Actions are also distinguished based on their timing and sequences [57]. The dynamics of the MAS can be shown for both continuous-time [58] and discrete-time systems [38]. Other than time-based control techniques, we may define event-triggered mechanisms that reduce network congestion. These techniques initiate and release control commands only after detecting triggering conditions defined based on certain error thresholds [59]. Any sampling [60], transmission [61], estimation [62], or control [63] can be modified to an event-triggered mechanism.

The actions of MASs are subject to the constraints in their settings, resources, and requirements [64,65]. In an IoTs system in which agents represent devices in the system, the constraints can reflect settings, technical limitations, or working range of the devices in the network [21]. For example, it is necessary to consider the sensing limitations of the wireless sensor agents or computation capabilities of the agents in distributed processing. In practical applications, it is also necessary to model boundaries for the operations and inputs of actuator agents. These limitations can be modeled as input saturation constraints during the MAS modeling phase [66]. Communication constraints are caused by physical obstacles or artificial settings [67]. They can influence both agents' actions and system contexts by limiting network congestion in the system [67]. For example, physical barriers may limit the movements and actions of the multi-robot systems and need to be modeled as state constraints of the agents [68]. Most of the real-world multi-agent systems are also subject to constraints in resources, such as data sampling, computation, memory, and processing resource availability. The energy constraint of the individual agents is another constraint affecting the design and control of the distributed MASs [69]. Bandwidth capacity is a constraint that affects the agents' communications with each other and controller units [70].

2.3. Environment

Agents interact and influence the environment during their operation period. The environment can be defined based on its boundary, uncertainty, and observability levels. There is no exact definition for the boundaries of the environment in MASs, but in most of the literature, any external surrounding condition of the agents is defined as an environment [71]. Environment boundaries mainly depend on MAS application and in general, they include a shared physical, communication, and social structure and space of the agents, and resources and services defining the constraints, interaction rules, and relations between them [72]. In fact, this abstraction is considered an external world that provides a median

for coordination, maintaining the independence of the processes from the actions of the operating agents. The environment boundaries are also defined based on their interaction mediation and resource and context management mechanisms [73]. Other external entities interacting with agents or monitored by them are also considered part of the environment. For example, targets in tracking problems or reference models are part of the external environment in these problems [74]. In multi-agent programming, the environment is models based on aspects such as the action model, agents' perception model, a computational model for internal and external functionalities, the data exchange model between agents and the environment, and agents' distribution model [71].

In most of the real-world MASs, especially in the IoTs domain, agents operate in an uncertain environment. Therefore, it is necessary to investigate the environment's level of uncertainty and underlying hidden dynamics. The systems may perform in a deterministic environment [75] or be subject to a variety of uncertainties that affect the controllability of the MAS [76]. Uncertainty can be caused because of an unknown value function or constraint of the system [44,77]. For example, the location of other agents in path planning [78] and agents' behaviors and parameter settings in a heterogeneous control [79] can be considered as a source of uncertainties in the environment of the agent. Uncertainty in communications is another source that becomes critical in MAS networks, such as multi-robot systems [80]. Unknown disturbances can be one of the uncertainty sources in the MASs environment [81]. Sometimes uncertainty is a result of unmodeled dynamics between agents and the environment [82]. The environment can be uncertain in such a way that targets appear at random times [83]. Uncertainty may also originate in one of the environment parameters [84]. Uncertainty in the environment can be also reflected by injected noises in the system that affect the state of the agents [85]. The main source of the uncertainty arises from the operations of the other agents that make the environment unpredictable. These dynamics make the environment nonstationary due to the simultaneous learning of the agents for the best policies that change the overall state of the environment constantly [86]. Uncertainty in the environment is usually modeled by stochasticity in process equations, output/measurement equations, or communication channels between the agents [20].

In theory, the environment can be modeled on one of the categories of the fully observable, partially observable, or not observable categories [87]. In fully observable systems, agents are able to perceive complete information about the state of the environment and its elements, such as resources and other agents' states [74]. Agents in a partially observable environment, such as a partially observable MDP (POMDP), are only able to learn partial information about the states and their probability distributions instead of absolute values [88]. In these systems, the relations between actions and rewards are not clear and need to be estimated [89]. In a more extreme case, the agents may not be able to perceive anything about the environment and blindly act on their tasks and responsibilities. Visibility plays an important role in the controllability and efficiency of the MASs [90]. Most of the developed techniques are based on an unrealistic assumption for the observability of the environment and agents are always under constraints, forcing them to learn the abstractions of the environment instead of the details. One of the main assumptions for the observability of the environment for agents' actions and their rewards and penalties is that of the Markovian environment [91]. In this environment, the future state of the agents is determined by their current states. In non-Markovian environments, state dynamics are more complex and there may be strong dependencies on initial states or changes that are episodic over time [92]. Observability can be also limited by agents' information received from other agents. For example, they may not have access to the communication network all the time and get disconnected for a period of time [90]. The agents may also have noisy observations affecting their perceptions from the environment and other agents [93].

2.4. Information Flow

To design an effective MAS control mechanism, it is necessary to pay attention to the information flow between agents that can be defined in the forms of its message passing platform, information sharing level, and communication failures over time [94]. Information flow in an adaptive multi-agent network is implemented either for collecting agent status and outcomes and sending them to controllers or for delivering commands from controllers to interacting agents.

The message passing platform reflects the access level of the agents to high-level information about system status, goals, and constraints. Three main platforms for information flow between agents and controllers include centralized, layers or decentralized, and distributed frameworks (Figure 4). In a centralized approach, one agent has access to agent information and decides for the whole system [95,96]. Scalability, vulnerability to the controller agent failure, and a need for high communication resources for agents are common problems for this approach. To overcome this issue, some researchers used assumptions such as the parametric similarity of the agents to approximate all policies in only one unique policy [97]. After receiving a common policy in this framework, agents can locally explore their policies to minimize their own losses. The common policy is updated interactively using the feedback received from the trajectories of all agents. These techniques are considered as hybrid forms with some levels of decentralized information updates and flow between agents. In decentralized or layered approaches, the system is controlled by more than one controller in each layer or community [98]. This technique is considered more reliable than the previous technique, but it requires a precise definition of the communication and cooperation rules between the controllers. The distributed form is considered the most common technique in controlling MASs, where each agent is responsible for deciding and coordinating with other agents to achieve the main goal of the system [99]. In this technique agents have full autonomy to select their own actions [100]. This technique is more robust and scalable. However, it is more challenging in terms of finding the best decision of the agents given their limitations in exploration and information access. None of these platforms are considered a best option, and depending on the system goals and resources, one specific platform is chosen.

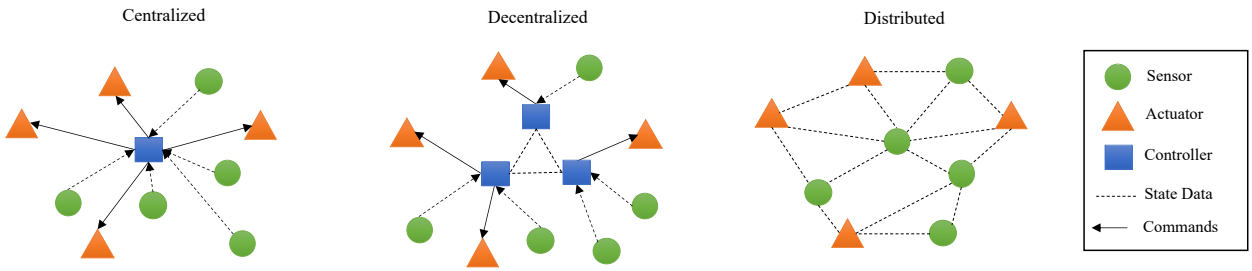

Figure 4. MAS information flow platforms.

It is also necessary to consider the knowledge and information sharing levels between agents [101]. It was shown that the degree of information sharing directly affects the learning process of the agents and system efficiency [102]. Some of the agents have minimum information-sharing regarding expected rewards, agent configuration, or adaptation tactics [28]. Although in most of the existing literature it is assumed that agents are only able to communicate with their neighboring agents, some studies targeted other types of peer-to-peer communications between agents, such as broadcasting or communications through middle agents [94]. The middleware can be a matchmaker that manages and matches the right information to the right agent or a broker that filters or rewords the communicated information and then distributes it to the related agents. The agents may also use a trace manager that receives the data and sends it to the subscribed agents based on their interests and registered requests. In the literature, more flexible information sharing was proposed, whereby agents are able to compare and choose their own communication platforms [103].

They can also have the authority to select their communication source from a list of multiple signal sources [104]. On the other hand, targeted communications architecture helps agents to choose the contents and receivers of their messages in cooperative, competitive, or mixed environments [105].

These communication and information dynamics are subject to failures and challenges, such as delays, data losses, quantization errors, noises, unknown dynamics, fading channels, nodes-access competition, and sampling intervals [60]. In communication latencies, state information of the neighboring agents is received by delays due to limitations on the capacity of communication channels and network congestion [106]. In some cases, delays are not fixed and vary due to differences in the observability of the state over time [107]. This is different from input delays that reflect the processing and connecting times of the incoming data flow in the MASs [108]. The other influencing factor is to consider data losses in defining MAS communications. The packet dropout rate in the network is usually variable and stochastic due to fluctuations in the power supply and the traffic of the system, and they are usually modeled by the Bernoulli process [109]. Information dynamics are also designed based on missing information rates during communications and the usefulness of the collected information for their decision-making process [110]. Noises in agents' statuses and measurements are another challenge influencing the quality of the information flow in multi-agent networks [111]. Two major types of noises are additive and multiplicative noises. The noises caused by external sources are modeled as additive-influencing nodes measurements. The multiplicative noises are the results of missing information and a failure in modeling the internal dynamics of the system, and influence agents' states. They are also modeled as random noise with known probabilities and unknown non-random noise with bounded energy or bounded magnitude [62].

3. Monitoring Paradigms

To survive in dynamic environments, an adaptive multi-agent system requires the constant monitoring of its internal and external states. The data collected by distributed sensing agents are processed to explore the environment, evaluate system performance, and define the next optimal actions. The data analysis can be implemented in a central processor or in distributed local processors. The centralized single processing of the data will be very challenging given the complex relations between agents and the high levels of multicollinearity between variables [96]. Traditional data mining techniques for flat vectorial data analysis are inefficient for handling large-scale data with inherent relational dependencies, weights, edge directions, and heterogeneity between system elements [112]. The other challenge for processing the data collected through MASs is resource limitation for transmission, processing, and storing high-dimensional data streams collected by the agents over time. In the literature, a wide range of techniques were proposed for alleviating the data collection burden, and event-triggered data sampling is one of the most acceptable approaches [62]. Distributed processing, data abstraction, and subgraph selection are other examples for reducing the computation times of data processing in MASs [65]. Some researchers also focused on developing innovative techniques for analyzing distributed data streams, which are mainly categorized as spatio-temporal data analyzing techniques [113]. In these techniques, deep learning algorithms such as graph neural networks (GNN), graph convolutional networks (GCN), graph autoencoders (GAE), graph recurrent neural networks (GRNN), or graph reinforcement learning were widely applied for processing the data collected from interconnected system elements or agents [114]. These approaches were applied for one of the data mining tasks, such as dimension reduction and prediction, and pattern mining, clustering, and anomaly detection in large-scale networks, such as multi-agent systems. The required analytics for these techniques can be implemented in agent or system levels. In this survey, we focus on system-level techniques and omit reviewing techniques in single-agent levels. This is mainly because most of the agent-level techniques are implemented on traditional vectorial data and are not specified for MAS platforms.

3.1. Dimension Reduction and Filtering

One of the main requirements for MASs is a way to abstract the overall data stream or filter unnecessary information collected over time. This is very important in distributed systems such as the IoTs and can considerably reduce the computation time of the main system. These techniques are also effective for clustering and categorizing the system status or their quick comparison with the target network.

Some initial techniques for abstracting MASs and their relations was the use of graph theory and the eigenvectors of the network adjacency matrix [115]. These techniques are not sufficient for complex systems with dynamic multi-dimensional agents. To solve this problem, encoding techniques such as GAE were applied for encoding or decoding graphs into vectors [114]. The encoding techniques are called network-embedding mechanisms and may encode the topological features of the nodes and their first- and second-order proximity information [116], agents' attributes [117,118], MAS dynamics and evolution over time [119], and information diffusion or dynamic role evolutions [120]. Generative network automata techniques are also applied for the simultaneous representation of the state transitions and topology transformations of the network based on graph rewriting concepts [121]. Other deep neural techniques that can be applied for MAS abstraction include long–short-term memory (LSTM) recurrent neural networks and variational autoencoders [122]. Some of these techniques can be also utilized for modeling opponent agents based on the local information of the main agents [123]. Some knowledge distillation techniques such as pruning and low-rank decomposition can be also applied for original multi-agent systems to remove redundant information collected from distributed agents [124]. There are other abstraction techniques in the literature that reduce the number of local states by collapsing data values. These techniques are mostly applied for the verification and testing of multi-agent systems [125]. Traditional dimension-reduction techniques such as principal component analysis were also modified successfully and applied for multi-agent networks, such as WSN [126]. The only drawback of these techniques is their high computation time for large-scale systems and losing the neighborhood pattern of the agents. This can be solved by relying more on the structural roles of nodes and increasing the flexibility of learning node representations [127]. Learning over a common communication grounding through autoencoding was another solution to reduce computation time and increase total system performance [128].

3.2. Anomaly Detection

The remote access of different devices and the distributed nature of computing in the IoTs makes it vulnerable to various attacks. Many different techniques were proposed to identify anomalies in distributed platforms, such as MAS [129]. Anomalies are states of the system that remarkably differ from normal system operations. These techniques can be applied for malicious interactions or attacks that may delete or manipulate the data related to the network structure and node or link statuses [130]. Anomalies can be in three different levels, such as point, contextual, or collective anomalies [114]. In point anomalies, irregularity happens in one agent that can be observed without any reason. Context abnormalities include a higher range of agent anomalies over time. For example, communication patterns may change in part of the MAS. In collective anomalies, agents alone may seem completely normal, but a collection of the data collected from agents shows unusual patterns. The survey for the anomaly detection in the node, edge, subgraph, and graph levels is reviewed in reference [131]. These anomalies can emerge in structural, attributed, or dynamic temporal graphs. In node-level anomaly detection, the agents that are significantly different from other agents are identified. In the IoTs platform, that can reflect abnormal users or a network intruder that injects fake information into the system [132]. These nodes also can represent agents with performances considerably deviating from the rest of the agents. In edge-level anomaly detection, unusual and unexpected connections and relations between agents are identified [133]. Subgraph-level anomaly detection focuses on multiple agents that collectively show anomalous behavior.

Identifying this group of agents will be very challenging and usually bipartite graphs are utilized to identify dense blocks in these networks [134]. Sometimes the anomalies are identified at a graph level and in certain snapshots of the temporal system, using its unusual evolving patterns and features. Deep neural networks such as LSTM were widely applied for this purpose [135]. Multi-agent systems are introduced as an efficient platform for anomaly detection techniques for IoTs systems [129]. Other techniques such as Kmean clustering proved their effectiveness for identifying unusual patterns of collected data in distributed networks [136].

3.3. Predictive Models

Predictive models can be applied at either system or entity levels for predicting certain features and characteristics over time [137]. For example, knowing and predicting future topological changes, such as removing and adding agents and new communications patterns, can have a substantial effect on designing control protocols. Deep graph generative models are widely used for this purpose to model a network structure without knowing its structural information [138]. They may also utilize recurrent neural networks for predicting future connections or edges in the graph based on node-ordering procedures [112]. There are other techniques that utilize efficient sampling strategies to extract patterns in input data and learn their dynamics to generate a predicted temporal network [139]. Other than structural patterns of the system, estimating future states of other agents can help agents to optimize their own actions more efficiently. It was proved that convolutional neural networks successfully captured the spatio-temporal characteristics of the networked entities and predicted their future features (nodes) and communications (edges) with neighboring agents [140]. As an example, predicting the future trajectories of the agents is one of the common problems in the literature, with a main application in robot planning, autonomous driving, and traffic prediction [141]. These estimation techniques are useful in missing data treatment of IoTs systems as well [142]. The other application of predictive models is in the early detection of events in event-triggered techniques or identifying hotspots with an unusual density of certain events [143]. Classification techniques for a network of connected agents can be applied for the performance monitoring of the system and initiating corrective actions if classified as an abnormal graph [144]. One of the common platforms for this purpose is DeepSphere, which was applied to learn the evolving patterns of a system over time and identify anomalous snapshots of the network over time [145]. The distributed estimation of certain features of the system or environment is another example for system-level predictive models. These techniques can either run at the same time or need consensus in their two sampling stages, and utilize a wide range of approaches, including the adaptive observer, Kalman filter, Luenberger observer, Bayesian filter, belief propagation inference, and the $H\infty$ filter [146].

3.4. Clustering

Monitoring MAS dynamics can be also targeted for clustering agents based on their performance or hidden interestsThis is mainly because grouping agents may help to identify non-cooperative agents or design more dedicated control frameworks for each group of the agents [147]. Sometimes clustering is helpful when heterogeneous agents show different dynamics in consensus, which requires more customized control protocols [148]. The similarities between a group of agents or their trajectories can be measured by dynamic correlations between agents [149] or using principal component analysis (PCA) projects [150]. Clustering agents can be investigated by applying traditional graph partitioning and community detection techniques, such as spectral clustering, hierarchical clustering, Markov models, and modularity maximization methods [151,152]. Deep learning is also considered an effective tool for community detection in high-dimensional multi-label graph-structured data [153]. This problem was also investigated in dynamic time-varying graphs, with the evolution of groups and their growth, contraction, merging, splitting, and birth and death over time [154]. The results show that the emergence of these communities is independent

of initial measurements and settings and mostly depends on node dynamics, underlying interaction graphs, and coupling strengths between agents [155]. Cluster dynamics are also investigated in controlling a swarm of agents in formation control and their aggregation, and splitting patterns and other quantitative features, such as size and cluster distances from each other [156]. Sometimes, clustering is a dynamic process in which agents with similar behaviors find each other and join together in the state space [157]. This will be similar to the dynamics of the group consensus techniques presented in the literature [158]. Other similar variations were proposed for defining monitoring agents to optimize the process of automated clustering [159].

3.5. Pattern Recognition

To design more robust and resilient MASs, they should be able to learn the behavior of other agents and environment dynamics to improve their performance over time. Pattern mining is considered one of the main techniques for knowledge discovery and identifying causal structures and associations [160]. For this purpose, graph pattern mining techniques are recognized as suitable approaches for knowledge discovery in multi-agent network systems. The pattern mining techniques can focus either on structural patterns such as frequent subgraphs, paths, cliques, and motifs, or the label evolution patterns of dynamic graphs with single or multiple attributes [161]. Knowledge discovery on graphs can be also applied to find periodic patterns that repeatedly are observed in agents' communications or states [162]. Sometimes considering more than one label in agents and their communications adds interesting findings for the dynamic changes of the attributed multi-agent graph. As an example, mining-trend motifs help to identify a group of nodes or agents that show similar increasing or decreasing trends over time [163]. More complex trends such as recurrent trends are identified in a set of nodes over a sequence of time intervals using algorithms, such as RPMiner [164]. The identified patterns in attributes and states may entail changes in the network topological structures and agnets communication, which are known as triggering patterns [165]. It was shown that in the pattern mining of multi-agent networks, relying on occurrence frequency is sometimes misleading and new metrics such as sequence virtual growth rate are necessary to identify highly correlated patterns with a significant trend sequence in the graph [166]. Norm mining is another category that investigates events triggering rewards and penalties and identifies the norms in a varying environment setting [167]. This will help agents to survive and adapt to their environment without the deprivation of their resources and services.

4. Development Approaches

Multi-agent systems are one of the main paradigms proposed for implementing the IoTs. The first step for developing these systems is to determine their main platform and define the features reviewed in Section 2. Then, appropriate learning mechanisms are selected to empower their adaptiveness against a new environments and possible changes in external and internal dynamics. At the end, a suitable control mechanism is designed to guarantee achieving the main goals of the system.

4.1. Main Platform

Adaptive multi-agent networks are developed on the main platform, enabling information flow, intelligent learning, and the real-time decision making of the included agents and elements. This abstraction framework for modeling the structural, behavioral, and social models of the agents can be defined using appropriate an AOP, which is a specialization of object-oriented programming [168]. These platforms initially were inspired by adaptive organizational models and later more structured techniques (e.g., JADE) were proposed in the software engineering domain [169]. Rapid advances in computation capabilities offered more customized models, such as O-MaSE, that utilized three concepts of the meta-model, method fragments, and guidelines based on method engineering concepts [170]. These platforms varied case by case and targeted either a specific application domain, such

as a microgrid [15], or flexible general-purpose platforms [171]. Some of the programming languages applied for implementing agent-oriented platforms include Java, C/C++, Python, AgentSpeak, NetLoGo, XML, and GAML [172]. The majority of these platforms are designed by adding reasoning and cognitive models, such as the procedural reasoning system (PRS) and/or belief–desire–intention (BDI) models [173]. PRSs help reasoning about processes, enabling agents to interact with the dynamic environment and use procedures for selecting intentions. These procedures are triggered when they can contribute to achieve certain goals. In a BDI model, which is the most common approach, the behavior of the agent is defined in terms of its beliefs, goals, and plans. In this model, the interpreter is responsible for updating these features based on the feedback received from the environment and the managing agents' intentions/or actions. These models showed great success in integrating AI as a pluggable component [174] or in meta-level plans [175]. The multi-agent-oriented programming (MAOP) platforms such as JaCaMo have a structured approach based on three concepts of agent, environment, and organization dimensions [176]. They also successfully integrated with the IoTs to offer self-adaptive applications in human-centric environments [177]. Systems of systems is another efficient methodology to develop meta-models to manage MASs with different subsystems [178]. During the last decades, tens of these approaches and updates for old versions were proposed. To choose the best option from the long list of the proposed AOP techniques and platforms, they can be compared based on basic platform properties, usability and scalability, stability and operating abilities, security management, and their applicability in practice [172]. They also need to be investigated and evaluated based on their architectural debt and their long-term effects on the health of a software system [179]. The evaluation frameworks for agent-oriented methodologies are reviewed in reference [180]. Although various AOPs were proposed in the literature, they still suffer strong reasoning and decision modules for modeling costs, preferences, time, resources, and durative actions, etc., [181]. As a result, developers are still reluctant to switch to these platforms and prefer to utilize the current programming language with small modifications of the main code [181].

4.2. Learning Mechanism

One of the main frameworks in adaptive multi-agent networks is their learning mechanism. Most of the recent learning frameworks are online, helping the system to learn the dynamics of their environment and the best responses to these changes. Various aspects such as knowledge-access level and learning technique were investigated for grouping the MAS learning literature [182]. Agents may have full autonomy to learn and share their knowledge with other agents [100] or may be restricted to only communicate and share their states with a central learner [183]. If learning is system-wise, one agent or the main ruler learns the policies for all agents in the system. In this case, the learner has full observability to discover the states of the involved agents without a detailed focus on the individual agent's actions. In this learning mechanism, information is collected from distributed agents and fed to the central learner of the system. This helps to achieve high-level information about system dynamics without getting trapped in the difficulties of coordinating information flow between multiple learners. Centralized learning and training can be integrated with either a central decision maker or decentralized excitation [184]. In the second category, the centralized learner learns the value function using the criteria for guiding distributed actors [185]. The centralized learning mechanisms suffer from problems such as the complexity of the state process and learning process. They were also developed based on some unrealistic assumptions, such as consistent and complete access to all agents' information. The other challenge of centralized learning is its vulnerability against the failures of the learner, a need for the high computation and memory resources in the central learner, and scalability for large-scale systems with thousands of distributed agents [186]. As a result of this learning, it may entail homogeneous team learning with one policy for all agents or heterogeneous learning with a unique behavior for each individual agent [187]. Other variations of this platform to alleviate the listed challenges are QMIX

with mixed global and local components [188]. Coordinated sampling by means of the maximum-entropy RL technique and policy distillation were other proposed solutions for improving centralized learning mechanisms [189]. There are also some hybrid techniques that agents learn individually but then share in a centralized common knowledge memory for sorting and storing their knowledge [190]. In distributed learning techniques, each agent is responsible for their own learning [191]. In these learning frameworks, the agents have limited observability and only explore their surrounding environments and they are unable to learn the overall dynamics of the systems in real-time. Most of these learning techniques are integrated with the decision-making process of the control mechanisms. The role-based learning technique is another trend in the literature in which the complex tasks are decomposed into different roles. The RODE technique is one of the examples for such learning that utilize agents clustering to discover roles and the required learning groups [90]. The learning and updating learned models can get initiated based on certain events or on discrete or continuous-time updates over the system's operation. Some of the common learning techniques applied in the literature include reinforcement learning, supervised learning, deep learning, game theory, probabilistic, swarm systems, applied logic, evolutionary algorithms, or a combination of some of them [192]. Since most of these techniques were applied for the simultaneous learning and control of the MASs, they are reviewed in the next section. Some of the proposed techniques for learning are initiated after receiving initial domain knowledge. In the literature, these techniques are identified as transfer learning methods [182].

4.3. Control Solutions

Control solutions can be investigated from aspects such as applications and techniques. There are some literature that surveyed existing control techniques for multi-agent systems [193]. They focused on interaction limitations and categorized them into sensing-based control, event-based control, pinning-based control, resilient control, and collaborative control. This sections reviews the applications and techniques of MASs control especially those that can be applied in controlling and managing smart cities.

4.3.1. MASs Applications

The multi-agent systems control is applied for a variety of domains including consensus and synchronization, leader-following coordination, formation control, containment and surrounding control, coverage, and distributed optimization and estimation [32,194]. Consensus and synchronization are the most common domains for the MAS literature. The main goal in this category is to reach an agreement in all agents and lead their states to a common state or time-varying reference [21]. This is achieved by monitoring and information exchanges between neighboring nodes. Therefore, consensus highly depends on the communication graphs between agents and their information sharing levels [195]. One of the main features of a smart city is consensus between smart service systems that interact and coordinate decisions in the system. For example, in a typical autonomous transportation method, multiple agents such as client agents, car agents, parking agents, route agents, and many other agents interact with each other while seeking the agreement and balance of interests [196]. Other examples for the application of MAS consensus are presented for smart parking that tests various negotiation strategies to reach agreement between the involved agents [197]. Applications of consensus in other platforms of a smart city such as in block chains [198] and smart grids [199], a smart factory [200], or other services provided by multi-robot rendezvous [201] are investigated in the literature. Consensus can be also applied for distributed computing where different processors need to reach the same estimation after iterative computations [21]. Leader–follower consensus can be considered as a special case of consensus in which the main goal is to minimize tracking error and state difference between leader and follower agents [24]. Examples of leader–follower control for smart city applications are presented for IoTs-based digital twins [202] and irrigation management systems [203]. This can be used in pinning control, in which

the leader reflects the desired trajectory of the system [41]. One of the best applications of pinning control is to restore complex cyberphysical networks in the smart city to their initial states after mixed attack strategies [204]. Group consensus is another variation in which agents with different task distributions converge to multiple consensus values [205]. Applications of this consensus for capturing the supportability of applications on smart city platforms and the IoTs [206], or recovery control with ideal structures [207], are also presented in the literature.

Formation control is another domain for applying MASs [19]. In this category of control framework, network evolution is guided to reach and maintain a desired geometric form by monitoring and controlling their absolute or relative distance from other agents [208]. In problems with monitoring agents' positions, depending on their interactions with other agents, they might follow displacement-based control [209] or distance-based control mechanisms [210]. The relative position of neighboring agents is measured with respect to a global coordinate system in displacement-based control, while the base is changed to the agent's local coordinate system in distance-based control systems [19]. Flocking is a special case of formation control in which the main goal is to keep all agents at an equal distance from their neighbors [211]. Applications of formation control in smart cities are presented in the literature for search and rescue, operations, intelligent highways, and mobile sensor networks [212]. Flocking is another techniques that can be used to solve robust problems in distributed environments [213]. In this technique, a large number of agents organize into a coordinated motion using three simple rules of cohesion, separation, and alignment [214]. One potential application for this technique can be forming a setting in which each agent is equally distanced from its neighbors [32]. This idea is applied to propose simple and scalable protocols for the migration of virtual machines (VMs) in IoTs cloud platforms [215].

Containment control is similar to distributed average tracking with multiple leaders. However, in this technique, the control protocol guides the followers and their state decisions to the convex hull formed by the leaders instead of leaders' state averages [216]. This can be helpful in smart cities when a failure or accident happens in certain areas. Containment control guides multi-agents such as autonomous robots to secure and remove undesirable outcomes while limiting their movements into other populated areas in the city [217]. In the surrounding control problems, the main goal is to protect a set of stationary or moving agents from possible threats surrounding them using other controlled agents [218]. One of the main applications for this technique is unmanned ground vehicles or unmanned surface vessels [219]. There are very few applications for containment and surrounding control in the literature and this area needs further investigation to find suitable applications in future connected smart cities [220].

One of the common goals in utilizing MASs is distributed optimization [49]. These techniques cover problems such as constrained, unconstrained, dynamic, and time-varying models. The applications for this domain of MAS control in smart cities are promising [221]. Examples of such applications are presented in energy [222,223], transportation [148,224], health care [225], and supply chains [226,227]. Task allocations in cloud computing platforms are considered as one of the main applications of distributed multi-agent optimization in smart cities [220]. Distributed estimation can be also reformulated as a distributed optimization problem in which the main goal is to reduce computation time and estimation error by splitting the task between multiple agents [228,229]. Some of the variations for this domain include distributed parameter estimation and distributed data regression using an average consensus algorithm, and a distributed Kalman filtering algorithm [32]. One of the main applications of distributed estimation in smart cities is monitoring the environmental state using deployed sensors in the system [142]. The state estimation of wireless power transfer systems in IoTs applications [230], and crowd sensing [231], are other examples of distributed estimation in smart cities. Simulating real-world systems can be also considered one of the initial applications of multi-agent systems [232].

4.3.2. Control Techniques

In the literature, various techniques were applied for controlling multi-agent systems, namely graph theory, game theory, control theory, and machine learning, which are considered the most commonly used techniques for this purpose [29,45]. There are also other algorithms such as optimization and bio-inspired algorithms, which were applied for the collective behavior of these systems [233].

Graph theory was applied for defining the structural controllability of multi-agent systems [234]. One of the most frequently used techniques in this domain is to use a graph Laplacian matrix to investigate MAS dynamics and convergence rates [67]. In the literature, consensus problems were also investigated using an edge Laplacian matrix by defining the edge state as a relative state difference for the edges [235]. The graph Laplacian spectrum has the second smallest and largest eigenvalues and their ratios play an important role in MASs control [236]. In recent literature, graph theory was mostly used for an initial analysis of the network dynamics, and other complementary approaches were applied to reach consensus states [237]. Applications of graph theory are not limited to using an Laplacian matrix and, in some cases, only adjacency and valency matrices were applied to investigate the network dynamic and its synchronization [238].

Game theory is another popular technique for modeling the dynamics and decision making of rationale collaborator agents in MASs [239]. In most of the distributed games, the main goal is to reach the Nash equilibrium while optimizing its own performance metric [240]. Markov games or stochastic games are examples of the initial applications of sequential multi-agent games that can be solved using dynamic programming (DP), Q-learning, or linear programming techniques [45]. Uncertainties in an agent's payoff and reward/utility can be also modeled using Bayesian–Stackelberg games [241]. Evolutionary games are other variations of game theory techniques applied for modeling the collective behavior of the agents, with bounded rationality repeatedly looking for equilibrium points [242]. The majority of the MAS problems were solved using control theory and its variations, such as adaptive control. For example, distributed model predictive control is widely applied for modeling different types of dynamics, with goals of regulation, tracking, or economic considerations in the system [64]. A survey for online learning control mechanisms in multi-agent systems was presented [194]. Neuro-adaptive optimal control is considered one of the most popular techniques in controlling complex MASs by solving an associated matrix of equations, such as the coupled Riccati equations or coupled Hamilton–Jacobi equations [243]. In the literature, other traditional optimization techniques such as variational inequality [106], duality theory [244], and the alternating direction method of multipliers (ADMM) [245] were applied for optimizing the operations of multi-agent systems.

Reinforcement learning is one of the well-known semi-supervised learning techniques applied in the simultaneous learning and control of adaptive MASs [246]. The multi-agent reinforcement learning (MARL) technique is usually applied for perceiving the environment based on partial information, such as rewards and penalties received as feedback from the previous actions and decisions [86]. Other techniques developed based on RL include Q-learning and policy gradient techniques that try to learn the optimal policy of the agents. The MARL techniques with networked agents are a special case of the MARL algorithms, in which agents can communicate with neighboring agents in a time-varying communication topology [247]. These algorithms were developed for both cooperative and non-cooperative settings. Using RL in MASs is very challenging because of the joint action space and dynamics generated with multiple autonomous decision makers that make the environment nonstationary and difficult to be perceived [248]. Using an ensemble of the policies is another technique proposed for designing control frameworks robust to environmental change and nonstationary dynamics [249]. Reference [188] proposed QMIX, which exploits a linear decomposition of the joint value function across agents while keeping the local and global maximum value functions monotonically over standard Q-learning. The other challenge is the high computation time for processing continuous

states and actions in MAS settings modeled by the Markov decision process, Markov game, or extensive form games [250]. In these settings, due to increasing number of state action pairs, it is very challenging to approximate value function or optimal policy. One of the main solutions for this problem is to apply deep reinforcement learning(DRL) that integrates deep neural networks in the learning process of RL iterations [251]. Two well-known variations of these techniques include deep Q-learning [252] and the deep deterministic policy gradient (DDPG), which are designed based on actor–critic networks with a replay buffer [253]. Policy gradient techniques such as DDPG perform better in MASs due to the independence of approximation from system dynamics. Other challenges of using RL-based control mechanisms in MASs include credit assignment problems that reflect a lack of tracing agents' actions and their influences on system outcomes [187]. This problem may result in the emergence of lazy inactive agents not willing to contribute to learning system dynamics. One of the proposed platforms for addressing this challenge is counterfactual multi-agent (COMA) policy gradients using a counterfactual baseline that keeps other agents' actions fixed while marginalizing the actions of the single agent [185]. An overview of the reviewed techniques is presented in Figure 5.

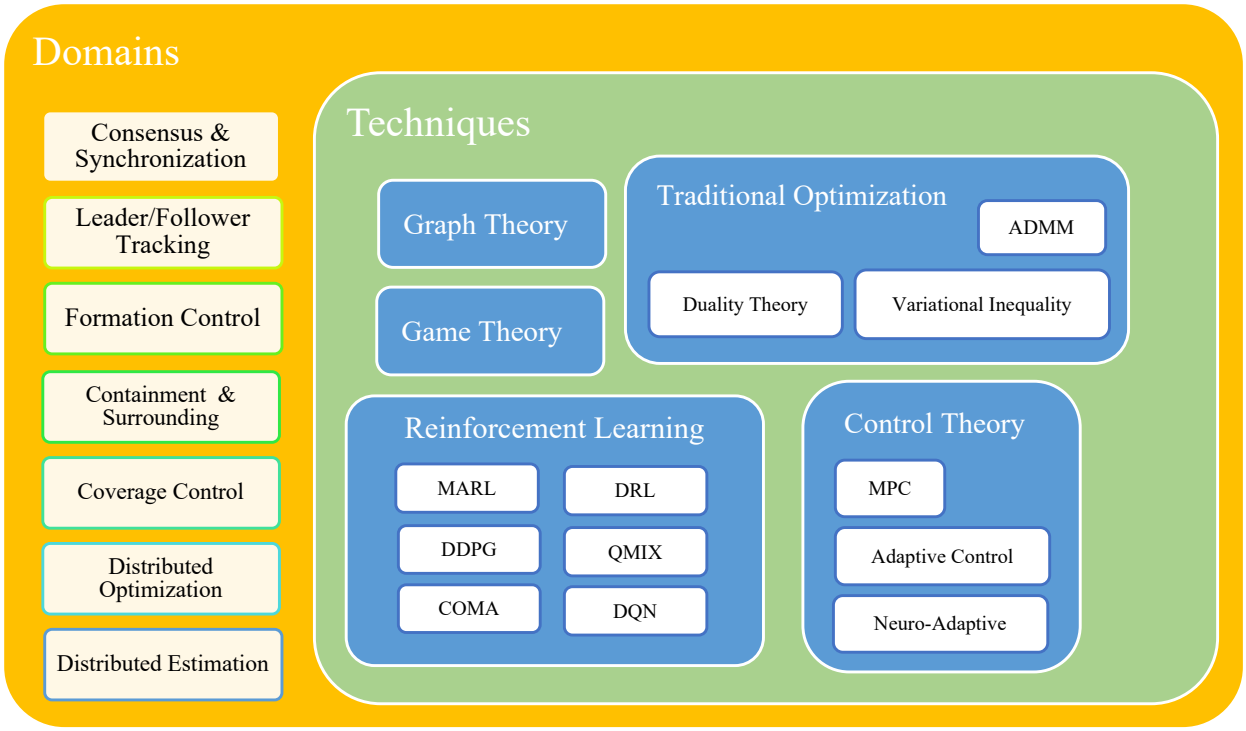

Figure 5. MAS control overview.

5. Evaluation Metrics

The next step in implementing adaptive MASs is their evaluation and validation. We focus on two main factors for this topic that include the main performance indicators applied for evaluating these systems and also existing test platforms and datasets applied for this purpose.

5.1. Performance Indicators

The performance of MASs is calculated based on various factors, such as convergence, stability, optimality, robustness, security, and other practical indicators, such as the quality of services, security, scalability, and bandwidth utilization [46]. The performance of MASs can also be measured based on more practical statistical and quantitative factors for outputs and resource utilization, such as throughput, response time, the number of concurrent agents/tasks, computational time, and communications overheads [254].

Convergence is applied in many MASs techniques and is considered one of the main performance criteria of the algorithm. The Nash equilibrium is a point and setting that all agents will prefer in which no agent can gain any more by changing only its own decisions. This point highly relies on its underlying assumptions, such as the rationality and reasoning capabilities of the agents [255]. Since convergence is not practical in real-world problems, researchers presented finite time convergence rules for controlling MASs. The main drawback of these rules is their dependency to initial states that makes them infeasible for cases with unknown initial states. Therefore, a fixed time stability rule is presented that works well with any arbitrary initial state of the agents. This type of analysis was tested for both time-triggered, and event-triggered systems and worked efficiently for both categories of the problems. These assumptions are violated in the bounded rationality and limitations of the mutual modeling of the agents. Moreover, it was shown that many of the value-based MARL algorithms do not converge to a stationary NE, and we may need to define cyclic equilibrium instead of unique states. To overcome this problem, some of the recent techniques used regret concepts instead of NE, which measure performance compared to the best static strategy of the agents.

Stability is one of the main indicators in evaluating the control mechanism of MASs. This measure shows whether the proposed scheme will deviate the convergence of the agents when facing future changes of the state and output of the system and distributions [256]. Lyapunov-based stability is known as one of the primary methods for testing the stability of MASs [257]. The Routh–Hurwitz stability criterion was also applied for a stability analysis of high-order consensus problems [258]. A system is also considered stable if after disturbance its solution and state are bounded in a certain region. It was proved that stability analysis for the cooperative control of heterogeneous agents with nonlinear dynamics is more challenging and needs error compensation controllers to eliminate error dynamics for the equilibrium point [259]. Barbalat's Lemma is an extension for Lyapunov analysis that overcomes the limitations of this technique in handling the stability of autonomous and time-varying nonlinear systems [260]. One of the measures to evaluate the proposed control mechanism is optimality, in which the solution is compared with the optimal solution of the centralized technique [64]. It is also important to quantify the optimality gap or bounds that the solution deviates from in its equivalent centralized problem solution [261]. Most of the MAS problems involve a nonconvex objective function with the local optimum point. To improve the efficiency of the search of the agents, some researchers defined conditions for the optimality of the consensus protocols [262].

Robustness is defined to reflect the control system's capability in handling future external unknown perturbations [263]. Determining the robustness of MASs with a large number of nodes is an NP-hard problem such that its calculations require more complex techniques, such as machine learning and neural networks [264]. There are specific considerations that can help to increase the robustness of these systems. For example, it was shown that existing feedback controls and responsibility declarations of the agents are very important in system robustness [265]. The robustness of the system can be investigated against perturbations of the coupling strengths [266], communication delays [267], and agents' dynamics [268] by introducing required formulations and protocols. These conditions for large open systems with a heterogeneous group of agents and unpredictable dynamics were investigated in reference [269]. Increasing the number of agents and decreasing the contribution of each agent on overall dynamics was considered an influencing factor in the robustness of the synchronization of heterogeneous MASs [270]. Communications strength and the nominal magnitude of the edge weights were identified as other important factors for the robustness of consensus networks [271]. The security of MASs and their control protocols need to be investigated for evaluating and adopting a suitable platform. The analysis of security against the physical faults and cyber-attacks of sensors and actuators, and surveys for recent advances, were summarized in reference [272]. The surveyed techniques investigated the security from the detection of the attack or fault and the techniques

and protocols proposed for secure and fault-tolerant control mechanisms. Another survey investigated the security from access control and trust models standpoints [273].

5.2. Test Datasets and Platforms

There are multiple test platforms and datasets, which are mainly used for comparing the efficiency of the proposed MASs control mechanisms. In machine learning, one of the common platforms for testing MARL and other MAS techniques is MiniGrid in Openai-Gym [274]. The TRACILOGIS platform is utilized for dynamic resource allocation and scheduling [275]. There are also two- and multiple-player games such as the MultiStep MNIST Game, FindGoal, RedBlueDoors, and StarCraft II that can be used for testing and comparing MAS techniques [90,128,276]. Other MAS control problems such as consensus and containment control are tested using simulation on small-scale numerical experiments with predefined equations for system dynamics and state changes, and their initial communication topologies or Laplacian matrices [24]. There are also datasets such as GraphGT from various domains for graph generation and transformation problems [277]. The SNAP data set [278] and the co-authorship network Hepth, communication network AS, and interaction user network Stov are other examples of graph data sets that can be utilized for testing and evaluating multi-agent network monitoring platforms [119].

6. Conclusions

This research surveyed the techniques proposed for designing and controlling adaptive multi-agent networked systems. These systems provide distributed frameworks for implementing IoTs systems comprising smart nodes and devices. Agents in these systems reflect the required levels of reactivity, autonomy, proactiveness, and social ability enabling the main system to resist possible external disturbances and internal failures. This is considered as one of the main requirements for smart cities. Future cities will no longer be considered as a set of disconnected systems, and they will change to interconnected networks with millions of smart agents. These agents and subsystems will constantly sense, monitor, plan, and communicate with each other to provide more adaptive, dynamic, efficient, and reliable services for future citizens. To achieve this goal, we require new perspectives, integrating existing advancements and variations in defining, monitoring, planning, and evaluating multi-agent systems. This research reviewed and summarized the recent advances to meet these requirements for the successful implementation of future smart cities.

Most of the proposed MAS techniques in the literature were developed as separate modules of the system for either designing and planning or evaluating these systems. Few researchers linked the proposed techniques to establish a solid integrated framework that is able to simultaneously monitor, adapt, control, and evaluate its performance. For example, big-data analytics techniques are not fully linked to control techniques and their potential applicability in reducing the computation steps of these search platforms is not fully investigated in the literature. Integrating more advanced state monitoring platforms such as clustering and pattern mining on the collected agents' data streams enables the designing of more customized frameworks for the various communities of the agents. Investigating the potential advantages of state monitoring techniques in designing noise- and fault-tolerant control mechanisms are other suggestions for future research in the MAS domain.

One of the other existing challenges that need to be addressed before using MASs for smart city platforms is heterogeneity of the involved agents in the network. A smart city should be able to efficiently control and monitor a wide range of heterogeneous systems with different entities, actions, and information flows. Most of the problems in the literature are simplified by defining initial assumptions on system structure, autonomy, and entity types and dynamics. More investigations and research are required for evaluating the proposed control mechanisms for heterogeneous systems, in which the agents not only have nonidentical definitions and duties but also differ in levels of uncertainty, disturbance, and noise.

The other missing aspect in the existing literature is considering the high level of environment uncertainty for systems exploring new platforms. The majority of the proposed techniques in this domain suffer from high computation times for iterative search to estimate system dynamics and best policy. Moreover, the recently presented MASs techniques in the machine learning domain neglected the power of information sharing and communication between agents, and the techniques still rely on a central processor for storing, processing, and decision making. The reviewed literature highlighted the need for revising these approaches to make them suitable for future distributed multi-agent systems that are able to establish wireless communications with other agents while exploring solutions for emerging complex situations, such as internal failure or cyber-attacks. The control mechanisms can be also improved by utilizing middle agents for the matching and abstracting of the signal-sharing process to reduce the computation time and increase their convergence rates. Most of the techniques were developed for static platforms and did not include potential changes in agents' locations, communications, qualifications, and reliability. Developing more general frameworks which are robust against episodic dynamics with predictable cycles and trends is another great idea for empowering adaptive multi-agent networks. For the successful implementation of these systems on the internet of things, it is necessary to constantly evaluate the systems and its changes to prevent potential security issues that infected agents may cause. The control mechanisms need to be designed in a more dynamic platform that can immediately react to changes in agents' availabilities, and thereby trust change. Designing and defining unique and general performance indicators and test frameworks can also provide a fair measurement for comparing existing techniques and determining their strengths and weaknesses in controlling multi-agent networks with predefined features and requirements.

Author Contributions: This paper represents a result of collegial teamwork. N.N. designed the research, conducted the literature reviews, and prepared the original draft of the manuscript. A.G. finalized and transformed the manuscript to meet MDPI draft, and submitted to the journal. 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.

Acknowledgments: The authors did not receive support from any organization for the submitted work.

Conflicts of Interest: The authors declare that there is no conflict of interest.

References

1. Cocchia, A. Smart and digital city: A systematic literature review. In *Smart City*; Springer: Cham, Switzerland, 2014; pp. 13–43.
2. Kim, T.; Ramos, C.; Mohammed, S. Smart city and IoT. *Future Gener. Comput. Syst.* **2017**, *76*, 159–162. [CrossRef]
3. Khan, J.Y.; Yuce, M.R. *Internet of Things (IoT): Systems and Applications*; CRC Press: Boca Raton, FL, USA, 2019.
4. Park, E.; Del Pobil, A.P.; Kwon, S.J. The role of Internet of Things (IoT) in smart cities: Technology roadmap-oriented approaches. *Sustainability* **2018**, *10*, 1388. [CrossRef]
5. Mukhopadhyay, S.C.; Suryadevara, N.K. Internet of things: Challenges and opportunities. In *Internet of Things*; Springer: Berlin/Heidelberg, Germany, 2014; Volume 9, pp. 1–17.
6. Iqbal, A.; Suryani, M.A.; Saleem, R.; Suryani, M.A. Internet of things (IoT): On-going security challenges and risks. *Int. J. Comput. Sci. Inf. Secur.* **2016**, *14*, 671.
7. Roscia, M.; Longo, M.; Lazaroiu, G.C. Smart City by multi-agent systems. In Proceedings of the 2013 International Conference on Renewable Energy Research and Applications (ICRERA), Madrid, Spain, 20–23 October 2013; pp. 371–376.
8. Dorri, A.; Kanhere, S.S.; Jurdak, R. Multi-agent systems: A survey. *IEEE Access* **2018**, *6*, 28573–28593. [CrossRef]
9. do Nascimento, N.M.; de Lucena, C.J.P. FIoT: An agent-based framework for self-adaptive and self-organizing applications based on the Internet of Things. *Inf. Sci.* **2017**, *378*, 161–176. [CrossRef]
10. Forestiero, A. Multi-agent recommendation system in Internet of Things. In Proceedings of the 2017 17th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGRID), Madrid, Spain, 14–17 May 2017; pp. 772–775.

11. Aseere, A.M. Multiagent Systems Applied to Smart City. *J. Eng. Appl. Sci.* **2020**, *7*, 29–36.
12. Longo, M.; Roscia, M.; Lazaroiu, C. Innovating multi-agent systems applied to smart city. *Res. J. Appl. Sci. Eng. Technol.* **2014**, *7*, 4296–4302. [CrossRef]
13. Krupitzer, C.; Roth, F.M.; VanSyckel, S.; Schiele, G.; Becker, C. A survey on engineering approaches for self-adaptive systems. *Pervasive Mob. Comput.* **2015**, *17*, 184–206. [CrossRef]
14. Boes, J.; Migeon, F. Self-organizing multi-agent systems for the control of complex systems. *J. Syst. Softw.* **2017**, *134*, 12–28. [CrossRef]
15. Kantamneni, A.; Brown, L.E.; Parker, G.; Weaver, W.W. Survey of multi-agent systems for microgrid control. *Eng. Appl. Artif. Intell.* **2015**, *45*, 192–203. [CrossRef]
16. Kaviani, S. Multi-Agent Clinical Decision Support Systems: A Survey. In Proceedings of the 1st Korea Artificial Intelligence Conference, Jeju Island, Korea, 23–25 April 2020
17. Dominguez, R.; Cannella, S. Insights on multi-agent systems applications for supply chain management. *Sustainability* **2020**, *12*, 1935. [CrossRef]
18. Gjikopulli, A.A.; Banerjee, A. A survey on Multi-Agent Systems (MAS). *Netw. Archit. Serv.* **2020**, 55–59.
19. Oh, K.K.; Park, M.C.; Ahn, H.S. A survey of multi-agent formation control. *Automatica* **2015**, *53*, 424–440. [CrossRef]
20. Ma, L.; Wang, Z.; Han, Q.L.; Liu, Y. Consensus control of stochastic multi-agent systems: A survey. *Sci. China Inf. Sci.* **2017**, *60*, 120201. [CrossRef]
21. Li, Y.; Tan, C. A survey of the consensus for multi-agent systems. *Syst. Sci. Control Eng.* **2019**, *7*, 468–482. [CrossRef]
22. Rahmani, A.; Ji, M.; Mesbahi, M.; Egerstedt, M. Controllability of multi-agent systems from a graph-theoretic perspective. *SIAM J. Control Optim.* **2009**, *48*, 162–186. [CrossRef]
23. Diaconescu, I.M.; Wagner, G. Modeling and simulation of web-of-things systems as multi-agent systems. In *German Conference on Multiagent System Technologies*; Springer: Cham, Switzerland, 2015; pp. 137–153.
24. Liang, H.; Liu, G.; Zhang, H.; Huang, T. Neural-Network-Based Event-Triggered Adaptive Control of Nonaffine Nonlinear Multiagent Systems With Dynamic Uncertainties. *IEEE Trans. Neural Netw. Learn. Syst.* **2021**, *32*, 2239–2250. [CrossRef]
25. Lowe, R.; Gupta, A.; Foerster, J.; Kiela, D.; Pineau, J. On the interaction between supervision and self-play in emergent communication. *arXiv* **2020**, arXiv:2002.01093.
26. Fouad, H.; Moskowitz, I.S. Meta-Agents: Using Multi-Agent Networks to Manage Dynamic Changes in the Internet of Things. In *Artificial Intelligence for the Internet of Everything*; Elsevier: Cambridge, MA, USA, 2019; pp. 271–281.
27. Xu, X.; Chen, S.; Huang, W.; Gao, L. Leader-following consensus of discrete-time multi-agent systems with observer-based protocols. *Neurocomputing* **2013**, *118*, 334–341. [CrossRef]
28. D'Angelo, M.; Gerasimou, S.; Ghahremani, S.; Grohmann, J.; Nunes, I.; Pournaras, E.; Tomforde, S. On learning in collective self-adaptive systems: State of practice and a 3d framework. In Proceedings of the 2019 IEEE/ACM 14th International Symposium on Software Engineering for Adaptive and Self-Managing Systems (SEAMS), Montreal, QC, Canada, 25 May 2019; pp. 13–24.
29. Zheng, Y.; Wang, L. Consensus of heterogeneous multi-agent systems without velocity measurements. *Int. J. Control* **2012**, *85*, 906–914. [CrossRef]
30. Gottifredi, S.; Tamargo, L.H.; García, A.J.; Simari, G.R. Arguing about informant credibility in open multi-agent systems. *Artif. Intell.* **2018**, *259*, 91–109. [CrossRef]
31. Kendrick, P.; Hussain, A.; Criado, N.; Randles, M. Multi-agent systems for scalable internet of things security. In Proceedings of the Second International Conference on Internet of things, Data and Cloud Computing, Cambridge, UK, 22–23 March 2017; pp. 1–6.
32. Chen, F.; Ren, W. On the control of multi-agent systems: A survey. *Found. Trends® Syst. Control* **2019**, *6*, 339–499. [CrossRef]
33. Zuo, Z.; Zhang, J.; Wang, Y. Adaptive fault-tolerant tracking control for linear and Lipschitz nonlinear multi-agent systems. *IEEE Trans. Ind. Electron.* **2014**, *62*, 3923–3931. [CrossRef]
34. Amirkhani, A.; Barshooi, A.H. Consensus in multi-agent systems: A review. *Artif. Intell. Rev.* **2021**, *2021*, 1–39. [CrossRef]
35. Jiang, C.; Du, H.; Zhu, W.; Yin, L.; Jin, X.; Wen, G. Synchronization of nonlinear networked agents under event-triggered control. *Inf. Sci.* **2018**, *459*, 317–326. [CrossRef]
36. Zhang, L.; Chen, B.; Lin, C.; Shang, Y. Fuzzy adaptive finite-time consensus tracking control for nonlinear multi-agent systems. *Int. J. Syst. Sci.* **2021**, *52*, 1346–1358. [CrossRef]
37. Wang, Z.; Xue, H.; Pan, Y.; Liang, H. Adaptive neural networks event-triggered fault-tolerant consensus control for a class of nonlinear multi-agent systems. *AIMS Math.* **2020**, *5*, 2780–2800. [CrossRef]
38. Guan, Y.; Ji, Z.; Zhang, L.; Wang, L. Controllability of heterogeneous multi-agent systems under directed and weighted topology. *Int. J. Control* **2016**, *89*, 1009–1024. [CrossRef]
39. Ren, W.; Beard, R.W. Consensus seeking in multiagent systems under dynamically changing interaction topologies. *IEEE Trans. Autom. Control* **2005**, *50*, 655–661. [CrossRef]
40. Desaraju, V.R.; How, J.P. Decentralized path planning for multi-agent teams with complex constraints. *Auton. Robot.* **2012**, *32*, 385–403. [CrossRef]
41. Movric, K.H.; Lewis, F.L. Cooperative optimal control for multi-agent systems on directed graph topologies. *IEEE Trans. Autom. Control* **2013**, *59*, 769–774. [CrossRef]

42. De Nijs, F. Resource-Constrained Multi-Agent Markov Decision Processes. Ph.D. Thesis, Delft University of Technology, Delft, The Netherlands, 2019.
43. Radulescu, R.; Mannion, P.; Roijers, D.M.; Nowé, A. *Recent Advances in Multi-Objective Multi-Agent Decision Making*; Benelux Association for Artificial Intelligence: Leiden, The Netherlands, 2020; pp. 392–394.
44. Lee, D.; Hu, J. Primal-dual distributed temporal difference learning. *arXiv* **2018**, arXiv:1805.07918.
45. Rizk, Y.; Awad, M.; Tunstel, E.W. Decision making in multiagent systems: A survey. *IEEE Trans. Cogn. Dev. Syst.* **2018**, *10*, 514–529. [CrossRef]
46. Rossi, F.; Bandyopadhyay, S.; Wolf, M.; Pavone, M. Review of multi-agent algorithms for collective behavior: A structural taxonomy. *IFAC-PapersOnLine* **2018**, *51*, 112–117. [CrossRef]
47. Aydin, M.E. Coordinating metaheuristic agents with swarm intelligence. *J. Intell. Manuf.* **2012**, *23*, 991–999. [CrossRef]
48. Zhu, L.; Xiang, Z. Aggregation analysis for competitive multiagent systems with saddle points via switching strategies. *IEEE Trans. Neural Netw. Learn. Syst.* **2017**, *29*, 2931–2943. [CrossRef]
49. Yang, T.; Yi, X.; Wu, J.; Yuan, Y.; Wu, D.; Meng, Z.; Hong, Y.; Wang, H.; Lin, Z.; Johansson, K.H. A survey of distributed optimization. *Annu. Rev. Control* **2019**, *47*, 278–305. [CrossRef]
50. Xiao, W.; Cao, L.; Li, H.; Lu, R. Observer-based adaptive consensus control for nonlinear multi-agent systems with time-delay. *Sci. China Inf. Sci.* **2020**, *63*, 1–17. [CrossRef]
51. Brown, R.; Rossi, F.; Solovey, K.; Tsao, M.; Wolf, M.T.; Pavone, M. On Local Computation for Network-Structured Convex Optimization in Multi-Agent Systems. *IEEE Trans. Control Netw. Syst.* **2021**, *8*, 542–554. [CrossRef]
52. Shen, Q.; Shi, P.; Zhu, J.; Wang, S.; Shi, Y. Neural Networks-Based Distributed Adaptive Control of Nonlinear Multiagent Systems. *IEEE Trans. Neural Netw. Learn. Syst.* **2020**, *31*, 1010–1021. [CrossRef]
53. Calvaresi, D.; Cid, Y.D.; Marinoni, M.; Dragoni, A.F.; Najjar, A.; Schumacher, M. Real-time multi-agent systems: Rationality, formal model, and empirical results. *Auton. Agents Multi-Agent Syst.* **2021**, *35*, 12. [CrossRef]
54. Eriksson, A.; Hansson, J. *Distributed Optimisation in Multi-Agent Systems Through Deep Reinforcement Learning*; TRITA-EECS-EX: Stockholm, Sweden, 2019.
55. Yu, H.; Shen, Z.; Leung, C.; Miao, C.; Lesser, V.R. A survey of multi-agent trust management systems. *IEEE Access* **2013**, *1*, 35–50.
56. Tariverdi, A.; Talebi, H.A.; Shafiee, M. Fault-tolerant consensus of nonlinear multi-agent systems with directed link failures, communication noise and actuator faults. *Int. J. Control* **2021**, *94*, 60–74. [CrossRef]
57. Calvaresi, D.; Marinoni, M.; Sturm, A.; Schumacher, M.; Buttazzo, G. The challenge of real-time multi-agent systems for enabling IoT and CPS. In Proceedings of the International Conference on Web Intelligence, Leipzig, Germany, 23–26 August 2017; pp. 356–364.
58. Cheng, L.; Hou, Z.G.; Tan, M.; Lin, Y.; Zhang, W. Neural-network-based adaptive leader-following control for multiagent systems with uncertainties. *IEEE Trans. Neural Netw.* **2010**, *21*, 1351–1358. [CrossRef] [PubMed]
59. Ding, L.; Han, Q.L.; Ge, X.; Zhang, X.M. An overview of recent advances in event-triggered consensus of multiagent systems. *IEEE Trans. Cybern.* **2017**, *48*, 1110–1123. [CrossRef]
60. Shen, B.; Wang, Z.; Liu, X. A Stochastic Sampled-Data Approach to Distributed $H\infty$ Filtering in Sensor Networks. *IEEE Trans. Circuits Syst. I Regul. Pap.* **2011**, *58*, 2237–2246. [CrossRef]
61. Heemels, W.H.; Donkers, M.; Teel, A.R. Periodic event-triggered control for linear systems. *IEEE Trans. Autom. Control* **2012**, *58*, 847–861. [CrossRef]
62. Ge, X.; Han, Q.L.; Ding, L.; Wang, Y.L.; Zhang, X.M. Dynamic event-triggered distributed coordination control and its applications: A survey of trends and techniques. *IEEE Trans. Syst. Man Cybern. Syst.* **2020**, *50*, 3112–3125. [CrossRef]
63. Zhang, X.M.; Han, Q.L.; Zhang, B.L. An overview and deep investigation on sampled-data-based event-triggered control and filtering for networked systems. *IEEE Trans. Ind. Inform.* **2016**, *13*, 4–16. [CrossRef]
64. Negenborn, R.; Maestre, J. Distributed Model Predictive Control: An overview of features and research opportunities. In Proceedings of the 11th IEEE International Conference on Networking, Sensing and Control, Miami, FL, USA, 7–9 April 2014.
65. Liu, P.; Xiao, F.; Wei, B.; Wang, A. Distributed constrained optimization problem of heterogeneous linear multi-agent systems with communication delays. *Syst. Control Lett.* **2021**, *155*, 105002. [CrossRef]
66. Chen, Z.; Li, Z.; Chen, C.P. Adaptive neural control of uncertain MIMO nonlinear systems with state and input constraints. *IEEE Trans. Neural Netw. Learn. Syst.* **2016**, *28*, 1318–1330. [CrossRef] [PubMed]
67. Wen, G.; Duan, Z.; Yu, W.; Chen, G. Consensus in multi-agent systems with communication constraints. *Int. J. Robust Nonlinear Control* **2012**, *22*, 170–182. [CrossRef]
68. Zhang, Y.; Liang, H.; Ma, H.; Zhou, Q.; Yu, Z. Distributed adaptive consensus tracking control for nonlinear multi-agent systems with state constraints. *Appl. Math. Comput.* **2018**, *326*, 16–32. [CrossRef]
69. Zhang, D.; Xu, Z.; Srinivasan, D.; Yu, L. Leader-follower consensus of multiagent systems with energy constraints: A Markovian system approach. *IEEE Trans. Syst. Man Cybern. Syst.* **2017**, *47*, 1727–1736. [CrossRef]
70. Marcotte, R.J.; Wang, X.; Mehta, D.; Olson, E. Optimizing multi-robot communication under bandwidth constraints. *Auton. Robot.* **2020**, *44*, 43–55. [CrossRef]
71. Ricci, A.; Piunti, M.; Viroli, M. Environment programming in multi-agent systems: An artifact-based perspective. *Auton. Agents Multi-Agent Syst.* **2011**, *23*, 158–192. [CrossRef]

72. Weyns, D.; Omicini, A.; Odell, J. Environment as a first class abstraction in multiagent systems. *Auton. Agents Multi-Agent Syst.* **2007**, *14*, 5–30. [CrossRef]
73. Platon, E.; Mamei, M.; Sabouret, N.; Honiden, S.; Parunak, H.V.D. Mechanisms for environments in multi-agent systems: Survey and opportunities. *Auton. Agents Multi-Agent Syst.* **2007**, *14*, 31–47. [CrossRef]
74. Johansson, K.; Rosolia, U.; Ubellacker, W.; Singletary, A.; Ames, A.D. Mixed Observable RRT: Multi-Agent Mission-Planning in Partially Observable Environments. *arXiv* **2021**, arXiv:2110.01002.
75. Bourne, R.A.; Excelente-Toledo, C.B.; Jennings, N.R. Run-time selection of coordination mechanisms in multi-agent systems. In *14th European Conference on Artificial Intelligence (ECAI-2000)*; IOS Press: Amsterdam, The Netherlands, 2000.
76. Chen, S.; Wang, M.; Li, Q. Second-order consensus of hybrid multi-agent systems with unknown disturbances via sliding mode control. *IEEE Access* **2020**, *8*, 34973–34980. [CrossRef]
77. Hernandez-Leal, P.; Kartal, B.; Taylor, M.E. A survey and critique of multiagent deep reinforcement learning. *Auton. Agents Multi-Agent Syst.* **2019**, *33*, 750–797. [CrossRef]
78. Wagner, G.; Choset, H. Path planning for multiple agents under uncertainty. In Proceedings of the Twenty-Seventh International Conference on Automated Planning and Scheduling, Pittsburgh, PA, USA, 18–23 June 2017.
79. Li, Z.; Duan, Z.; Xie, L.; Liu, X. Distributed robust control of linear multi-agent systems with parameter uncertainties. *Int. J. Control* **2012**, *85*, 1039–1050. [CrossRef]
80. Amato, C. Decision-Making Under Uncertainty in Multi-Agent and Multi-Robot Systems: Planning and Learning. In Proceedings of the IJCAI, Stockholm, Sweden, 13–19 July 2018; pp. 5662–5666.
81. Peng, Z.; Zhang, J.; Hu, J.; Huang, R.; Ghosh, B.K. Optimal containment control of continuous-time multi-agent systems with unknown disturbances using data-driven approach. *Sci. China Inf. Sci.* **2020**, *63*, 209205. [CrossRef]
82. Hu, G. Robust consensus tracking for an integrator-type multi-agent system with disturbances and unmodelled dynamics. *Int. J. Control* **2011**, *84*, 1–8. [CrossRef]
83. Khazaeni, Y.; Cassandras, C.G. Event-driven cooperative receding horizon control for multi-agent systems in uncertain environments. *IEEE Trans. Control Netw. Syst.* **2016**, *5*, 409–422. [CrossRef]
84. Zuo, Z.; Wang, C.; Ding, Z. Robust consensus control of uncertain multi-agent systems with input delay: A model reduction method. *Int. J. Robust Nonlinear Control* **2017**, *27*, 1874–1894. [CrossRef]
85. Zhou, Z.; Xu, H. Mean field game and decentralized intelligent adaptive pursuit evasion strategy for massive multi-agent system under uncertain environment. In Proceedings of the 2020 American Control Conference (ACC), Denver, CO, USA, 1–3 July 2020; pp. 5382–5387.
86. Busoniu, L.; Babuska, R.; De Schutter, B. A comprehensive survey of multiagent reinforcement learning. *IEEE Trans. Syst. Man Cybern. Part C (Appl. Rev.)* **2008**, *38*, 156–172. [CrossRef]
87. Khan, N. Learning to Cooperate Using Deep Reinforcement Learning in a Multi-Agent System. Ph.D. Thesis, University of Minnesota, Minneapolis, MN, USA, 2020.
88. Omidshafiei, S.; Pazis, J.; Amato, C.; How, J.P.; Vian, J. Deep decentralized multi-task multi-agent reinforcement learning under partial observability. In Proceedings of the International Conference on Machine Learning, PMLR, Sydney, Australia, 6–11 August 2017; pp. 2681–2690.
89. Shang, W.; Li, Q.; Qin, Z.; Yu, Y.; Meng, Y.; Ye, J. Partially observable environment estimation with uplift inference for reinforcement learning based recommendation. *Mach. Learn.* **2021**, *110*, 2603–2640. [CrossRef]
90. Wang, R.E.; Everett, M.; How, J.P. R-MADDPG for partially observable environments and limited communication. *arXiv* **2020**, arXiv:2002.06684.
91. Li, W.; Xie, L.; Zhang, J.F. Containment control of leader-following multi-agent systems with Markovian switching network topologies and measurement noises. *Automatica* **2015**, *51*, 263–267. [CrossRef]
92. Perez, J.; Silander, T. Non-markovian control with gated end-to-end memory policy networks. *arXiv* **2017**, arXiv:1705.10993.
93. Mansour, A.M. Cooperative Multi-Agent Vehicle-to-Vehicle Wireless Network in a Noisy Environment. *Int. J. Circuits, Syst. Signal Process.* **2021**, *15*, 135–148. [CrossRef]
94. Búrdalo, L.; Terrasa, A.; Julián, V.; García-Fornes, A. The information flow problem in multi-agent systems. *Eng. Appl. Artif. Intell.* **2018**, *70*, 130–141. [CrossRef]
95. Baki, B.; Bouzid, M.; Ligęza, A.; Mouaddib, A.I. A centralized planning technique with temporal constraints and uncertainty for multi-agent systems. *J. Exp. Theor. Artif. Intell.* **2006**, *18*, 331–364. [CrossRef]
96. Ge, M.; Bangui, H.; Buhnova, B. Big data for internet of things: A survey. *Future Gener. Comput. Syst.* **2018**, *87*, 601–614. [CrossRef]
97. Khan, A.; Zhang, C.; Lee, D.D.; Kumar, V.; Ribeiro, A. Scalable centralized deep multi-agent reinforcement learning via policy gradients. *arXiv* **2018**, arXiv:1805.08776.
98. Huang, D.; Jiang, H.; Yu, Z.; Hu, C.; Fan, X. Cluster-delay consensus in MASs with layered intermittent communication: A multi-tracking approach. *Nonlinear Dyn.* **2019**, *95*, 1713–1730. [CrossRef]
99. Ge, X.; Yang, F.; Han, Q.L. Distributed networked control systems: A brief overview. *Inf. Sci.* **2017**, *380*, 117–131. [CrossRef]
100. Sayed, A.H. Adaptive networks. *Proc. IEEE* **2014**, *102*, 460–497. [CrossRef]
101. Zhuge, H. Knowledge flow network planning and simulation. *Decis. Support Syst.* **2006**, *42*, 571–592. [CrossRef]

102. Zhang, C.; Lesser, V.R.; Abdallah, S. Self-organization for coordinating decentralized reinforcement learning. In Proceedings of the 9th International Conference on Autonomous Agents and Multiagent Systems: IFAAMAS, Richland, SC, USA, 9–13 May 2010; Volume 1, pp. 739–746.
103. Althnian, A.; Agah, A. Evolutionary learning of goal-oriented communication strategies in multi-agent systems. *J. Autom. Mob. Robot. Intell. Syst.* **2015**, *9*, 52–64. [CrossRef]
104. Zhang, T.; Zhu, Q. Informational design of dynamic multi-agent system. *arXiv* **2021**, arXiv:2105.03052.
105. Das, A.; Gervet, T.; Romoff, J.; Batra, D.; Parikh, D.; Rabbat, M.; Pineau, J. Tarmac: Targeted multi-agent communication. In Proceedings of the International Conference on Machine Learning, PMLR, Long Beach, CA, USA, 9–15 June 2019; pp. 1538–1546.
106. Wang, D.; Wang, Z.; Chen, M.; Wang, W. Distributed optimization for multi-agent systems with constraints set and communication time-delay over a directed graph. *Inf. Sci.* **2018**, *438*, 1–14. [CrossRef]
107. Jiang, X.; Xia, G.; Feng, Z. Output consensus of high-order linear multi-agent systems with time-varying delays. *IET Control Theory Appl.* **2019**, *13*, 1084–1094. [CrossRef]
108. Tan, X.; Cao, J.; Li, X.; Alsaedi, A. Leader-following mean square consensus of stochastic multi-agent systems with input delay via event-triggered control. *IET Control Theory Appl.* **2017**, *12*, 299–309. [CrossRef]
109. Han, F.; Wei, G.; Ding, D.; Song, Y. Local condition based consensus filtering with stochastic nonlinearities and multiple missing measurements. *IEEE Trans. Autom. Control* **2017**, *62*, 4784–4790. [CrossRef]
110. Cholvy, L.; da Costa Pereira, C. Usefulness of information for goal achievement. In *International Conference on Principles and Practice of Multi-Agent Systems*; Springer: Cham, Switzerland, 2019; pp. 123–137.
111. Djaidja, S.; Wu, Q.H.; Fang, H. Leader-following consensus of double-integrator multi-agent systems with noisy measurements. *Int. J. Control Autom. Syst.* **2015**, *13*, 17–24. [CrossRef]
112. Bacciu, D.; Micheli, A.; Podda, M. Edge-based sequential graph generation with recurrent neural networks. *Neurocomputing* **2020**, *416*, 177–189. [CrossRef]
113. Atluri, G.; Karpatne, A.; Kumar, V. Spatio-temporal data mining: A survey of problems and methods. *ACM Comput. Surv. (CSUR)* **2018**, *51*, 1–41. [CrossRef]
114. Wu, Z.; Pan, S.; Chen, F.; Long, G.; Zhang, C.; Philip, S.Y. A comprehensive survey on graph neural networks. *IEEE Trans. Neural Netw. Learn. Syst.* **2020**, *32*, 4–24. [CrossRef]
115. Laurence, E.; Doyon, N.; Dubé, L.J.; Desrosiers, P. Spectral dimension reduction of complex dynamical networks. *Phys. Rev. X* **2019**, *9*, 011042. [CrossRef]
116. Wang, D.; Cui, P.; Zhu, W. Structural deep network embedding. In Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, 13–17 August 2016; pp. 1225–1234.
117. Kipf, T.N.; Welling, M. Variational graph auto-encoders. *arXiv* **2016**, arXiv:1611.07308.
118. Hamilton, W.L.; Ying, R.; Leskovec, J. Inductive representation learning on large graphs. In Proceedings of the 31st International Conference on Neural Information Processing Systems, Long Beach, CA, USA, 4–9 December 2017; pp. 1025–1035.
119. Mahdavi, S.; Khoshraftar, S.; An, A. Dynamic joint variational graph autoencoders. *arXiv* **2019**, arXiv:1910.01963.
120. Barros, C.D.; Mendonça, M.R.; Vieira, A.B.; Ziviani, A. A Survey on Embedding Dynamic Graphs. *arXiv* **2021**, arXiv:2101.01229.
121. Sayama, H.; Laramee, C. Generative network automata: A generalized framework for modeling adaptive network dynamics using graph rewritings. In *Adaptive Networks*; Springer: Heidelberg, Germany, 2009; pp. 311–332.
122. Taheri, A.; Gimpel, K.; Berger-Wolf, T. *Learning Graph Representations with Recurrent Neural Network Autoencoders*; KDD Deep Learning Day: London, UK, 2018.
123. Papoudakis, G.; Albrecht, S.V. Variational autoencoders for opponent modeling in multi-agent systems. *arXiv* **2020**, arXiv:2001.10829.
124. Zhang, K.; Ying, H.; Dai, H.N.; Li, L.; Peng, Y.; Guo, K.; Yu, H. Compacting Deep Neural Networks for Internet of Things: Methods and Applications. *IEEE Internet Things J.* **2021**, *8*, 11935–11959. [CrossRef]
125. Lomuscio, A.; Qu, H.; Russo, F. Automatic data-abstraction in model checking multi-agent systems. In *International Workshop on Model Checking and Artificial Intelligence*; Springer: Heidelberg, Germany, 2010; pp. 52–68.
126. Rassam, M.A.; Zainal, A.; Maarof, M.A. An adaptive and efficient dimension reduction model for multivariate wireless sensor networks applications. *Appl. Soft Comput.* **2013**, *13*, 1978–1996. [CrossRef]
127. Grover, A.; Leskovec, J. node2vec: Scalable feature learning for networks. In Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, 13–17 August 2016; pp. 855–864.
128. Lin, T.; Huh, J.; Stauffer, C.; Lim, S.N.; Isola, P. Learning to Ground Multi-Agent Communication with Autoencoders. In Proceedings of the Advances in Neural Information Processing Systems, Virtual, 6–14 December 2021; Volume 34.
129. García, N.M. Multi-agent system for anomaly detection in Industry 4.0 using Machine Learning techniques. *ADCAIJ Adv. Distrib. Comput. Artif. Intell. J.* **2019**, *8*, 33–40.
130. Tahsien, S.M. A Neural Network Guided Genetic Algorithm for Flexible Flow Shop Scheduling Problem with Sequence Dependent Setup Time. Ph.D. Thesis, The University of Guelph, Guelph, ON, Canada, 2020.
131. Ma, X.; Wu, J.; Xue, S.; Yang, J.; Sheng, Q.Z.; Xiong, H. A Comprehensive Survey on Graph Anomaly Detection with Deep Learning. *arXiv* **2021**, arXiv:2106.07178.
132. Ding, K.; Li, J.; Liu, H. Interactive anomaly detection on attributed networks. In Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining, Melbourne, VIC, Australia, 11–15 February 2019; pp. 357–365.

133. Duan, D.; Tong, L.; Li, Y.; Lu, J.; Shi, L.; Zhang, C. AANE: Anomaly Aware Network Embedding For Anomalous Link Detection. In Proceedings of the 2020 IEEE International Conference on Data Mining (ICDM), Sorrento, Italy, 17–20 November 2020; pp. 1002–1007.
134. Zheng, M.; Zhou, C.; Wu, J.; Pan, S.; Shi, J.; Guo, L. Fraudne: A joint embedding approach for fraud detection. In Proceedings of the 2018 International Joint Conference on Neural Networks (IJCNN), Rio de Janeiro, Brazil, 8–13 July 2018; pp. 1–8.
135. Ruff, L.; Vandermeulen, R.; Goernitz, N.; Deecke, L.; Siddiqui, S.A.; Binder, A.; Müller, E.; Kloft, M. Deep one-class classification. In Proceedings of the International Conference on Machine Learning, PMLR, Stockholm, Sweden, 10–15 July 2018; pp. 4393–4402.
136. Louati, F.; Ktata, F.B. A deep learning-based multi-agent system for intrusion detection. *SN Appl. Sci.* **2020**, *2*, 1–13. [CrossRef]
137. Guo, X.; Zhao, L. A systematic survey on deep generative models for graph generation. *arXiv* **2020**, arXiv:2007.06686.
138. You, J.; Ying, R.; Ren, X.; Hamilton, W.; Leskovec, J. Graphrnn: Generating realistic graphs with deep auto-regressive models. In Proceedings of the International Conference on Machine Learning, PMLR, Stockholm, Sweden, 10–15 July 2018; pp. 5708–5717.
139. Zhou, D.; Zheng, L.; Han, J.; He, J. A data-driven graph generative model for temporal interaction networks. In Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, Virtual Event, CA, USA, 23–27 August 2020; pp. 401–411.
140. Peng, H.; Wang, H.; Du, B.; Bhuiyan, M.Z.A.; Ma, H.; Liu, J.; Wang, L.; Yang, Z.; Du, L.; Wang, S.; et al. Spatial temporal incidence dynamic graph neural networks for traffic flow forecasting. *Inf. Sci.* **2020**, *521*, 277–290. [CrossRef]
141. Li, L.; Yao, J.; Wenliang, L.; He, T.; Xiao, T.; Yan, J.; Wipf, D.; Zhang, Z. GRIN: Generative Relation and Intention Network for Multi-agent Trajectory Prediction. In Proceedings of the Advances in Neural Information Processing Systems, Virtual, 6–14 December 2021; Volume 34.
142. Guastella, D.; Camps, V.; Gleizes, M.P. Multi-agent Systems for Estimating Missing Information in Smart Cities. In *11th International Conference on Agents and Artificial Intelligence-ICAART 2019*; SCITEPRESS-Science and Technology Springer: Prague, Czech Republic, 2019; pp. 214–223.
143. Feng, W.; Zhang, C.; Zhang, W.; Han, J.; Wang, J.; Aggarwal, C.; Huang, J. STREAMCUBE: Hierarchical spatio-temporal hashtag clustering for event exploration over the Twitter stream. In Proceedings of the 2015 IEEE 31st International Conference on Data Engineering, Seoul, Korea, 13–17 April 2015; pp. 1561–1572.
144. Zhao, L.; Akoglu, L. On using classification datasets to evaluate graph outlier detection: Peculiar observations and new insights. *Big Data* **2021**, *2021*, 69. [CrossRef] [PubMed]
145. Teng, X.; Yan, M.; Ertugrul, A.M.; Lin, Y.R. Deep into hypersphere: Robust and unsupervised anomaly discovery in dynamic networks. In Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, Stockholm, Sweden, 13–19 July 2018.
146. Ierardi, C.; Orihuela, L.; Jurado, I. Distributed estimation techniques for cyber-physical systems: A systematic review. *Sensors* **2019**, *19*, 4720. [CrossRef]
147. Jing, G.; Zheng, Y.; Wang, L. Flocking of multi-agent systems with multiple groups. *Int. J. Control* **2014**, *87*, 2573–2582. [CrossRef]
148. Chen, K.; Wang, J.; Zhang, Y.; Lewis, F.L. Cluster consensus of heterogeneous linear multi-agent systems. *IET Control Theory Appl.* **2018**, *12*, 1533–1542. [CrossRef]
149. Belghache, E.; Georgé, J.P.; Gleizes, M.P. DREAM: Dynamic data relation extraction using adaptive multi-agent systems. In Proceedings of the 2017 Twelfth International Conference on Digital Information Management (ICDIM), Fukuoka, Japan, 12–14 September 2017; pp. 292–297.
150. Li, W.; Yang, J.Y. Comparing networks from a data analysis perspective. In *International Conference on Complex Sciences*; Springer: Heidelberg, Germany, 2009; pp. 1907–1916.
151. Kim, J.; Lee, J.G. Community detection in multi-layer graphs: A survey. *ACM SIGMOD Rec.* **2015**, *44*, 37–48. [CrossRef]
152. Banka, A.A.; Naaz, R. Large Scale Graph Analytics for Communities Using Graph Neural Networks. In *International Conference on Computational Data and Social Networks*; Springer: Cham, Switzerland, 2020; pp. 39–47.
153. Liu, F.; Xue, S.; Wu, J.; Zhou, C.; Hu, W.; Paris, C.; Nepal, S.; Yang, J.; Yu, P.S. Deep learning for community detection: Progress, challenges and opportunities. *arXiv* **2020**, arXiv:2005.08225.
154. Cazabet, R.; Rossetti, G.; Amblard, F. *Dynamic Community Detection*; Springer: New York, NY, USA, 2017.
155. Rossetti, G.; Cazabet, R. Community discovery in dynamic networks: A survey. *ACM Comput. Surv. (CSUR)* **2018**, *51*, 1–37. [CrossRef]
156. Chen, Z.; Liao, H.; Chu, T. Aggregation and splitting in self-driven swarms. *Phys. A Stat. Mech. Appl.* **2012**, *391*, 3988–3994. [CrossRef]
157. Ogston, E.; Overeinder, B.; Van Steen, M.; Brazier, F. A method for decentralized clustering in large multi-agent systems. In Proceedings of the Second International Joint Conference on Autonomous Agents and Multiagent Systems, Melbourne, VIC, Australia, 14–18 July 2003; pp. 789–796.
158. Cai, N.; Diao, C.; Khan, M.J. A novel clustering method based on quasi-consensus motions of dynamical multiagent systems. *Complexity* **2017**, *2017*, 4978613. [CrossRef]
159. Kadar, M.; Muntean, M.V.; Csabai, T. A Multi-agent System with Self-optimization for Automated Clustering (MASAC). In *Agents and Multi-Agent Systems: Technologies and Applications 2019*; Springer: Singapore, 2019; pp. 117–128.
160. Sequeira, P.; Antunes, C. Real-time sensory pattern mining for autonomous agents. In *International Workshop on Agents and Data Mining Interaction*; Springer: Heidelberg, Germany, 2010; pp. 71–83.

161. Fournier-Viger, P.; He, G.; Cheng, C.; Li, J.; Zhou, M.; Lin, J.C.W.; Yun, U. A survey of pattern mining in dynamic graphs. *Wiley Interdiscip. Rev. Data Min. Knowl. Discov.* **2020**, *10*, e1372. [CrossRef]
162. Halder, S.; Samiullah, M.; Lee, Y.K. Supergraph based periodic pattern mining in dynamic social networks. *Expert Syst. Appl.* **2017**, *72*, 430–442. [CrossRef]
163. Jin, R.; McCallen, S.; Almaas, E. Trend motif: A graph mining approach for analysis of dynamic complex networks. In Proceedings of the Seventh IEEE International Conference on Data Mining (ICDM 2007), Omaha, NE, USA, 28–31 October 2007; pp. 541–546.
164. Cheng, Z.; Flouvat, F.; Selmaoui-Folcher, N. Mining recurrent patterns in a dynamic attributed graph. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*; Springer: Cham, Switzerland, 2017; pp. 631–643.
165. Kaytoue, M.; Pitarch, Y.; Plantevit, M.; Robardet, C. Triggering patterns of topology changes in dynamic graphs. In Proceedings of the 2014 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM 2014), Beijing, China, 17–20 August 2014; pp. 158–165.
166. Fournier-Viger, P.; Cheng, C.; Cheng, Z.; Lin, J.C.W.; Selmaoui-Folcher, N. Mining significant trend sequences in dynamic attributed graphs. *Knowl.-Based Syst.* **2019**, *182*, 104797. [CrossRef]
167. Mahmoud, M.A.; Ahmad, M.S.; Mostafa, S.A. Norm-based behavior regulating technique for multi-agent in complex adaptive systems. *IEEE Access* **2019**, *7*, 126662–126678. [CrossRef]
168. Venkatesan, D. A Novel Agent-Based Enterprise Level System Development Technology. Ph.D. Thesis, Anna University, Tamil Nadu, India, 2018.
169. Bellifemine, F.; Bergenti, F.; Caire, G.; Poggi, A. JADE—A java agent development framework. In *Multi-Agent Programming*; Springer: Boston, MA, USA, 2005; pp. 125–147.
170. DeLoach, S.A.; Garcia-Ojeda, J.C. O-MaSE: A customisable approach to designing and building complex, adaptive multi-agent systems. *Int. J. Agent-Oriented Softw. Eng.* **2010**, *4*, 244–280. [CrossRef]
171. Cardoso, R.C.; Ferrando, A. A Review of Agent-Based Programming for Multi-Agent Systems. *Computers* **2021**, *10*, 16. [CrossRef]
172. Kravari, K.; Bassiliades, N. A survey of agent platforms. *J. Artif. Soc. Soc. Simul.* **2015**, *18*, 11. [CrossRef]
173. Bordini, R.H.; El Fallah Seghrouchni, A.; Hindriks, K.; Logan, B.; Ricci, A. Agent programming in the cognitive era. *Auton. Agents Multi-Agent Syst.* **2020**, *34*, 37. [CrossRef]
174. Costantini, S. ACE: A flexible environment for complex event processing in logical agents. In *International Workshop on Engineering Multi-Agent Systems*; Springer: Cham, Switzerland, 2015; pp. 70–91.
175. Araujo, P.; Rodríguez, S.; Hilaire, V. A metamodeling approach for the identification of organizational smells in multi-agent systems: Application to ASPECS. *Artif. Intell. Rev.* **2018**, *49*, 183–210. [CrossRef]
176. Boissier, O.; Bordini, R.H.; Hübner, J.F.; Ricci, A. Dimensions in programming multi-agent systems. *Knowl. Eng. Rev.* **2019**, *34*, e2. [CrossRef]
177. Rahimi, H.; Trentin, I.F.; Ramparany, F.; Boissier, O. SMASH: A Semantic-enabled Multi-agent Approach for Self-adaptation of Human-centered IoT. *arXiv* **2021**, arXiv:2105.14915.
178. Baek, Y.M.; Song, J.; Shin, Y.J.; Park, S.; Bae, D.H. A meta-model for representing system-of-systems ontologies. In Proceedings of the 2018 IEEE/ACM 6th International Workshop on Software Engineering for Systems-of-Systems (SESoS), Gothenburg, Sweden, 29 May 2018; pp. 1–7.
179. Pigazzini, I.; Briola, D.; Fontana, F.A. Architectural Technical Debt of Multiagent Systems Development Platforms. In Proceedings of the WOA 2021: Workshop "From Objects to Agents", Bologna, Italy, 1–3 September 2021.
180. Jazayeri, A.; Bass, E.J. Agent-Oriented Methodologies Evaluation Frameworks: A Review. *Int. J. Softw. Eng. Knowl. Eng.* **2020**, *30*, 1337–1370. [CrossRef]
181. Logan, B. An agent programming manifesto. *Int. J. Agent-Oriented Softw. Eng.* **2018**, *6*, 187–210. [CrossRef]
182. Da Silva, F.L.; Costa, A.H.R. A survey on transfer learning for multiagent reinforcement learning systems. *J. Artif. Intell. Res.* **2019**, *64*, 645–703. [CrossRef]
183. Dusparic, I.; Cahill, V. Autonomic multi-policy optimization in pervasive systems: Overview and evaluation. *ACM Trans. Auton. Adapt. Syst. (TAAS)* **2012**, *7*, 1–25. [CrossRef]
184. Sharma, P.K.; Fernandez, R.; Zaroukian, E.; Dorothy, M.; Basak, A.; Asher, D.E. Survey of recent multi-agent reinforcement learning algorithms utilizing centralized training. In *Artificial Intelligence and Machine Learning for Multi-Domain Operations Applications III*; International Society for Optics and Photonics: Orlando, FL, USA, 2021; Volume 11746, p. 117462K.
185. Foerster, J.; Farquhar, G.; Afouras, T.; Nardelli, N.; Whiteson, S. Counterfactual multi-agent policy gradients. In Proceedings of the AAAI Conference on Artificial Intelligence, New Orleans, LA, USA, 2–7 February 2018; Volume 32.
186. De Lemos, R.; Giese, H.; Müller, H.A.; Shaw, M.; Andersson, J.; Litoiu, M.; Schmerl, B.; Tamura, G.; Villegas, N.M.; Vogel, T.; et al. Software engineering for self-adaptive systems: A second research roadmap. In *Software Engineering for Self-Adaptive Systems II*; Springer: Heidelberg, Germany, 2013; pp. 1–32.
187. Panait, L.; Luke, S. Cooperative multi-agent learning: The state of the art. *Auton. Agents Multi-Agent Syst.* **2005**, *11*, 387–434. [CrossRef]
188. Rashid, T.; Samvelyan, M.; Schroeder, C.; Farquhar, G.; Foerster, J.; Whiteson, S. Qmix: Monotonic value function factorisation for deep multi-agent reinforcement learning. In Proceedings of the International Conference on Machine Learning, PMLR, Stockholm, Sweden, 10–15 July 2018; pp. 4295–4304.

189. Chen, G. A New Framework for Multi-Agent Reinforcement Learning–Centralized Training and Exploration with Decentralized Execution via Policy Distillation. *arXiv* **2019**, arXiv:1910.09152.
190. Pesce, E.; Montana, G. Improving coordination in small-scale multi-agent deep reinforcement learning through memory-driven communication. *Mach. Learn.* **2020**, *109*, 1727–1747. [CrossRef]
191. Czarnowski, I.; Jędrzejowicz, P. An agent-based framework for distributed learning. *Eng. Appl. Artif. Intell.* **2011**, *24*, 93–102. [CrossRef]
192. D'Angelo, M. Engineering Decentralized Learning in Self-Adaptive Systems. Ph.D. Thesis, Linnaeus University Press, Vaxjo, Sweden, 2021.
193. Shi, P.; Yan, B. A survey on intelligent control for multiagent systems. *IEEE Trans. Syst. Man Cybern. Syst.* **2020**, *51*, 161–175. [CrossRef]
194. Poveda, J.I.; Benosman, M.; Teel, A.R. Hybrid online learning control in networked multiagent systems: A survey. *Int. J. Adapt. Control Signal Process.* **2019**, *33*, 228–261. [CrossRef]
195. Tahbaz-Salehi, A.; Jadbabaie, A. A necessary and sufficient condition for consensus over random networks. *IEEE Trans. Autom. Control* **2008**, *53*, 791–795. [CrossRef]
196. Svítek, M.; Skobelev, P.; Kozhevnikov, S. Smart City 5.0 as an urban ecosystem of Smart services. In *International Workshop on Service Orientation in Holonic and Multi-Agent Manufacturing*; Springer: Cham, Switzerland, 2019; pp. 426–438.
197. Alves, B.R.; Alves, G.V.; Borges, A.P.; Leitão, P. Experimentation of negotiation protocols for consensus problems in smart parking systems. In *International Conference on Industrial Applications of Holonic and Multi-Agent Systems*; Springer: Cham, Switzerland, 2019; pp. 189–202.
198. Yu, H.; Yang, Z.; Sinnott, R.O. Decentralized big data auditing for smart city environments leveraging blockchain technology. *IEEE Access* **2018**, *7*, 6288–6296. [CrossRef]
199. Yang, S.; Tan, S.; Xu, J.X. Consensus based approach for economic dispatch problem in a smart grid. *IEEE Trans. Power Syst.* **2013**, *28*, 4416–4426. [CrossRef]
200. De Sousa, A.L.; De Oliveira, A.S. Distributed MAS with Leaderless Consensus to Job-Shop Scheduler in a Virtual Smart Factory with Modular Conveyors. In Proceedings of the 2020 Latin American Robotics Symposium (LARS), 2020 Brazilian Symposium on Robotics (SBR) and 2020 Workshop on Robotics in Education (WRE), Natal, Brazil, 9–13 November 2020; pp. 1–6.
201. Cardona, G.A.; Calderon, J.M. Robot swarm navigation and victim detection using rendezvous consensus in search and rescue operations. *Appl. Sci.* **2019**, *9*, 1702. [CrossRef]
202. Saad, A.; Faddel, S.; Youssef, T.; Mohammed, O.A. On the implementation of IoT-based digital twin for networked microgrids resiliency against cyber attacks. *IEEE Trans. Smart Grid* **2020**, *11*, 5138–5150. [CrossRef]
203. Lee, S.; Yang, Y.; Nayel, M.; Zhai, Y. Leader-follower irrigation system management with Shapley value. In *International Workshop on Automation, Control, and Communication Engineering (IWACCE 2021)*; SPIE: Beijing, China, 2021; Volume 11929, pp. 8–14.
204. Song, Z.; Liu, Y.; Tan, M. Robust pinning synchronization of complex cyberphysical networks under mixed attack strategies. *Int. J. Robust Nonlinear Control* **2019**, *29*, 1265–1278. [CrossRef]
205. Miao, G.; Ma, Q. Group consensus of the first-order multi-agent systems with nonlinear input constraints. *Neurocomputing* **2015**, *161*, 113–119. [CrossRef]
206. Yamakami, T. A dimensional framework to evaluate coverage of IoT services in city platform as a service. In Proceedings of the 2017 International Conference on Service Systems and Service Management, Dalian, China, 16–18 June 2017; pp. 1–5.
207. Etemadyrad, N.; Li, Q.; Zhao, L. Deep Graph Spectral Evolution Networks for Graph Topological Evolution. In Proceedings of the AAAI Conference on Artificial Intelligence, Vancouver, BC, Canada, 2–9 February 2021; Volume 35, pp. 7358–7366.
208. Ren, W.; Beard, R.W. Formation feedback control for multiple spacecraft via virtual structures. *IEE Proc. Control Theory Appl.* **2004**, *151*, 357–368. [CrossRef]
209. Cortés, J. Global and robust formation-shape stabilization of relative sensing networks. *Automatica* **2009**, *45*, 2754–2762. [CrossRef]
210. Krick, L.; Broucke, M.E.; Francis, B.A. Stabilisation of infinitesimally rigid formations of multi-robot networks. *Int. J. Control* **2009**, *82*, 423–439. [CrossRef]
211. Olfati-Saber, R.; Jalalkamali, P. Collaborative target tracking using distributed Kalman filtering on mobile sensor networks. In Proceedings of the 2011 American Control Conference, San Francisco, CA, USA, 29 June–1 July 2011; pp. 1100–1105.
212. Wang, H.; Shi, D.; Song, B. A dynamic role assignment formation control algorithm based on hungarian method. In Proceedings of the 2018 IEEE SmartWorld, Ubiquitous Intelligence & Computing, Advanced & Trusted Computing, Scalable Computing & Communications, Cloud & Big Data Computing, Internet of People and Smart City Innovation (SmartWorld/SCALCOM/UIC/ATC/CBDCom/IOP/SCI), Guangzhou, China, 8–12 October 2018; pp. 687–696.
213. Barve, A.; Nene, M.J. Survey of Flocking Algorithms in multi-agent Systems. *Int. J. Comput. Sci. Issues (IJCSI)* **2013**, *10*, 110–117.
214. Olfati-Saber, R. Flocking for multi-agent dynamic systems: Algorithms and theory. *IEEE Trans. Autom. Control* **2006**, *51*, 401–420. [CrossRef]
215. Abdelwahab, S.; Hamdaoui, B. Flocking virtual machines in quest for responsive iot cloud services. In Proceedings of the 2017 IEEE International Conference on Communications (ICC), Paris, France, 21–25 May 2017; pp. 1–6.
216. Haghshenas, H.; Badamchizadeh, M.A.; Baradarannia, M. Containment control of heterogeneous linear multi-agent systems. *Automatica* **2015**, *54*, 210–216. [CrossRef]

217. Ji, M.; Ferrari-Trecate, G.; Egerstedt, M.; Buffa, A. Containment control in mobile networks. *IEEE Trans. Autom. Control* **2008**, *53*, 1972–1975. [CrossRef]
218. Xu, B.; Zhang, H.T.; Meng, H.; Hu, B.; Chen, D.; Chen, G. Moving target surrounding control of linear multiagent systems with input saturation. *IEEE Trans. Syst. Man Cybern. Syst.* **2020**, *52*, 1705–1715. [CrossRef]
219. Hu, B.B.; Zhang, H.T.; Liu, B.; Meng, H.; Chen, G. Distributed Surrounding Control of Multiple Unmanned Surface Vessels With Varying Interconnection Topologies. *IEEE Trans. Control Syst. Technol.* **2021**, *30*, 400–407. [CrossRef]
220. Li, M.; Wang, Z.; Li, K.; Liao, X.; Hone, K.; Liu, X. Task Allocation on Layered Multiagent Systems: When Evolutionary Many-Objective Optimization Meets Deep Q-Learning. *IEEE Trans. Evol. Comput.* **2021**, *25*, 842–855. [CrossRef]
221. Amini, M.H.; Mohammadi, J.; Kar, S. Promises of fully distributed optimization for iot-based smart city infrastructures. In *Optimization, Learning, and Control for Interdependent Complex Networks*; Springer: Cham, Switzerland, 2020; pp. 15–35.
222. Raju, L.; Sankar, S.; Milton, R. Distributed optimization of solar micro-grid using multi agent reinforcement learning. *Procedia Comput. Sci.* **2015**, *46*, 231–239. [CrossRef]
223. Mohamed, M.A.; Jin, T.; Su, W. Multi-agent energy management of smart islands using primal-dual method of multipliers. *Energy* **2020**, *208*, 118306. [CrossRef]
224. Olszewski, R.; Pałka, P.; Turek, A. Solving "Smart City" Transport Problems by Designing Carpooling Gamification Schemes with Multi-Agent Systems: The Case of the So-Called "Mordor of Warsaw". *Sensors* **2018**, *18*, 141. [CrossRef]
225. Euchi, J.; Zidi, S.; Laouamer, L. A new distributed optimization approach for home healthcare routing and scheduling problem. *Decis. Sci. Lett.* **2021**, *10*, 217–230. [CrossRef]
226. Lin, F.r.; Kuo, H.c.; Lin, S.m. The enhancement of solving the distributed constraint satisfaction problem for cooperative supply chains using multi-agent systems. *Decis. Support Syst.* **2008**, *45*, 795–810. [CrossRef]
227. Hsieh, F.S. Dynamic configuration and collaborative scheduling in supply chains based on scalable multi-agent architecture. *J. Ind. Eng. Int.* **2019**, *15*, 249–269. [CrossRef]
228. Liu, Q.; Yang, S.; Wang, J. A collective neurodynamic approach to distributed constrained optimization. *IEEE Trans. Neural Netw. Learn. Syst.* **2016**, *28*, 1747–1758. [CrossRef]
229. Necoara, I.; Nedelcu, V.; Dumitrache, I. Parallel and distributed optimization methods for estimation and control in networks. *J. Process Control* **2011**, *21*, 756–766. [CrossRef]
230. Rana, M.M.; Abdelhadi, A.; Shireen, W. Monitoring Operating Conditions of Wireless Power Transfer Systems Using Distributed Estimation Process. In Proceedings of the 2021 IEEE International Conference on Communications Workshops (ICC Workshops), Montreal, QC, Canada, 14–23 June 2021; pp. 1–4.
231. Guastella, D.A.; Campss, V.; Gleizes, M.P. A Cooperative Multi-Agent System for Crowd Sensing Based Estimation in Smart Cities. *IEEE Access* **2020**, *8*, 183051–183070. [CrossRef]
232. Tan, R.K.; Bora, Ş. Exploiting of Adaptive Multi Agent System Theory in Modeling and Simulation: A Survey. *J. Appl. Math. Comput. (JAMC)* **2018**, *1*, 21–26. [CrossRef]
233. Rossi, F.; Bandyopadhyay, S.; Wolf, M.T.; Pavone, M. Multi-Agent Algorithms for Collective Behavior: A structural and application-focused atlas. *arXiv* **2021**, arXiv:2103.11067.
234. Chen, F.; Ren, W. Multi-Agent Control: A Graph-Theoretic Perspective. *J. Syst. Sci. Complex.* **2021**, *34*, 1973–2002. [CrossRef]
235. Zelazo, D.; Rahmani, A.; Mesbahi, M. Agreement via the edge laplacian. In Proceedings of the 2007 46th IEEE Conference on Decision and Control, New Orleans, LA, USA, 12–14 December 2007; pp. 2309–2314.
236. You, K.; Xie, L. Network topology and communication data rate for consensusability of discrete-time multi-agent systems. *IEEE Trans. Autom. Control* **2011**, *56*, 2262–2275. [CrossRef]
237. Shi, C.X.; Yang, G.H. Robust consensus control for a class of multi-agent systems via distributed PID algorithm and weighted edge dynamics. *Appl. Math. Comput.* **2018**, *316*, 73–88. [CrossRef]
238. Wang, Q.; Wang, Y. Cluster synchronization of a class of multi-agent systems with a bipartite graph topology. *Sci. China Inf. Sci.* **2014**, *57*, 1–11. [CrossRef]
239. Shoham, Y.; Leyton-Brown, K. *Multiagent Systems: Algorithmic, Game-Theoretic, and Logical Foundations*; Cambridge University Press: Cambridge, UK, 2008.
240. Zhu, M.; Martínez, S. Distributed coverage games for energy-aware mobile sensor networks. *SIAM J. Control Optim.* **2013**, *51*, 1–27. [CrossRef]
241. Sengupta, S.; Kambhampati, S. Multi-agent reinforcement learning in bayesian stackelberg markov games for adaptive moving target defense. *arXiv* **2020**, arXiv:2007.10457.
242. Sun, C.; Wang, X.; Liu, J. Evolutionary game theoretic approach for optimal resource allocation in multi-agent systems. In Proceedings of the 2017 Chinese Automation Congress (CAC), Jinan, China, 20–22 October 2017; pp. 5588–5592.
243. Yan, B.; Shi, P.; Lim, C.C.; Shi, Z. Optimal robust formation control for heterogeneous multi-agent systems based on reinforcement learning. *Int. J. Robust Nonlinear Control* **2021**, *32*, 2683–2704. [CrossRef]
244. Wai, H.T.; Yang, Z.; Wang, Z.; Hong, M. Multi-agent reinforcement learning via double averaging primal-dual optimization. *arXiv* **2018**, arXiv:1806.00877.
245. Jian, L.; Zhao, Y.; Hu, J.; Li, P. Distributed inexact consensus-based ADMM method for multi-agent unconstrained optimization problem. *IEEE Access* **2019**, *7*, 79311–79319. [CrossRef]
246. Kapoor, S. Multi-agent reinforcement learning: A report on challenges and approaches. *arXiv* **2018**, arXiv:1807.09427.

247. Zhang, K.; Yang, Z.; Başar, T. Decentralized multi-agent reinforcement learning with networked agents: Recent advances. *arXiv* **2019**, arXiv:1912.03821.
248. Papoudakis, G.; Christianos, F.; Rahman, A.; Albrecht, S.V. Dealing with non-stationarity in multi-agent deep reinforcement learning. *arXiv* **2019**, arXiv:1906.04737.
249. Lowe, R.; Wu, Y.; Tamar, A.; Harb, J.; Abbeel, P.; Mordatch, I. Multi-agent actor-critic for mixed cooperative-competitive environments. *arXiv* **2017**, arXiv:1706.02275.
250. 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: Cham, Switzerland, 2021; pp. 321–384.
251. Du, W.; Ding, S. A survey on multi-agent deep reinforcement learning: From the perspective of challenges and applications. *Artif. Intell. Rev.* **2021**, *54*, 3215–3238. [CrossRef]
252. Mnih, V.; Badia, A.P.; Mirza, M.; Graves, A.; Lillicrap, T.; Harley, T.; Silver, D.; Kavukcuoglu, K. Asynchronous methods for deep reinforcement learning. In Proceedings of the International Conference on Machine Learning, PMLR, New York, NY, USA, 20–22 June 2016; pp. 1928–1937.
253. Lillicrap, T.P.; Hunt, J.J.; Pritzel, A.; Heess, N.; Erez, T.; Tassa, Y.; Silver, D.; Wierstra, D. Continuous control with deep reinforcement learning. *arXiv* **2015**, arXiv:1509.02971.
254. Lee, L.C.; Nwana, H.S.; Ndumu, D.T.; De Wilde, P. The stability, scalability and performance of multi-agent systems. *BT Technol. J.* **1998**, *16*, 94–103. [CrossRef]
255. Liu, J.; Zhang, Y.; Sun, C.; Yu, Y. Fixed-time consensus of multi-agent systems with input delay and uncertain disturbances via event-triggered control. *Inf. Sci.* **2019**, *480*, 261–272. [CrossRef]
256. Chli, M.; De Wilde, P.; Goossenaerts, J.; Abramov, V.; Szirbik, N.; Correia, L.; Mariano, P.; Ribeiro, R. Stability of multi-agent systems. In Proceedings of the SMC'03 Conference Proceedings, 2003 IEEE International Conference on Systems, Man and Cybernetics. Conference Theme-System Security and Assurance (Cat. No. 03CH37483), Washington, DC, USA, 8 October 2003; Volume 1, pp. 551–556.
257. Maadani, M.; Butcher, E.A. Consensus stability in multi-agent systems with periodically switched communication topology using Floquet theory. *Trans. Inst. Meas. Control* **2021**, *43*, 1239–1254. [CrossRef]
258. Miao, G.; Xu, S.; Zou, Y. Consentability for high-order multi-agent systems under noise environment and time delays. *J. Frankl. Inst.* **2013**, *350*, 244–257. [CrossRef]
259. Wang, B.; Fang, X.; Zhao, Y. Stability Analysis of Cooperative Control for Heterogeneous Multi-agent Systems with Nonlinear Dynamics. In Proceedings of the 2019 IEEE International Conference on Industrial Technology (ICIT), Melbourne, VIC, Australia, 13–15 February 2019; pp. 1446–1453.
260. Liu, Y.; Min, H.; Wang, S.; Liu, Z.; Liao, S. Distributed consensus of a class of networked heterogeneous multi-agent systems. *J. Frankl. Inst.* **2014**, *351*, 1700–1716. [CrossRef]
261. Sun, X.; Cassandras, C.G.; Meng, X. Exploiting submodularity to quantify near-optimality in multi-agent coverage problems. *Automatica* **2019**, *100*, 349–359. [CrossRef]
262. Katsuura, H.; Fujisaki, Y. Optimality of consensus protocols for multi-agent systems with interaction. In Proceedings of the 2014 IEEE International Symposium on Intelligent Control (ISIC), Juan Les Pins, France, 8–10 October 2014; pp. 282–285.
263. Yang, X.; Wang, J.; Tan, Y. Robustness analysis of leader–follower consensus for multi-agent systems characterized by double integrators. *Syst. Control Lett.* **2012**, *61*, 1103–1115. [CrossRef]
264. Wang, G.; Xu, M.; Wu, Y.; Zheng, N.; Xu, J.; Qiao, T. Using machine learning for determining network robustness of multi-agent systems under attacks. In *Pacific Rim International Conference on Artificial Intelligence*; Springer: Cham, Switzerland, 2018; pp. 491–498.
265. Baldoni, M.; Baroglio, C.; Micalizio, R. Fragility and Robustness in Multiagent Systems. In *International Workshop on Engineering Multi-Agent Systems*; Springer: Cham, Switzerland 2020; pp. 61–77.
266. Tian, Y.P.; Liu, C.L. Robust consensus of multi-agent systems with diverse input delays and asymmetric interconnection perturbations. *Automatica* **2009**, *45*, 1347–1353. [CrossRef]
267. Münz, U.; Papachristodoulou, A.; Allgöwer, F. Delay robustness in consensus problems. *Automatica* **2010**, *46*, 1252–1265. [CrossRef]
268. Trentelman, H.L.; Takaba, K.; Monshizadeh, N. Robust synchronization of uncertain linear multi-agent systems. *IEEE Trans. Autom. Control* **2013**, *58*, 1511–1523. [CrossRef]
269. Minsky, N.H.; Murata, T. On manageability and robustness of open multi-agent systems. In *International Workshop on Software Engineering for Large-Scale Multi-Agent Systems*; Springer: Heidelberg, Germany, 2003; pp. 189–206.
270. Kim, J.; Yang, J.; Shim, H.; Kim, J.S. Robustness of synchronization in heterogeneous multi-agent systems. In Proceedings of the 2013 European Control Conference (ECC), Zurich, Switzerland, 17–19 July 2013; pp. 3821–3826.
271. Zelazo, D.; Bürger, M. On the robustness of uncertain consensus networks. *IEEE Trans. Control Netw. Syst.* **2015**, *4*, 170–178. [CrossRef]
272. Zhang, D.; Feng, G.; Shi, Y.; Srinivasan, D. Physical safety and cyber security analysis of multi-agent systems: A survey of recent advances. *IEEE/CAA J. Autom. Sin.* **2021**, *8*, 319–333. [CrossRef]
273. Jung, Y.; Kim, M.; Masoumzadeh, A.; Joshi, J.B. A survey of security issue in multi-agent systems. *Artif. Intell. Rev.* **2012**, *37*, 239–260. [CrossRef]

274. Chevalier-Boisvert, M.; Willems, L.; Pal, S. Minimalistic Gridworld Environment for OpenAI Gym. 2018. Available online: https://github.com/maximecb/gym-minigrid (accessed on 10 January 2022).
275. Mezgebe, T.T.; Demesure, G.; El Haouzi, H.B.; Pannequin, R.; Thomas, A. CoMM: A consensus algorithm for multi-agent-based manufacturing system to deal with perturbation. *Int. J. Adv. Manuf. Technol.* **2019**, *105*, 3911–3926. [CrossRef]
276. Foerster, J.N.; Assael, Y.M.; De Freitas, N.; Whiteson, S. Learning to communicate with deep multi-agent reinforcement learning. *arXiv* **2016**, arXiv:1605.06676.
277. Du, Y.; Wang, S.; Guo, X.; Cao, H.; Hu, S.; Jiang, J.; Varala, A.; Angirekula, A.; Zhao, L. GraphGT: Machine Learning Datasets for Graph Generation and Transformation. In Proceedings of the Thirty-Fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2), Virtual, 6–14 December 2021.
278. Leskovec, J.; Krevl, A. *SNAP Datasets: Stanford Large Network Dataset Collection*; SNAP: Santa Monica, CA, USA, 2014.