Chapter 1
Multiagent Systems Under Uncertainty

The impact of the advent of the computer on modern societies can hardly be overstated; every single day we are surrounded by more devices equipped with on-board computation capabilities taking care of the ever-expanding range of functions they perform for us. Moreover, the decreasing cost and increasing sophistication of hardware and software opens up the possibility of deploying a large number of devices or systems to solve real-world problems. Each one of these systems (e.g., computer, router, robot, person) can be thought of as an *agent* which receives information and makes decisions about how to act in the world. As the number and sophistication of these agents increase, controlling them in such a way that they consider and cooperate with each other becomes critical. In many of these *multiagent systems (MASs)*, cooperation is made more difficult by the fact that the environment is unpredictable and the information available about the world and other agents (through sensors and communication channels) is noisy and imperfect. Developing agent controllers by hand becomes very difficult in these complex domains, so automated methods for generating solutions from a domain specification are needed. In this book, we describe a formal framework, called the *decentralized partially observable Markov decision process (Dec-POMDP),* that can be used for decision making for a team of cooperative agents. Solutions to Dec-POMDPs optimize the behavior of the agents while considering the uncertainty related to the environment and other agents. As discussed below, the Dec-POMDP model is very general and applies to a wide range of applications.

From a historical perspective, thinking about interaction has been part of many different 'fields' or, simply, aspects of life, such as philosophy, politics and war. Mathematical analyses of problems of interaction date back to at least the beginning of the eighteenth century [Bellhouse, 2007], driven by interest in games such as chess [Zermelo, 1913]. This culminated in the formalization of game theory with huge implications for the field of economics since the 1940s [von Neumann and Morgenstern, 1944]. Other single-step cooperative team models were studied by Marschak [1955] and Radner [1962], followed by systems with dynamics modeled as team theory problems [Marschak and Radner, 1972, Ho, 1980] and the resulting complexity analysis [Papadimitriou and Tsitsiklis, 1987]. In the 1980s, people

F.A. Oliehoek and C. Amato, *A Concise Introduction to Decentralized POMDPs*,
SpringerBriefs in Intelligent Systems, DOI 10.1007/978-3-319-28929-8_1

in the field of Artificial Intelligence took their concept of *agent*, an artificial entity that interacts with its environment over a sequence of time steps, and started thinking about multiple such agents and how they could interact [Davis and Smith, 1983, Grosz and Sidner, 1986, Durfee et al., 1987, Wooldridge and Jennings, 1995, Tambe, 1997, Jennings, 1999, Lesser, 1999]. Dec-POMDPs represent a probabilistic generalization of this multiagent framework to model uncertainty with respect to outcomes, environmental information and communication. We first discuss some motivating examples for the Dec-POMDP model and then provide additional details about multiagent systems, the uncertainty considered in Dec-POMDPs and application domains.

1.1 Motivating Examples

Before diving deeper, this section will present two motivating examples for the models and techniques described in this book. The examples briefly illustrate the difficulties and uncertainties one has to deal with when automating decisions in real-world decentralized systems. Several other examples and applications are discussed in Section 1.4.

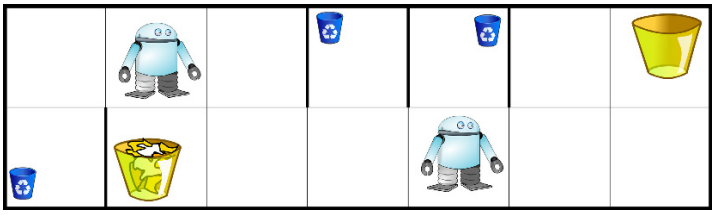

Fig. 1.1: Illustration of a simple Recycling Robots example, in which two robots remove trash in an office environment with three small (blue) trash cans and two large (yellow) ones. In this situation, the left robot may observe that the large trash can next to it is full, and the other robot may detect that the adjacent small trash can is empty. Note that neither robot can be sure of the trash can's true state due to limited sensing capabilities, nor do the robots see the state of trash cans further away. Also, the robots cannot observe each other at this distance and they do not know the observations of the other robot due to a lack of communication.

Multirobot Coordination: Recycling Robots Consider a team of robots tasked with removing trash from an office building, depicted in Figure 1.1. The robots have sensors to find marked trash cans, motors to move around in order to look for cans, as well as gripper arms to grasp and carry a can. Small trash cans are light and compact enough for a single robot to carry, but large trash cans require multiple robots to carry them out together. It is not certain exactly where the trash cans may be or how

fast they will fill up, but it is known that, because more people use them, the larger trash cans fill up more quickly. Each robot needs to take actions based on its own knowledge: while we encourage the robots to share some important information, we would not want them to communicate constantly, as this could overload the office's wireless network and seems wasteful from an energy perspective. Each robot must also ensure that its battery remains charged by moving to a charging station before it expires. The battery level for a robot degrades due to the distance the robot travels and the weight of the item being carried. Each robot only knows its own battery level (but not that of the other robots) and the location of other robots within sensor range. The goal of this problem is to remove as much trash as possible in a given time period. To accomplish this goal we want to find a plan, or policy, that specifies for each robot how to behave as a function of its own observations, such that the joint behavior is optimal. While this problem may appear simple, it is not. Due to uncertainty, the robots cannot accurately predict the amount of battery reduction that results from moving. Furthermore, due to noisy and insufficient sensors, each robot does not accurately know the position and state of the trash cans and other robots. As a result of these information deficiencies, deciding which trash cans to navigate to and when to recharge the battery is difficult. Moreover, even if hand-coding the solution for a single robot would be feasible, predicting how the combination of policies (one for each robot) would perform in practice is extremely challenging.

Efficient Sensor Networks

Another application that has received significant interest over the last two decades is that of sensor networks. These are networks of sensors (the agents) that are distributed in a particular environment with the task of measuring certain things about that environment and distilling this into high-level information. For instance, one could think about sensor networks used for air pollution monitoring [Khedo et al., 2010], gas leak detection [Pavlin et al., 2010], tracking people in office environments [Zajdel et al., 2006, Satsangi et al., 2015] or tracking of wildlife [Garcia-Sanchez et al., 2010]. Successful application of sensor networks involves answering many questions, such as what hardware to use, how the information from different sensors can be fused, and how the sensors should measure various parts of their environment to maximize information while minimizing power use. It is especially questions of the latter type, which involve local decisions by the different sensors, for which the Dec-POMDP framework studied in this book is relevant: in order to decide about when to sense at a specific sensor node we need to reason about the expected information gain from turning that sensor on, which depends on the actions taken at other sensors, as well as how the phenomenon to be tracked moves through the spatial environment. For example, when tracking a person in an office environment, it may be sufficient to only turn on a sensor at the location where the target is expected given all the previous observations in the entire system. Only when the target is not where it is expected to be might other sensors be needed. However, when communication bandwidth or energy concerns preclude the sharing of such previous information, deciding when to turn on or not is even further complicated. Again, finding plans for such problems is highly nontrivial: even if we were able to

specify plans for each node by hand, we typically would not know how good the joint behavior of the sensor network is, and whether it could be improved.

Concluding, we have seen that in both these examples there are many different aspects such as decentralization and uncertainties that make it very difficult to specify good actions to take. We will further elaborate on these issues in the remainder of this introductory chapter and give an overview of many more domains for which Decentralized POMDPs are important in Section 1.4.

1.2 Multiagent Systems

This book focuses on settings where there are multiple decision makers, or *agents*, that jointly influence their environment. Such an environment together with the multiple agents that operate in it is called a *multiagent system (MAS)*. The field of MAS research is a broad interdisciplinary field with relations to distributed and concurrent systems, artificial intelligence (AI), economics, logic, philosophy, ecology and the social sciences [Wooldridge, 2002]. The subfield of AI that deals with principles and design of MASs is also referred to as 'distributed AI'. Research on MASs is motivated by the fact that it can potentially provide [Vlassis, 2007, Sycara, 1998]:

- Speedup and efficiency, due to the asynchronous and parallel computation.
- Robustness and reliability, since the whole system can undergo a 'graceful degradation' when one or more agents fail.
- Scalability and flexibility, by adding additional agents as required.
- Lower cost, assuming the agents cost much less than a centralized system.
- Lower development cost and reusability, since it is easier to develop and maintain a modular system.

There are many different aspects of multiagent systems, depending on the type of agents, their capabilities and their environment. For instance, in a homogeneous MAS all agents are identical, while in a heterogeneous MAS the design and capabilities of each agent can be different. Agents can be cooperative, self-interested or adversarial. The environment can be dynamic or static. These are just a few of many possible parameters, leading to a number of possible settings too large to describe here. For a more extensive overview, we refer the reader to the texts by Huhns [1987], Singh [1994], Sycara [1998], Weiss [1999], Stone and Veloso [2000], Yokoo [2001], Wooldridge [2002], Bordini et al. [2005], Shoham and Leyton-Brown [2007], Vlassis [2007], Buşoniu et al. [2008] and Weiss [2013]. In this book, we will focus on decision making for heterogeneous, fully cooperative MASs in dynamic, uncertain environments in which agents need to act based on their individual knowledge about the environment. Due to the complexity of such settings, hand-

coded solutions are typically infeasible for such settings [Kinny and Georgeff, 1997, Weiss, 2013]. Instead, the approach advocated in this book is to describe such problems using a formal model—the *decentralized partially observable Markov decision process (Dec-POMDP)*—and to develop automatic decision making procedures, or *planning methods*, for them.

Related Approaches We point out that due to the multi-disciplinary nature of the field of MASs, there are many disciplines that are closely related to the topic at hand, and we point out the most relevant of them here.

For instance, in the field of game theory, much research focuses on extensive form games and partially observable stochastic games, both of which are closely related to Dec-POMDPs (more on this connection in Section 2.4.5). The main difference is that game theorists have typically focused on self-interested settings.

The 'classical planning problem' as studied in the AI community also deals with decision making, but for a single agent. These methods have been extended to the multiagent setting, resulting in a combination of planning and coordination, e.g. distributed problem solving (DPS) [Durfee, 2001]. However, like classical planning itself, these extensions typically fail to address stochastic or partially observable environments [desJardins et al., 1999, de Weerdt et al., 2005, de Weerdt and Clement, 2009].

The field of teamwork theory also considers cooperative MAS, and the *belief-desire-intension (BDI)* model of practical reasoning [Bratman, 1987, Rao and Georgeff, 1995, Georgeff et al., 1999] has inspired many teamwork theories, such as *joint intentions* [Cohen and Levesque, 1990, 1991a,b] and *shared plans* [Grosz and Sidner, 1990, Grosz and Kraus, 1996], and implementations [Jennings, 1995, Tambe, 1997, Stone and Veloso, 1999, Pynadath and Tambe, 2003]. While such BDI-based approaches do allow for uncertainty, they typically rely on (manually) pre-specified plans that might be difficult to specify and have as a drawback the fact that it is difficult to define clear quantitative measures for their performance, making it hard to judge their quality [Pynadath and Tambe, 2002, Nair and Tambe, 2005].

Finally, there are also close links to the operations research (OR) and control theory communities. The Dec-POMDP model is a generalization of the (single-agent) MDP [Bellman, 1957] and POMDP [Åström, 1965] models which were developed in OR, and later became popular in AI as a framework for planning for agents [Kaelbling et al., 1996, 1998]. Control theory and especially optimal control essentially deals with the same type of planning problems, but with an emphasis on continuous state and action spaces. Currently, researchers in the field of decentralized control are working on problems very similar to Dec-POMDPs [e.g., Nayyar et al., 2011, 2014, Mahajan and Mannan, 2014], and, in fact, some results have been established in parallel both in this and the AI community.

1.3 Uncertainty

Many real-world applications for which we would want to use MASs are subject to various forms of uncertainty. This makes it difficult to predict the outcome of a particular plan (e.g., there may be many possible outcomes) and thus complicates finding good plans. Here we discuss different types of uncertainty that the Dec-POMDP framework can cope with.

Outcome Uncertainty. In many situations, the outcome or effects of actions may be uncertain. In particular we will assume that the possible outcomes of an action are known, but that each of those outcomes is realized with some probability (i.e., the state of the environment changes stochastically). For instance, due to different surfaces leading to varying amounts of wheel slip, it may be difficult, or even impossible, to accurately predict exactly how far our recycling robots move. Similarly, the amount of trash being put in a bin depends on the activities performed by the humans in the environment and is inherently stochastic from the perspective of any reasonable model.[1]

State Uncertainty. In the real world an agent might not be able to determine what the state of the environment exactly is. In such cases, we say that the environment is *partially observable.* Partial observability results from noisy and/or limited sensors. Because of *sensor noise* an agent can receive faulty or inaccurate sensor readings, or *observations*. For instance, the air pollution measurement instruments in a sensor network may give imperfect readings, or gas detection sensors may fail to detect gas with some probability. When sensors are *limited,* the agent is unable to observe the differences between certain states of the environment because they inherently cannot be distinguished by the sensor. For instance, a recycling robot may simply not be able to tell whether a trash can is full if it does not first navigate to it. Similarly, a sensor node typically will only make a local measurement. Due to such sensor limitations, the same sensor reading might require different action choices, a phenomenon referred to as *perceptual aliasing.* In order to mitigate these problems, an agent may use the history of actions it took and the observations it made to get a better estimate of the state of the environment.

Multiagent Uncertainty: Uncertainty with Respect to Other Agents. Another complicating factor in MASs is the presence of multiple agents that each make decisions that influence the environment. The difficulty is that each agent can be uncertain regarding the other agents' actions. This is apparent in self-interested and especially adversarial settings, such as games, where agents may not share information or try to mislead other agents [Binmore, 1992]. In such settings each agent should try to accurately predict the behavior of the others in order to maximize its payoff. But even in cooperative settings, where the agents have the same goal and therefore are willing to coordinate, it is nontrivial how such coordination should be

[1] To be clear, here we exclude models that try to predict human activities in a deterministic fashion (e.g., this would require perfectly modeling the current activities as well as the 'internal state' of all humans in the office building) from the set of reasonable models.

performed [Boutilier, 1996]. Especially when communication capabilities are limited or absent, the question of how the agents should coordinate their actions is problematic (e.g., given the location of the other recycling robot, should the first robot move to the trash can which is known to be full?). This problem is magnified in partially observable environments: as the agents are not assumed to observe the complete state of the environment—each agent only knows its own observations made and actions taken—there is no common signal that they can use to condition their actions on (e.g., given that the first robot only knows that a trash can was not full four hours ago, should it check if it is full now?). Note that this problem is in addition to the problem of partial observability, and not a substitute for it; even if the agents could freely and instantaneously communicate their individual observations, the joint observations would in general not disambiguate the true state of the environment.

Other Forms of Uncertainty. We note that in this book we build upon the framework of probability to represent the aforementioned uncertainties. However, there are other manners by which one can represent uncertainty, such as Dempster-Shafer belief functions, possibility measures, ranking functions and plausibility measures [Halpern, 2003]. In particular, many of these alternatives try to overcome some of the shortcomings of probability in representing uncertainty. For instance, while probability can represent that the outcome of a die roll is uncertain, it requires us to assign numbers (e.g., $1/6$) to the potential outcomes. As such, it can be difficult to deal with settings where these numbers are simply not known. For a further discussion on such issues and alternatives to probability, we refer you to Halpern [2003].

1.4 Applications

Decision making techniques for cooperative MASs under uncertainty have a great number of potential applications, ranging from more abstract tasks located in a digital or virtual environment to a real-world robotics setting. Here we give an overview of some of these.

An example of a more abstract task is **distributed load balancing** among queues. Here, each agent represents a processing unit with a queue, and can only observe its own queue size and that of its immediate neighbors. The agents have to decide whether to accept new jobs or pass them to another queue. Such a restricted problem can be found in many settings, for instance, industrial plants or a cluster of webservers. The crucial difficulty is that in many of these settings, the overhead associated with communication is too high, and the processing units will need to make decisions on local information [Cogill et al., 2004, Ouyang and Teneketzis, 2014].

Another abstract, but very important domain is that of transmission protocols and routing in **communication networks**. In these networks, the agents (e.g., routers) operate under severe communication restrictions, since the cost of send-

ing meta-level communication (occupying the communication channels) is prohibitively large. For example, consider the problem faced by transmission protocols such as TCP: when should packets be sent across a shared channel to be both fair and efficient for the endpoint computers? Because each computer only has information such as the number of packets in its queue or the latency of its own packets, each computer must make decisions based on its own information. A simple two-agent networking example was first modeled as a Dec-POMDP by Bernstein et al. [2005], but more recently, more realistic congestion control problems have been studied in simulation [Winstein and Balakrishnan, 2013]. Peshkin [2001] treated a packet routing application in which agents are routers and have to minimize the average transfer time of packets. They are connected to immediate neighbors and have to decide at each time step to which neighbor to send each packet. Some other approaches to modeling and optimizing communication networks using decentralized, stochastic, partially observable systems are given by Ooi and Wornell [1996], Tao et al. [2001] and Altman [2002].

The application domain of **sensor networks** [Lesser et al., 2003], as mentioned above, had received much attention in the Dec-POMDP community. These problems are inherently partial observable and—when assuming that adding extensive communication infrastructure between the sensor nodes is infeasible—decentralized. In addition the systems they are intended to monitor are seldom deterministic. This means that these domains have all the facets of a Dec-POMDP problem. However, there are also some special properties, such as the fact that the nodes usually have a static location and that they typically[2] do not change the environment by monitoring, that make them easier to deal with, as we will discuss in Chapter 8.

Another interesting potential application area is the **control of networks of traffic lights**. Smart traffic light controllers have the potential to significantly increase the throughput of traffic [Van Katwijk, 2008], but controlling networks of traffic lights is still challenging, since the traffic flows are stochastic and the networks are large. To avoid a central point of failure and expensive communication infrastructure, the traffic lights should make decisions based on local information, but reasoning about non-local effects and interactions is necessary. A number of papers address such problems from the learning perspective [Wiering, 2000, Wiering et al., 2004, Bazzan, 2005, Kuyer et al., 2008, Bazzan et al., 2010]. Wu et al. [2013] presented a simplified Dec-POMDP traffic control benchmark. The structure present in such traffic problems is similar to the structure exploited by several recent solution methods [Oliehoek et al., 2015b].

A very exciting application domain is that of **cooperative robotics** [Arai et al., 2002]. Robotic systems are notorious for being complicated by stochasticity, sensor noise and perceptual aliasing, and not surprisingly many researchers have used POMDPs to address these problems [Roy et al., 2003, Pineau and Gordon, 2005, Theocharous et al., 2004, Smith, 2007, Kaelbling and Lozano-Pérez, 2013]. In case of multiple cooperative robots, as in the RECYCLING ROBOTS example, the POMDP model no longer suffices; in most of these domains full communication is

[2] This assumes the absence of the so-called *observer effect,* as present in quantum mechanics.

either not possible (e.g., there is too little bandwidth to transmit video streams from many cameras or transmission is not sufficiently powerful) or consumes resources (e.g., battery power) and thus has a particular cost. Therefore Dec-POMDPs are crucial for essentially all teams of embodied agents. Examples of such settings are considered both in theory/simulation, such as multirobot space exploration [Becker et al., 2004b, Witwicki and Durfee, 2010b], as well as in real hardware robot implementation, e.g., multirobot search of a target [Emery-Montemerlo, 2005], robotic soccer [Messias, 2014] and a physical implementation of a problem similar to RECYCLING ROBOTS [Amato et al., 2014].

A final, closely related, application area is that of **decision support systems** for complex real-world settings, such as crisis management. Also in this setting, it is inherently necessary to deal with the real world, which often is highly uncertain. For instance, a number of research efforts have been performed within the context of RoboCup Rescue [Kitano et al., 1999]. In particular, researchers have been able to model small subproblems using Dec-POMDPs [Nair et al., 2002, 2003a,b, Oliehoek and Visser, 2006, Paquet et al., 2005]. Another interesting application is presented by Shieh et al. [2014], who apply Dec-MDPs in the context of security games which have been used for securing ports, airports and metro-rail systems [Tambe, 2011].