Artif Intell Rev (2016) 45:25–96
DOI 10.1007/s10462-015-9438-6

Artificial intelligence based cognitive routing for cognitive radio networks

Junaid Qadir[1]

Published online: 3 September 2015

Abstract Cognitive radio networks (CRNs) are networks of nodes equipped with cognitive radios that can optimize performance by adapting to network conditions. Although various routing protocols incorporating varying degrees of adaptiveness and cognition have been proposed for CRNs, these works have mostly been limited by their system-level focus (that emphasizes optimization at the level of an individual cognitive radio system). The vision of CRNs as cognitive networks, however, requires that the research focus progresses from its current system-level fixation to the a network-wide optimization focus. This motivates the development of *cognitive routing protocols* envisioned as routing protocols that fully and seamlessly incorporate artificial intelligence (AI)-based techniques into their design. In this paper, we provide a self-contained exposition of various decision-theoretic and learning techniques from the field of AI and machine-learning that are relevant to the problem of cognitive routing in CRNs. Apart from providing necessary background, we present for each technique discussed in this paper their application in the context of CRNs in general and for the routing problem in particular. We also highlight challenges associated with these techniques and common pitfalls. Finally, open research issues and future directions of work are identified.

Keywords Routing · Cognitive networks · Artificial intelligence

1 Introduction

In cognitive radio networks (CRNs), nodes are equipped with *cognitive radios* (CRs) that can sense, learn, and react to changes in network conditions. Joseph Mitola coined the term

This work has been supported by Higher Education Commission (HEC), Pakistan under the NRPU programme.

✉ Junaid Qadir
junaid.qadir@seecs.edu.pk

[1] Electrical Engineering Department, School of Electrical Engineering and Computer Science (SEECS), National University of Sciences and Technology (NUST), Islamabad, Pakistan

"cognitive radio" in 1999 anticipating an evolution of the concept of software defined radios (SDRs) proposed also by Mitola in 1991. It was envisioned that incorporation of substantial artificial intelligence (AI)—in the form of machine learning, knowledge reasoning and natural language processing—into SDRs will help realize intelligent radios that can autonomously optimize its parameters (Mitola 2006). In a modern setting, this is achieved by incorporation of a cognitive engine (CE) that employs various AI-based techniques to build a knowledge base, based on which reasoning is performed to make 'optimal' decisions. After SDR technology, CRs represented the next big shift in the drive towards powerful programmable wireless *devices*. CRs are viewed as an essential component of next-generation wireless networks (Akyildiz et al. 2006; Haykin 2005), and have a wide range of applications including intelligent transport systems, public safety systems, femtocells, cooperative networks, dynamic spectrum access, and smart grid communications. CR can dramatically improve spectrum access, capacity, and link performance while also incorporating the needs and the context of the user.

Although cognitive behavior of CRNs can enable diverse applications, perhaps the most cited application of CRNs is dynamic spectrum access (DSA).[1] DSA is proposed as a solution to the problem of *artificial spectrum scarcity* that results from static allocation of the available wireless spectrum using the command-and-control licensing approach (Fette 2009). Under this approach, licensed applications represented by *primary users* (PUs) are allocated exclusive access to portions of the available wireless spectrum prohibiting other users from access even when the spectrum is idle. With most of the radio spectrum already being licensed in this fashion, innovation in wireless technology is constrained. The problem is compounded by the observation, replicated in numerous measurement based studies world over, that the licensed spectrum is grossly underutilized (Akyildiz et al. 2006; Fette 2009). The DSA paradigm proposes allowing secondary users (SUs), also called cognitive users, access to the licensed spectrum subject to the condition that SUs do not interfere with the operations of the primary network of incumbents.

While CRs have been defined differently (He et al. 2010), the following tasks are considered integral to them: (1) *observation* or *awareness*, (2) *reconfiguration*, and (3) *cognition*. In this paper, we will be occupied mostly with cognition as we seek to build cognitive, AI-based, routing protocols. Cognition subsumes both planning and learning with *planning* being the process of finding the appropriate action for particular situations to meet some system target, and *learning* being the process of accumulating knowledge based on the results of previous actions (He et al. 2010; Gavrilovska et al. 2013). Generally speaking, cognition for a CR entails understanding and reasoning about the radio environment so that informed decisions may be taken in order to optimize the performance of the radio and of the overall network. Both planning and learning are essential elements of cognition and a lot of research attention has rightly focused on incorporating cognition in CRs. Although, it is highly desirable to incorporate learning and adaptiveness into CRs to develop device level intelligence, it is important to point out that the larger vision of a 'cognitive network' will not be realized until network layer functions seamlessly incorporate intelligence (Thomas et al. 2006).

Cognitive networking broadly encompasses models of cognition and learning that have been defined for CRs while emphasizing an *end-to-end network-wide* scope. Such cognitive networks can perceive current conditions to plan, decide and act while catering to the network's overall end-to-end goals (Thomas et al. 2007; Fortuna and Mohorcic 2009). Figure 1

[1] DSA is such a dominantly cited application of CRNs that DSA and CRN are often assumed to be synonymous incorrectly. CRNs, in fact, is a much broader concept allowing for diverse applications representing intelligent behavior such as topology control, end-to-end routing, interference control, etc. (Fette 2009).

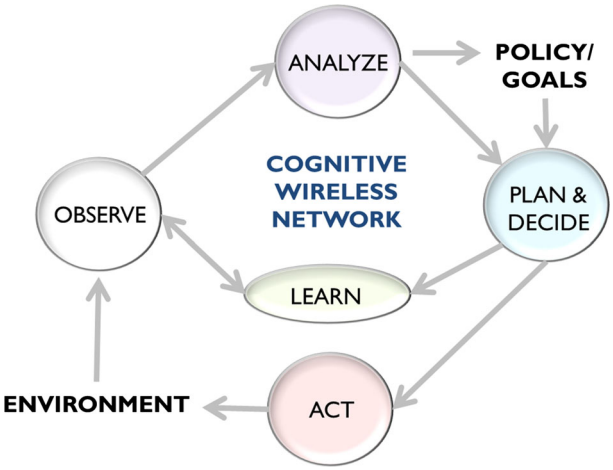

Fig. 1 Cognitive wireless network (CWN)

serves to illustrate the vision of cognitive networks as a class of networks that can observe and analyze the environment, and plan and decide to meet the policy constraints and objectives and goals and then act on the decided policy; importantly, such networks also learn from past interactions and the environment to improve performance over time.

To help CRNs become *cognitive networks* (CN), it is imperative that intelligence be integrated into the fabric of CRN architecture and protocols across the stack. In previous work on cognitive networks, Mähönen et al. proposed a cognitive resource manager as a framework for *network-wide* optimization of radio resources, and proposed utilizing machine-learning techniques to manage cross-layer optimization (Mähönen et al. 2006; Mähönen 2004). Some ten years ago, Clark et al. (2003) proposed that Internet must have a *knowledge plane* distinct from the data and the control planes that will allow building up an intelligent network capable of setting itself up given high level instructions, adapt itself to changing requirements, manage itself to automatically discover anomalies, and automatically fix problems or explain why it cannot do so. Clark et al. noted that building such a 'cognitive network' would require AI-based cognitive techniques and not just incremental algorithmic techniques.

While AI and computational intelligence has been for management of networks for quite some time (Sekercioğlu et al. 2001), the application of such techniques for routing is relatively limited. In this paper, we focus on the particular application of *AI-based cognitive routing* in CRNs. We will present decision-theoretic planning techniques and learning techniques that can be used to embed powerful AI techniques into the design of routing protocols for CRNs. While traditional wireless routing protocols do have some support for adapting to dynamic network conditions, cognitive routing protocols will enable a powerful new vision of adaptive network-wide intelligence that will facilitate dynamic optimization and will be an important cog in the overall framework of cognitive networking.

Contributions of this paper In this paper, we weave together ideas from multiple disciplines (such as optimization theory, game theory, machine learning, control theory, and artificial intelligence) to present a cogent and holistic overview of techniques that can be useful for network-layer decision making in CRNs particularly for the task of routing. This task has been non-trivial due to the multi-disciplinary nature of CRN research with different fields using different terminology and notation for related ideas. While this paper attempts to

be self-contained, exhaustive coverage of all related issues is not attempted due to the great breadth of the subject area. We instead focus on providing sufficient background on common AI techniques in the form of tutorial and then focus on discussing how these techniques may be used in the context of AI-based routing in CRNs. Previous survey articles that are similar to this work have focused mainly on application of machine-learning and AI techniques to problems of spectrum sensing, power control, and adaptive modulation in CRNs (He et al. 2010; Bkassiny et al. 2013). To the best of our knowledge, this is the first survey article that focuses on the application of AI techniques to the problems of routing and forwarding in CRNs.

Organization of this paper The rest of the paper is organized as follows. We begin by presenting an overview of traditional (non-AI-based) routing techniques in CRNs in Sect. 2. It is shown that while these routing protocols do support certain adaptive features, more work needs to be done to build AI-enabled cognitive routing protocols for CRNs. We list down important cognitive routing tasks in CRNs in Sect. 3. We then provide a detailed exposition of *decision and planning techniques* in Sect. 4. After providing necessary background of machine learning in Sect. 5, we discuss *learning techniques* at length in Sect. 6 and document their applications in CRNs and for routing. Open research directions are identified in Sect. 7. Finally, the paper is concluded in Sect. 8.

2 Traditional routing in CRNs

While our focus is on surveying techniques useful for *cognitive routing protocols* in the context of CRNs, it is also prudent to exploit and leverage the huge amount of previous work on traditional (i.e., non-AI-based) routing protocols for wireless networks. While wireless networks include both wireless local area networks (WLANs) and multi-hop wireless networks, our focus is going to be dominantly on multi-hop wireless networks such as mobile ad-hoc networks, wireless mesh networks and CRNs. We focus on these networks to build upon the insights that we can leverage for the design of effective cognitive routing protocols for CRNs. Previous work on routing in multi-hop wireless networks can be noted for the most part for the lack of learning from environment. Most of the classical wireless routing protocols do not utilize environment history for learning and predicting future evolution of environmental parameters and therefore cannot prioritize higher quality links over links of poor quality.

2.1 Traditional algorithmic routing approaches

Existing (non-AI-based) routing approaches have mostly relied on static graph-theoretic optimization-based algorithms. In this section, we will provide a broad introduction to such graph-theoretic algorithms while postponing a detailed discussion on optimization techniques to Sect. 4.

Graph theoretic models and algorithms Graph theoretic techniques are widely used for network routing problems both for wired and wireless networks. The first component of a graph theoretic solution is to first formulate the problem in a graph model by the process of *graph abstraction*. Thereafter, *graph algorithms* can operate on the abstract graph to solve the problem at hand (which can include shortest path problems, flow problems, etc.). The shortest path problems are especially relevant for routing since they aim to connect graph nodes to each other via the shortest possible paths. Network flow problems, on the other hand, typically aim to maximize the feasible flow through a single-source single-sink flow network.

Many important graph-theoretic algorithms for solving networks problems (especially the shortest path problems) are based on the technique of *dynamic programming* (Ahuja

et al. 1993). The term 'dynamic programming'[2] was originally used in the 1940s by Richard Bellman to describe the mathematical theory of *optimal multi-stage decision processes* in which one needs to make the best decision one stage after another. Popular dynamic programming algorithms for shortest path problems include the Bellman–Ford algorithm (for a single source) and the Floyd–Warshall algorithm (for all-pairs).

A detailed description of graph-based algorithmic techniques used for routing in CRNs is provided in Cesana et al. (2011). As can be seen in this paper, a host of techniques (such as *layered graphs*, *colored graphs*, and *conflict graphs*) can be used as part of a graph-theoretic routing solution. Since these approaches are mostly suited to static, or quasi-static, networks, we will not go into details of these methods here. The interested readers are referred to Cesana et al. (2011) for more details on these techniques.

Pitfalls and challenges Graph theoretic techniques typically assume full spectrum knowledge and static conditions. While full spectrum knowledge is a strong assumption, static approaches based on this assumption are applicable when a centrally maintained spectrum database (as proposed by FCC for opportunistic usage of white spaces in the spectrum below 900 MHz and in the 3 GHz range). However, in a more general setting, full spectrum knowledge is mostly inaccessible thus limiting techniques based on it. CRNs are also extremely dynamic in their conditions due to PU dynamics and are thus not well suited to static techniques.

2.2 Routing metric based categorization

Various *routing metrics* have been devised in CRNs to gauge the routing performance quantitatively (Youssef et al. 2014). A fundamental routing metric which intimately affects network performance is *delay* which is the average time required to deliver a packet from the origin to destination. The prime analytical framework used to analyze network delay is *queueing theory*. The problem of minimum delay routing has a long history with Gallager proposing a solution in 1977. Another fundamental routing metric is *throughput* which can be defined both for the network and for individual flows. The design of wireless routing protocols has to incorporate the (sometimes conflicting) requirements of minimum delay and throughput optimality. As an example, backpressure routing schemes (Tassiulas and Ephremides 1992; Dvir and Vasilakos 2011) are throughput optimal but are known to compromise on delay performance. This relationship, where throughput increase can result in deterioration in delay performance, can be seen in Fig. 2. The typical approach in networking is not to formulate the routing problem as an optimization problem (which would restrict their applicability to only static or quasi-static networks) but to compute a single shortest path from the origin to the destination using some heuristic link-cost metric. Heuristic algorithms typically adapt well to dynamic networks making them preferable for real networks. Apart from aiming to minimize delay and maximize throughput, routing protocols have been proposed that aim to maximize route stability and diversity, and minimize route-maintenance. In recent times, wireless-specific routing metrics have been developed [such as those enlisted in Campista et al. (2008) for wireless mesh networks].

While primitive protocols such as AODV, DSDV, and DSR have typically relied on basic metrics such as hop count or delay, other metrics were developed for wireless networks over time such as those that targeted: maximizing throughput (Couto et al. 2005), minimizing interference (Subramanian et al. 2006), load balancing (Raniwala and Chiueh 2005), and

[2] The term 'dynamic' in 'dynamic programming' refers to the temporal aspect of multi-stage decision making while 'programming' refers to optimization.

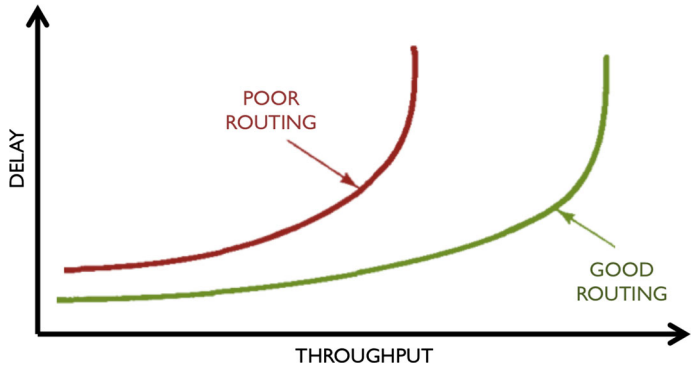

Fig. 2 The relationship of throughput versus delay [adapted from Bertsekas et al. (1992)]

choosing more reliable links (Couto et al. 2005). Since metrics designed for traditional wireless networks do not sufficiently capture the time-varying spectrum availability found in CRNs, some recent works have proposed more nuanced spectrum aware routing metrics (Pefkianakis et al. 2008; Huang et al. 2011; Zhu et al. 2008; Filippini et al. 2009; Caleffi et al. 2012). Many of these novel routing metrics are designed to be used with essentially 'least-used spectrum first' algorithms. Pefkianakis et al. (2008) proposed the SAMER protocol that proposed a new routing metric designed for CRNs that balances long-term route stability and short-term opportunistic high-performance. In another work, Huang et al. (2011) proposed new routing metrics incorporating spectrum temperature into routing which will favor the 'coolest' path which has seen the lowest spectrum utilization by the PUs. Zhu et al. (2008) have also proposed a new metric to be used with their hybrid (proactive/reactive) routing protocol that considers the PU activities as well as SU's QoS requirements. Filippini et al. (2009) have proposed another novel routing metric to cater to CRNs which considers the maintenance cost of routes that incorporates information about links that must be switched due to PU activity. Finally, Caleffi et al. (2012) have proposed a new routing metric OPERA for CRNs that is optimal (when combined with Bellman–Ford and Dijkstra based routing protocols) as well as accurate (since it measures the actual end-to-end delay of a route taking in account characteristics unique to CRNs).

We have categorized existing CRN routing protocols according to the routing metric being optimized in Table 1 with the categories being (1) throughput maximizing protocols, (2) delay minimizing protocols, (3) route-stability maximizing protocols, (4) route-maintenance minimizing protocols, and (5) diversity maximizing protocols. The reader is referred to Table 1 for references to, and a brief description of, routing protocols falling in these routing categories.

2.3 Routing approaches

In this section, we are going to describe some of the varied approaches proposed for routing in CRNs including a discussion on proactive versus reactive routing, opportunistic routing, multipath routing, and geographical routing.

2.3.1 Proactive versus reactive routing

In *proactive routing protocols*, each node maintains routing information about all the other nodes proactively so that a routing path is readily available when communication is needed.

Table 1 Summary of representative CRN routing protocols categorized per routing metric

References	Type	PU awareness and model	Comments
Throughput maximizing			
Cacciapuoti et al. (2012)	Reactive	Markov ON–OFF process	Reactive routing for mobile ad-hoc CRNs
Ding et al. (2009)		Not described	Cross-layer routing and dynamic spectrum allocation algorithm
SAMER (Pefkianakis et al. 2008)	Reactive	Bernoulli trial every t	Routes with highest spectrum availability ("least-used spectrum first")
SPEAR (Sampath et al. 2008)	Reactive	Not described	Joint spectrum and route discovery with distributed path reservations to minimize inter- and intra-flow interference
Delay minimizing			
How et al. (2011)	Reactive	2-State semi Markov model	Multi-metric (delay and stability) routing providing differentiated service
SEARCH (Chowdhury and Felice 2009)	Reactive	Not described	Designed for mobile CRNs based on geographic forwarding principles
CRP (Chowdhury and Akyildiz 2011)	Reactive	Markov ON–OFF process	Distributed joint route and spectrum selection protocol that explicitly protects PU receivers, and allows multiple classes of routes
Stability maximizing			
Coolest-first (Huang et al. 2011)	Reactive	Markov ON–OFF process	Proposed new routing metrics to capture the time-varying effects of spectrum availability
Tuggle (2010)	*Proactive*	Not considered	Proposes proactive multi-path routing
Gymkhana (Abbagnale and Cuomo 2010)	Reactive	Markov ON–OFF process	Path connectivity based distributed protocol that avoids poorly connected zones
Maintenance minimizing			
Zhu et al. (2008)	Hybrid	Not described	Combines proactive routing and on-demand route discovery
Filippini et al. (2009)		Ergodic random binary process	Optimal centralized, along with, distributed algorithms proposed both for exactly and statistically known PU activity
Diversity maximizing			
CAODV (Cacciapuoti et al. 2010)	Reactive	Markov ON–OFF process	Proposed a 'cognitive AODV' (CAODV) protocol for jointly exploiting path and spectrum diversity utilizing global information
D^2CARP (Rahman et al. 2012)	Reactive	Markov ON–OFF process	Proposed exploiting joint path and spectrum diversity to counteract effects of PU activity using localized information

Proactive routing typically entails exchange of control packets (conventionally known as the HELLO packets) through which the current topological and routing information is periodically exchanged. Such an approach is appropriate when the number of network nodes are less since the proactive approach entails significant control overhead. In *reactive* (*also known as on-demand*) *routing protocols*, on the other hand, the routes are determined on demand by sending *route request* messages when communication needs to take place. Such an approach avoids the extra overhead associated with proactive routing at the cost of some extra delay in computing routes at run-time. This approach is more appropriate for dynamic topologies or for large topologies where the overhead of proactive routing would be prohibitive. As seen in Table 1, an overwhelming majority of CRN routing protocols utilize some variant of a *reactive* or an *on-demand routing protocol* to avoid the overhead of managing dynamic topologies proactively.

2.3.2 Opportunistic routing

The conventional routing approach adopted on the Internet has been to compute a single 'best' path and to use this fixed path for forwarding. This approach to routing is primed mainly towards wired networking, and is not well suited to wireless networking in which a transmission is received not only by the intended receiver but also potentially by other nodes in its vicinity due to the wireless broadcast advantage. In addition, the wireless medium is considerably more unreliable and unpredictable compared to the wired medium. Opportunistic routing has been proposed to exploit the broadcast nature of wireless medium by not pre-committing to a particular route before data transmission and by choosing the relaying node opportunistically in run-time. In particular, after the sender has broadcasted the packet, the node 'closest' to the destination can be selected to broadcast to opportunistically take advantage of lucky transmissions that reach unexpectedly far. ExOR (Biswas and Morris 2005) was the seminal work in this domain in which an opportunistic routing protocol was proposed for wireless mesh networks. Various other opportunistic protocols have been proposed for wireless networks in general networks such as SOAR (Rozner et al. 2009) and ROMER (Yuan et al. 2005) and for CRNs in particular such as the opportunistic cognitive routing (OCR) protocol (Liu et al. 2012).

2.3.3 Multipath routing

Traditional routing protocols have focused on computing a single path according to the optimization metric used. Such protocols are not optimized for networks with highly dynamic topologies such as mobile ad-hoc networks (MANETs) and CRNs since the computed path may become unavailable as the topology changes due to mobility of nodes or spectrum non-availability. *Multipath routing protocols* aim to redress this issue by computing multiple alternative paths. Multipath routing protocols provide extra reliability, fault-tolerance, load balancing, and in the specific case of CRNs, the ability to maintain communication with minimal disruption in the face of topology change due to PU arrivals. Various multipath routing protocols have been proposed for wireless networks including work for MANETs (Nasipuri and Das 1999), wireless sensor networks (WSNs) (Ganesan et al. 2001) as well as for CRNs (Beltagy et al. 2011).

2.3.4 Geographical routing

Geographic routing is an approach used for routing in networks (especially in wireless networks) in which packets are routed to the destination based on the geographical position of the destination (Cadger et al. 2013). This approach uses the location/position information to obviate the dependence on network topology and thus the requirement to share topological information either periodically (as in proactive protocols) or on-demand (as in reactive protocols). This helps in reducing the overhead associated with exchange of topological information. Various geographical routing protocols have been proposed for wireless networks (Zorzi and Rao 2003; Jain et al. 2001) including work proposed for CRNs such as the SEARCH protocol (Chowdhury and Felice 2009).

2.4 CRN routing protocols

A wide variety of routing protocols have been proposed for CRNs and a representative summary can be seen at Table 1. In this section, we will focus on a broad categorization of routing protocols that have been proposed for CRNs on the basis of addressing scope and PU awareness. Lastly, we will provide a comparison of CRN routing protocols representative of the current state-of-the-art.

2.4.1 Addressing scope of routing

Most of the routing protocols in CRNs have addressed the scope of unicast routing with relatively few studies in the literature addressing the *broadcast* and *multicast* routing problems in CRNs.

The problem of *broadcast routing* in CRNs is challenging as noted in Akyildiz et al. (2006). In CRNs, channel heterogeneity of channels, intermittent connectivity, and lack of a common control channel can constrain the ability to perform effective broadcast routing (Akyildiz et al. 2006). Recently, a work has been proposed for fully distributed broadcast routing in CRNs without requiring a common control channel (Song and Xie 2012). An adaptive channel assignment scheme that modifies the assignment to suit broadcast routing when the broadcasting traffic volume is significant is presented in Mir et al. (2012). Some other works that have addressed the problem of broadcasting in CRNs include Htike and Hong (2013) and Fahad et al. (2010).

The problem of *multicast routing* can be considered as a generalization of both unicast and broadcast routing as unicast and broadcast are special cases of multicast (in which the receiver group goes from the one extreme of a single receiver to the other extreme of all the network nodes as receivers). Various approaches have been proposed for multicasting including those based in optimization theory, heuristic algorithms, and network coding (Li et al. 2012). Kim et al. (2009) have proposed a multicast routing protocol (COCAST) for mobile ad-hoc networks with nodes equipped with CRs. Their work aimed at improving the scalability of the traditional 'on-demand multicast routing protocol' (ODMRP) multicasting protocol in an environment using CRs. In another work, Almasaeid and Kamal (2013) have addressed the problem of assisted multicast scheduling in cognitive wireless mesh networks, and have proposed two approaches for cooperative multicasting: the first depending on the assistance of multicast receivers in delivering multicast data to other receivers, while the second is network-coding based. Some other works that have addressed the joint problems of routing, channel assignment, and scheduling for multicast communication in multihop CRNs have also been proposed (Almasaeid et al. 2010; Ren et al. 2009; Gao et al. 2011).

The interested reader is referred to a dedicated survey on the topic of multicasting protocols for CRNs for more details Qadir et al. (2014).

2.4.2 PU awareness and modeling

With dynamic spectrum access (DSA) being envisioned as a prime application of CRNs, it is important for routing protocols for CRNs to incorporate PU traffic dynamics into its design. Some of the CRN routing protocols have conspicuously not catered to PU dynamics in their design (Chowdhury and Felice 2009; Cacciapuoti et al. 2012; Sampath et al. 2008; Yang et al. 2008; Deng et al. 2007), although more recent work Pefkianakis et al. (2008), Filippini et al. (2009), How et al. (2011) and Ding et al. (2010) have importantly incorporated PU awareness. We will later describe issues relating to PU modeling in detail in Sect. 6.9.

2.4.3 Comparison of CRN routing protocols

Sun et al. (2013) have conducted a detailed performance evaluation of three representative CRN routing protocols: SAMER (Pefkianakis et al. 2008), Coolest Path (Huang et al. 2011), and CRP (Chowdhury and Akyildiz 2011) using both simulations (on the NS2 simulator) and an empirical evaluation (on a testbed of 6 node testbed based on USRP2 platform). The three protocols evaluated SAMER (Pefkianakis et al. 2008), Coolest Path (Huang et al. 2011), and CRP (Chowdhury and Akyildiz 2011) all have different design objectives. SAMER aims mainly at finding the highest throughput path while considering both the PU/SU activities and the link quality. Coolest Path is designed to prefer paths that are more stable since it prefers path with the highest spectrum availability. CRP is designed to either find a path with minimum end-to-end delay along with satisfactory PU protection, or to offer more complete protection to PU receivers at the cost of some performance degradation to SUs. It has been shown in their simulation and testbed results that SAMER provides the highest throughput under low PU activity (since SAMER aims to calculate throughput maximizing paths explicitly) and is also shown to be robust to packet loss; however, its performance under high PU activity deteriorates, particularly in the simulation results. Sun et al. also provide qualitative insights into the design of CRN routing protocols. Their findings suggest that taking link-quality and interference between SUs into account can greatly improve routing performance particularly under low PU activity. For high PU activity, however, path stability and path length become more important. Another important finding is that estimating spectrum availability based only on local observations cannot guarantee path stability therefore suggesting improvements can be made through cooperation.

In this section, we have not attempted an exhaustive survey of all existing CRN routing protocols, and instead have focused on presenting a representative overview of traditional (non-AI-based) wireless routing protocols. These protocols can be noted for their lack of incorporation of AI-methods for providing enhanced adaptive features. In the remainder of the paper, we will discuss various AI-based techniques, along with their applications in CRNs and routing, which can be used to construct future cognitive routing protocols. Interested readers are referred to the following survey papers on routing in CRNs and the references therein to find more information about the various routing protocols proposed for CRNs (Cesana et al. 2011; Al-Rawi and Yau 2013).

3 Cognitive routing tasks in CRNs

As noted earlier, although CRN routing protocols do mostly incorporate spectrum-awareness into their design, future cognitive networks will require greater architectural support from fully 'cognitive routing protocols' that will seamlessly incorporate AI-based techniques such as learning, planning, and reasoning in their design. Some of the important tasks that future cognitive routing should incorporate include:

T1. *Optimal decision making* Optimization may entail optimal configuration of a *single parameter* (e.g., deciding at a node the spectrum in a spectrum decision problem, or the next-hop for a routing problem) or of *multiple parameters* (e.g., interference control involving choice of multiple parameters such as transmission power, spectrum, etc.). Optimization in the setting of multiple agents must also support *interactive decision making* and incorporate not only the environment but also the decision of other agents in the decision making (e.g., two CRs operating in the same environment and attempting to maximize the throughput by choosing the transmission waveform).
T2. *Network-wide optimization* Cognitive routing protocols must support *cognitive networking* functionality with the network displaying intelligent behavior.
T3. *Adaptive behavior to accommodate network dynamics* to ensure that the network adapts to the continuously changing network environment and optimizes autonomously. Future cognitive routing protocols should provide native *support for dynamic spectrum access* (*DSA*) which is a cornerstone of modern CRNs.
T4. *Learning from experience* for an entity to display cognitive behavior, it is important that the entity supports learning from experience.[3]
T5. *Reasoning from learned knowledge* The intelligent entity must store its knowledge in a knowledge base and then support reasoning and inferencing to make *optimal* decisions. This step should be able to support any policy that needs to be supported.
T6. *Inference of future network dynamics* an intelligent cognitive routing framework can be more proactive in its decision making by inductively predicting future network dynamics and reacting accordingly.

3.1 Challenges in performing cognitive routing

Some challenges for effective cognitive routing in CRNs include (1) intermittent connectivity with neighbors in DSA networks causing a highly dynamic topology, (2) heterogeneous channels with diverse channel properties whose availability is time-varying (Akyildiz et al. 2006), (3) potential non-availability of common control channel (Lo 2011), (4) unknown, or incompletely known, environments, (5) ensuring intelligent network-wide behavior when multiple distributed agents interact selfishly with limited local knowledge, (6) unreliable spectrum sensing, and (7) limited signaling (or communication) between SU nodes (if any). These significant challenges complicate the problem of cognitive routing in CRNs. Various approaches tackle these challenges differently as we shall see when study decision making, planning techniques (Sect. 4) and learning techniques (Sect. 6.8) later on.

[3] While purely adaptive technology such as policy programmed expert systems can make radios increasingly aware, it is widely held that a radio must incorporate elements of learning to be deemed cognitive (Mitola 2006).

3.2 AI-based techniques for cognitive routing

The major focus of this paper is on AI based techniques that can be useful for routing in CRNs. Broadly speaking, AI based techniques comprise both decision making/planning techniques and machine learning techniques. Various decision making/planning AI techniques have been proposed in literature with optimization theory, Markov decision processes, and game theory being most relevant for the task of routing. We will provide a detailed exposition of concepts and applications of these techniques in Sect. 4. These decision-making and planning techniques addressing the tasks **T**1, **T**2, and **T**3 listed above. It is noted here that while most of the proposed routing protocols do include certain adaptive decision making features, relatively little work has been done to integrate AI-based *learning* techniques into the routing solutions for CRNs. This is a promising new sub-field ripe for future research exploration. We will provide necessary background and discuss cognitive routing applications of learning techniques (such as hidden Markov models, reinforcement learning, learning with game theory, online learning, artificial neural networks, learning with metaheuristic algorithms, and Bayesian learning) in Sect. 6.9. These *learning, reasoning, and inferencing* techniques address the tasks **T**4, **T**5, and **T**6 techniques.

4 Deciding, planning, and optimization

At the outset of this section, we will discuss some important *decision, planning, and optimization cognitive routing tasks*. These decision, planning, and optimization techniques are needed in the context of cognitive networks to address tasks **T**1 and **T**2 described in Sect. 3. In the remainder of this section, we will discuss *major decision making/decision planning frameworks* that have been widely applied to CRNs. Specifically, we shall be studying *optimization theory*, *Markov decision processes* and *game theory*. The cognitive cycle which epitomizes the essence of a cognitive radio is based on a cognitive radio's ability to: (1) *observe* its operating environment, decide on how to (2) best *adapt* to the environment, and then as the cycle repeats, to (3) *reason* and (4) *learn* from past actions and observations (Gavrilovska et al. 2013). The term *planning*, for the purpose of our discussion, refers to any computational process that produces (or improves) a decision *policy* of how to interact with the environment given a model of the environment. Planning is sometimes often referred to as a *search* task, since we are essentially searching through the space of all possible plans (Mitchell 1997; Sutton and Barto 1998).

4.1 Optimization theory

Optimization theory is a richly developed theory comprising tools and techniques for determining "optimal" decisions in scenarios which may also incorporate certain constraints (Keshav 2012; Hillier and Lieberman 2001). Optimization theory is directly applicable where the decision agent interacts with a static network topology and known radio environment (with full spectrum knowledge). While this strong assumption is not always satisfied, optimization techniques are important for all scenarios where the SUs have access to static databases storing the spectrum maps as propounded recently by Federal Communications Commission (FCC). Optimization techniques have also been leveraged extensively for CRNs with the assumption that PU dynamics are negligible allowing static design of channel assignment and the routing among SUs. It must be noted that optimization theory does not directly model interactions of the decision agent with other self-optimizing decision agents; Such interac-

tive optimization/decision making is the subject of the field of game theory (to be studied in Sect. 4.3).

Formally, a mathematical optimization problem has the following form: minimize $f_0(x)$ subject to $f_i(x) \leq b_i$. Here the vector $\mathbf{x} = (x_1, \ldots, x_n)$ is called the *optimization variable*, and the function (f_0: $\mathbf{R_n} \rightarrow \mathbf{R}$) of the optimization variable, that we have the objective of minimizing, is known as the *objective function*. The functions f_i: $\mathbf{R_n} \rightarrow \mathbf{R}$, $i = 1, \ldots, m$ are the (inequality) *constraint functions*, and the constants $b_1, \ldots, b_m$ are the limits, or bounds, for the *constraints*. A vector $\mathbf{x}^*$ "solves" the optimization problem, or is deemed *optimal*, if it has the smallest objective function value among all vectors that satisfy the constraints defined. There are various classes of optimization problems generally characterized on the basis of the form of the objective and the constraint functions. In particular, for *linear program*, the objective function f_0 and the m constraint functions $f_1, \ldots, f_m$ are all linear: i.e., $f_i(\alpha x + \beta y) = \alpha f_i(x) + \beta f_i(y))$. If the optimization problem is not linear, it is called a *nonlinear optimization* problem. The class of *convex optimization*, which includes linear optimization as a special case, the objective function f_0 and the m constraint functions $f_1, \ldots, f_m$ are all convex: i.e., $f_i(\alpha x + \beta y) \leq \alpha f_i(x) + \beta f_i(y))$. Due to the inequality in the preceding constraint function, convex programming can be of both linear and nonlinear types.

In this paper, we will be mostly interested in discrete optimization, also called *combinatorial optimization* in which either the constraint set is finite or it has a discrete nature. Informally speaking, combinatorial algorithms are techniques for high speed manipulation of combinatorial objects such as permutations, graphs, and networks (Knuth 2006; Papadimitriou and Steiglitz 1998). The three most important combinatorial optimization techniques are linear programming, integer programming, and convex programming.

4.1.1 Linear programming

(LP) is commonly applied in these fields to realize "optimal" logistical planning and scheduling. An application area of LP, much closer to our subject, is in *network optimization*. Typical network optimization problems, that may be formulated as linear programming problems, are the shortest path problem, the min-cut max-flow problem, and the minimum cost-flow problem (Ahuja et al. 1993).

4.1.2 Integer programming

(IP) is relevant for those optimization problem in which it only makes sense for certain optimization variables to take on integer values (e.g., in a networking context, the number of packets, flows, etc. generally make sense only with integral values). If in an optimization model, certain optimization variables can only take on integer values while other can take real values, the class of optimization model is known as a *mixed integer programming* (MIP) model. In general, IP and MIP problems can belong to either of the linear or the nonlinear class. An IP optimization problem that belongs to the linear class—i.e., whose objective and constraint functions are both linear—is said to belong to the class of integer linear programming (ILP); Analogously, the linear MIP optimization model is referred to as mixed integer linear programming (MILP). IP techniques are useful in communication networks for *synthesis*, *assignment* and *scheduling* problems (Resende and Pardalos 2006). MILP provides a very general framework for addressing problems with discrete decisions and continuous variables and are widely applied. While LP problems generally entertain polynomial-time solutions, IP programs are more complex to solve and typically are in the class of nondeterministic polynomial-time (NP). Since IP problems are computationally intractable (i.e.,

they are NP-complete or NP-hard), various *relaxation techniques* have been used for producing approximate solutions. IP models can be solved more efficiently if the problem has network substructure in which case *Lagrangian relaxation* can be used to decompose the IP. Other solution concepts for IP include *branch-and-bound* and *branch-and-cut*. The most common relaxation method used for solving IP, though, is the linear programming relaxation through the restriction of optimization variables taking integer values is relaxed.

4.1.3 Convex optimization

Convex optimization is a general class of optimization problems subsuming the least-square, linear-optimization, conic programming (Nemirovski 2006), and geometric programming (Chiang 2005) classes of optimization (Boyd and Vandenberghe 2004). Interest in convex optimization has been reinvigorated by a few notable recent discoveries. It has been shown that interior-points methods developed for solving linear programming problems are useful for solving a broader much wider class of convex optimization problems. Secondly, it is now realized that convex optimization problems (beyond least-square and linear optimization problems) are much more prevalent than previously thought (Boyd and Vandenberghe 2004). Indeed, it is now believed that convexity, and not linearity, defines the demarcation, or the "watershed", between tractable and intractable problems (Chiang et al. 2007). Many routing problems can be formulated as a convex optimization problem—e.g., the optimal minimum-delay routing problem (Gallager 1977) is an alias for the classic convex optimization problem of minimum-cost multi-commodity flow problem.

4.1.4 Distributed optimization

Distributed optimization is distinct from game theory which also performs optimization in a distributed fashion but for interactive environments in which each player is interested in its personal utility. In distributed optimization, all the distributed agents are trying to essentially jointly solve the same problem and do not have conflicting interest between personal utility and network. *Decomposability techniques* have been extensively used in optimization to lead to distributed (and often iterative) algorithms that converge to the global optimum. In wireless networks, distributed solutions are particularly attractive as a centralized solution may be non-scalable, too costly or fragile (Chiang et al. 2007). Decomposition theory naturally provides the mathematical language to build an analytic foundation for the design of modularized and distributed control of networks (Chiang et al. 2007). The method of decomposition is considered an extremely important versatile tool vital for practical distributed solutions of optimization problems. It has been stressed in Chiang et al. (2007) that the importance of "decomposability" to distributed solutions is akin to the importance of "convexity" in efficient computation of global optimum.

4.1.5 Multi-objective optimization

While most optimization problems aim to optimize for one explicit parameter, there is often a need in CRNs, where numerous design variables are controllable, to simultaneously optimize for multiple optimization variables (such as throughput, delay, energy, etc.). A key aspect of multi-object optimization that various objectives typically compete for dominance: e.g., it is impossible to jointly minimize both bit error rate (BER) and transmit power simultaneously (Rondeau and Bostian 2009). One approach to solving such a problem is to look for a solution

on the so called 'Pareto Frontier' that defines the set of input parameters that define non-dominated solutions in any dimension. Jie and Kamal (2014) have recently proposed two multi-objective optimization algorithms to find multicast trees that minimize the worst-case delay and the number of transmission links while simultaneously maximize the multicast rate with their proposed algorithms able to find over 60 % of the approximate Pareto front.

4.1.6 Application of optimization theory to CRNs

Common applications of combinatorial optimization techniques (such as linear, convex, and integer programming) include scheduling, assignment, route planning, set covering, etc. Integer programming (and mixed integer/linear programming) techniques are especially useful for a wide variety of assignment, scheduling, and resource allocation problems (Resende and Pardalos 2006). Optimization techniques have also been used in concert with signal processing techniques (such as compressed sensing) in previous work (Xiang et al. 2011). One of the pioneering applications of optimization theory in the context of CRNs was the work of Hou et al. (2008) that addressed the problem of optimal spectrum sharing for multi-hop networking. In particular, their work addressed the cross-layer optimization problem of minimizing the requirements of network-wide radio spectrum resources to support a given number of user sessions characterized in terms of source-destination pairs with given rate requirement. The problem formulation was a mixed-integer non-linear program (MINLP) and a near optimal solution based on a sequential fixing solution. In another work, Shi et al. (2008) have proposed a distributed optimization algorithm for multi-hop CRNs for cross-layer optimization that jointly considers power-control, scheduling, and routing. In recent times, there have been efforts in designing optimizable networks (Chiang et al. 2007; Palomar and Chiang 2006) and protocol design for communication networks as a distributed resource allocation problem (Shakkottai et al. 2008). The interested reader is referred to the handbook (Resende and Pardalos 2006) in which the various *applications of optimization techniques to telecommunications* are surveyed in detail.

4.1.7 Application of optimization theory to routing

The application of optimization techniques such as linear programming, constrained and iterative optimization, dynamic programming is discussed in an old survey paper of Ephremides and Verdu (1989) with a specific description of application of optimization techniques to the area of network routing. The seminal work focused on optimal routing was Gallager's work in 1977 which formulated the *minimum delay routing problem* and proposed a solution which only required distributed computation and simple periodic information exchange with its neighbors. This work, and follow up works on minimum-delay routing, required the input traffic and network topology to be static or quasi-static (changing very slowly) and also required knowledge of global constants to ensure convergence which made such optimal routing impractical for real networks. It was later realized that Gallager's algorithm was an in fact a solution to a special case of known convex optimization problems (Bertsekas 1979). Gallager's algorithm is in fact compatible with the class of distributed Bellman–Ford type algorithms (Ephremides and Hajek 1998). Optimization based routing solutions generally make the strong assumption of availability of full spectrum knowledge. Full spectrum knowledge is not always available which limits the scope of the applicability of optimization techniques in CRNs. However, the use of optimization techniques is justified where this assumption is justified (an example scenario being TV band whitespace networking where the

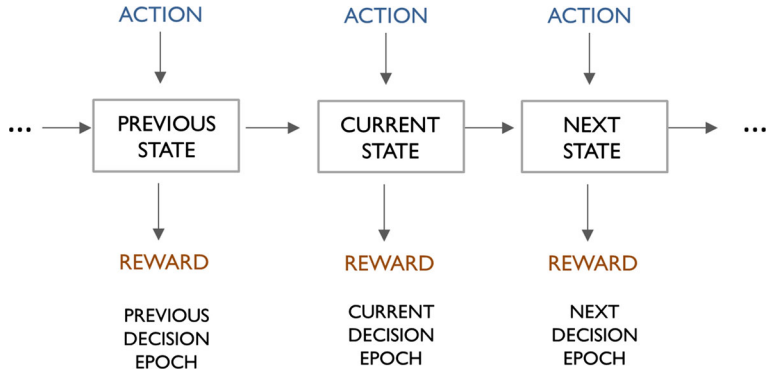

Fig. 3 The sequential decision making nature of MDP

SUs can query databases storing the spectrum map) and optimization based routing solutions (e.g., Shi et al. 2008; Ma and Tsang 2008) have been proposed for CRNs in such settings.

4.1.8 Pitfalls and challenges

Traditional optimization algorithms are designed for static environments with fully known characteristics. In CRNs, however, it is common for the environment to change rapidly and for only partial information to be available. The use of optimization theory in CRNs is also complicated by the fact that there is a high likelihood of the optimal point shifting due to the non-stationarity of the environment before the optimization solution converges. In such scenarios, the use of approximate, or near-optimal heuristic solutions may be warranted. Another motivation for the use of approximation algorithms is that most interesting optimization problems in CRNs are complex, typically NP-hard, and thus efficient practical solutions require some simplification/approximation to be made.

4.2 Markov decision processes

Markov decision processes (MDPs) is a widely used mathematical framework for sequential planning, or control, of a randomly evolving dynamical system. MDPs can be used to model the decision making of an agent in real-life stochastic situations where the outcome does not follow deterministically from actions (Puterman 2009). In such cases, the output (also, called the reward) is specified by a probability distribution that depends on the action adopted in a particular state. MDP can be envisioned as a discrete time stochastic optimal control process since it approaches the multi-stage decision-making process as an 'optimal' control problem in which the aim is to select actions that maximize some measure of long-term reward.[4] MDPs differ from classical deterministic AI planning algorithms in that its action model is stochastic (i.e., the outcome does not follow deterministically from the action chosen). A model of the sequential decision making nature of MDP can be seen in Fig. 3 in which it is illustrated that in MDPs an action is taken in each state corresponding to a decision epoch which returns a reward and changes the state of the system.

More formally, an MDP is a which can be defined as 5-tuple (S, A, P, R, γ) where S is a finite set of states, A is a finite set of actions, P is the transition function with $P_a(s, s')$

[4] Please see Fig. 4 to see how MDPs relate to other techniques and AI related fields.

representing the next-state distribution after adopting action a in state s and $\gamma \in [0, 1]$ is a discount factor. Every time step, the process is in some state $s \in S$, and the decision maker has to choose some action $a \in A$ from amongst the actions available in that state $A(s)$. After taking the action, the process will move randomly to some new state s', with the decision maker obtaining a corresponding reward $R_a(s, s')$. We note here that the reward is used in a neutral sense: it can imply both a positive reward or a negative reinforcement (i.e., a penalty). The choice of action a in state s influences the probability that the process will move to some new state s'. This probability (of going from state s to s' by taking action a) is given by the state transition function $P_a(s, s')$. The next state s', therefore, depends stochastically on current state s and the action a taken therein by the decision maker. In MDPs, an extra condition holds crucially: given s and a, the $P_a(s, s')$ is conditionally independent of all previous states and actions. This condition is known as the *Markov property* and this condition is critical for keeping MDP analysis tractable. In order to express preference for short-term rewards as compared to long-term rewards, a *discount factor* γ is often used which works by reducing future rewards by a factor of γ (chosen such that $0 \leq \gamma < 1$) for every future time step. If γ is chosen to be 0, the agent will become short sighted or 'myopic' and will consider current rewards only. As γ approaches 1, the agent will become long-sighted and it will strive for long-term rewards.

Solving an MDP The core problem in MDPs is to determine an optimal 'policy' for the decision maker which is defined to be a function π that maps a state s to an action $\pi(s)$. The roots of such problems can be traced to the work of Bellman (1957) who showed that the computational burden of solving an MDP can be reduced quite dramatically via techniques that are now referred to as *dynamic programming* (DP). Intuitively, the policy π specifies what action must the agent perform when in various states so that the long-term rewards are maximized. In a potentially infinite horizon environment, with continuous decision making which goes on forever, to reason about the various different possible policies, it is important that the reward function be finite. This is usually accomplished through discounting through which the preference of immediate rewards over delayed rewards may be quantified. To ensure that action values do not diverge, the discount factor should not be equal to, or exceed, 1. Solving an MDP now entails determining the policy π that maximizes the cumulative discounted reward function over a potentially infinite horizon: $\sum_{t=0}^{\infty} \gamma^t R_{a_t}(s_t, s_{t+1})$ where we choose $a_t = \pi(s_t)$, γ is the discount factor, and the subscript t refers to the time-step.

MDPs are sometimes referred to as *controlled Markov chains*. This, and the relationship of various Markov models and games that we will develop later in this paper, can be seen graphically in Fig. 4. To put MDPs into perspective, we note here that they are a generalization of Markov chains. The difference is that MDPs incorporate actions and rewards in the model while Markov chains do not. Conversely, the special case of MDPs with only one action available for each state and with identical rewards (e.g., zero) is in fact a Markov chain. It may be noted that once the MDP is specified with a policy, the action at various states is fixed, and the resulting MDP effectively behaves like a regular Markov chain.

We can also define the *value* of a state which follows naturally from the concept of rewards. Intuitively, the value of a state is a sum of discounted rewards that accrue from following the optimal policy onwards from that state. More precisely, $V(s)$ or the value of a state s will contain the expected sum of discounted rewards to be earned (on average) by following the policy π from state s. A *value function* is a mapping from the states to their values or expected upcoming cumulative reward. For compactness, we refer to $R_{a_t}(s_t, s_{t+1})$ where $a_t = \pi(s_t)$, or the reward achieved in time $t + 1$ by following the optimal policy π at time t simply as r_{t+1}. The value function mapping is shown below.

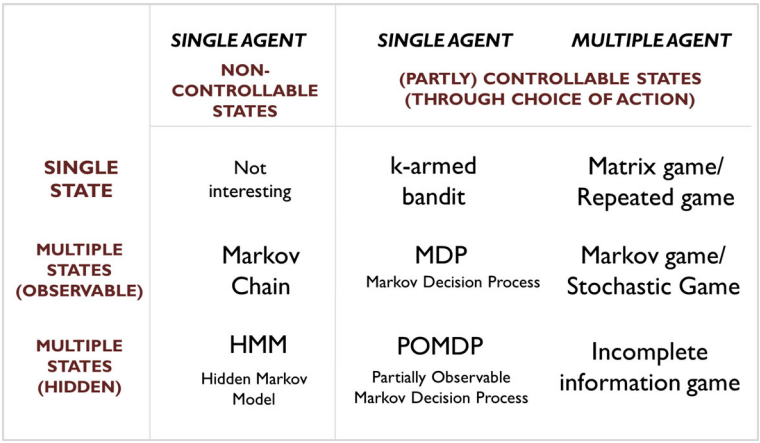

Fig. 4 Relationship between various Markov models, processes, and games

$$V(s_t) = E\left[r_{t+1} + \gamma r_{t+2} + \gamma r_{t+3} + \cdots\right] \tag{1}$$

It is worth emphasizing that the value abstraction is a key idea, and all efficient methods for solving sequential decision problems estimate value functions as an intermediate step (Talk on Deconstructing Reinforcement Learning' by Richard Sutton at ICML 2013). Apart from using the equation above (Eq. 1), another efficient, but remarkably simple, method can be used for calculating the value function on the basis of *bootstrapping*. We will see this method when we later will study Eq. 2 when the *Bellman equation* is introduced.

4.2.1 Dynamic programming solutions to MDPs

Assuming that we wish to calculate the policy that maximizes the expected discounted reward given that the state transition function P and the reward function R are known (this assumption is not always met, but we start with this simple case). The basic idea of the theory underlying dynamic programming is refreshingly simple. Optimal policy should be viewed as determining the decision required at each time in terms of the current state of the system. Regardless of the initial state and decisions, the remaining decisions must constitute an optimal policy π for the continuation process treating the current state as starting input. This is known as the *principle of optimality*. This strikingly simple insight allows computation of the optimal policy through backward induction starting at the terminal point. The concept of value function V is related to this, and it captures the expected future utility at any node of the decision tree, if we assume that an *optimal policy will be followed in the future*. The naive approach to the problem of optimal sequential decision making would be to consider the set of all feasible policies, compute the return for each, and then to choose the policy providing the maximum return. This brute-force approach will not work except for the most trivial problems and will be hopelessly inadequate for processes involving even a moderate number of states and actions.

There are two main approaches to provide a dynamic programming based solution to MDPs (value iteration and policy iteration) which we discuss next.

In *value iteration*, proposed by Bellman (1957), the policy function π is not used directly. The value of $\pi(s)$ is instead calculated indirectly within $V(s)$ whenever it is needed. This technique is also known by the name backward induction. Substituting the calculation of

$\pi(s)$ into the calculation of $V(s)$ gives us the following *Bellman equation* for this problem. The value iteration update works by iteratively calculating the values of $V(s)$.

$$V(s) = \max_a \left\{ \sum_{s'} P_a(s, s') \left(R_a(s, s') + \gamma V(s') \right) \right\} \tag{2}$$

It has been observed that it is possible to get an optimal policy even with inaccurate value function estimate, or before the convergence of the value function, especially when one action is clearly better than all others; in such a case, it becomes clear what action needs to be taken even with imprecise estimates of the exact value magnitudes (Russell and Norvig 1995). This insight can be exploited to devise a new strategy for calculating optimal policies called policy iteration which directly explores the policy space. *Policy iterating algorithms* begin from some initial policy π_0 and thereafter alternates between *policy evaluation* and *policy improvement* and terminates when the policy improvement step yields no change in the utilities. In 'policy evaluation', we are interested in calculating $V_i = V^{\pi_i}$ which provides the value of each state if π_i is to be executed for a given policy π_i. In 'policy improvement', on the other hand, we are interested in determining π_{i+1} using one step look ahead based on V_i for the given V_i.

The choice of the ideal solution method (value iteration or policy iteration) depends on various factors. If there are many actions, or if there exists already a fair policy, it is better to use policy iteration. On the other hand, if there are few actions, and acyclic state transitions, then value iteration is a better option. Interested readers are referred to texts with comprehensive treatment of MDPs for a more thorough discussion of these solution methods (Puterman 2009; Russell and Norvig 1995).

4.2.2 Partially observed MDPs

A MDP in which the environment is only partially observable is known as a partially observable MDP (POMDP). In the method discussed above for solving MDPs, it was assumed that the state s is known when the action is performed. This assumption does not hold for partially-observed MDPs in which the agent is unable to directly observe the state s but can avail an observation O which is probabilistically dependent on s. POMDPs are able to model the uncertain aspects of the environment such as the stochastic effects of actions, incomplete information and noisy observations over the environment. Although POMDPs have been known for decades, their widespread uptake is impeded for two main reasons: (1) it is difficult to satisfactorily model the environment dynamics (such as probabilities of action outcomes and the accuracy of data), and (2) it is difficult to solve the resulting model.

4.2.3 Decentralized MDPs

The cognitive routing task in CRNs involves multiple agents performing decision making in a decentralized fashion. The class of MDPs that deal with decentralized setting with multiple agents is known as DEC-MDP. DEC-MDP is a special case of POMDP in which the observations made by the agents (i.e., the CRs) collectively defines the state of the system with no individual agent having access to the complete set of observations. The general case of DEC-MDP is NEXP-Hard (class of decision problems which can not be solved in better than exponential time by non-deterministic Turing machines). Fortunately, in some special cases [including the formulation of optimal routing as a DEC-MDP (Friend 2009)], the problem can be polynomial-time solvable either for approximate or exact solution.

4.2.4 Solutions for complex MDPs

While classical value iteration and policy iteration DP algorithms perform well for simple and moderately complex MDPs, they break down for large-scale complex MDPs where the cost of computing, storing, and manipulating the so-called transition probability matrices becomes prohibitively high. In complex MDPs, two crippling problems arise: (1) *the curse of modeling*, and, (2) *the curse of dimensionality*. In the former problem, it becomes very difficult to compute the values of the transition probabilities while for the latter problem, storing or manipulating the elements of the so-called value function needed in DP becomes challenging due to the large dimensionality. Therefore, classical DP techniques are rather ineffective at solving large-scale complex MDPs (Gosavi 2009). If the probabilities of MDP are unknown, then the problem becomes a reinforcement learning (RL) task. We will see later that in RL we aim to determine for an agent what actions it should take in a stochastic environment. We will discuss more when we develop solutions for RL later in Sect. 6.2.

4.2.5 Application of MDPs in CRNs

MDPs have been applied to study a wide range of planning and optimization problems in CRNs. It is noted here that MDPs in their native form require complete knowledge of the system (such as the state transition probabilities and the number of states, etc.) and they are not directly applicable when CRs are operating in unknown RF environments. However, various techniques exist (such as reinforcement learning discussed later in Sect. 5.3) that can work in such scenarios where the environment is not completely known. Choi and Hossain (2011) proposed a partially observable Markov decision process (POMDP) based framework for channel access to opportunistically exploit frequency channels a primary network operates on. In another work, Zhao et al. (2007) had devised a POMDP framework to develop a cognitive MAC protocol. MDPs have also been applied extensively in communication networks. Interested readers are referred to a survey paper (Altman 2002) which highlights the applications of MDPs to communication networks, and also includes a discussion on its use for routing.

4.2.6 Application of MDPs for routing

The routing problem has been formalized in the setting of MDP in previous work. Lott and Teneketzis (2006) have proposed an MDP based formulation of opportunistic routing in which the optimal routing decision is to select the next relay node in each epoch based on the expected cost-to-forward from the neighbors to the destination. Opportunistic routing decisions, in contrast to conventional routing schemes, are made online by choosing the next relay node based on actual transmission outcomes and are modeled well by MDPs. The problem of optimal 'minimum-expected-cost' routing, in an application of the concept of 'cognitive networking' by Friend (2009), was formulated as a DEC-MDP (a special case of a POMDP). In other works, multi-agent POMDPs were used for network routing in Rathnasabapathy and Gmytrasiewicz (2003). Since the transition probabilities of the MDP are typically not available a priori, reinforcement based learning is popularly used to converge to an optimal policy even when the transition probabilities of the underlying MDP are unknown. Peshkin et al. have proposed a reinforcement learning based MDP solution for adaptive routing in Peshkin and Savova (2002). Nurmi (2007) presents a formulation of the routing problem as a reinforcement learning problem involving a POMDP and also presents an algorithm for solving this model.

4.2.7 Pitfalls and challenges

The MDP makes the Markovian assumption, based on which the future state only depends on the current state, which may not adequate model the problem being studied. MDP formulations typically require a stationary environment with known transition probabilities of the underlying Markovian process. MDPs are best suited to single user systems since in multi-user systems, the presence of other users make the system non-stationary (Xu et al. 2013). In practical CRNs, obtaining statistical information a priori may not be feasible limiting the applicability of MDPs in such cases. In addition, it common for the environment in CRNs to change rapidly leading to a likelihood of the optimal point shifting due to the non-stationarity of the environment before the convergence of the MDP. The convergence results available of MDPs generally apply only asymptotically (i.e., the solutions converge as the number of iterations becomes sufficiently large) and do not inform about the convergence speed. In practice, the convergence speed is extremely important and convergence needs to be fast for the MDP solution to be useful.

4.3 Game theory

Game theory is a mathematical decision framework composed of various models and tools through which we can study and analyze *competitive* interaction between multiple self-interested *rational* agents. Although, game theoretic models exist for both cooperative and non-cooperative settings, the ability to model competition mathematically distinguishes game theory from optimal control-theoretic frameworks such as the MDP (Haykin 2005). Game theory is also differentiated from optimization theory (which caters to a single decision maker scenario) in their ability to model *multi-agent decision making* scenarios where the decisions of each agent affect each other.

Every *game* involves a set of *players*, *actions* for each of the players representing how players interact, *preferences* for each of the players defined over all the possible outcomes. The preferences, or *payoffs*, are typically defined through a *utility function*, or a *payoff function*, which maps each possible outcome to a number representing that outcome's desirability. An outcome brings more reward, or is more desirable, if it has a higher utility (MacKenzie and DaSilva 2006). In order to maximize its *payoff*, each player acts according to its *strategy*. More formally, a game can be mathematically represented by the 3-tuple $G = (N, S, U)$ where N represents the set of players, S the set of strategies, and U the set of payoff functions.

The terms strategy and action should not be confused together: the strategy in fact specifies how the player should act in each possible situation, and can be envisioned as a complete algorithm documenting how the player will play the game. The strategy of a player can be a single action (for a single-shot or a *static* games) or a set of actions during the game (for a sequential or a *dynamic* games) (Felegyhazi and Hubaux 2006). A player's *strategy set* defines what strategies are available for it to play: the strategy set may be finite (e.g., when a choice is made from a countable discrete set of values) or infinite (e.g., when some continuous value is chosen). A *pure strategy* deterministically defines how a player will play a game, while a *mixed strategy* defines a stochastic definition by assigning probability to each pure strategy. The *strategy profile*, or the *action profile*, documents the strategy of each player and it fully specifies all actions in a game. The outcome of the game depends, possibly stochastically, on the player's strategy profile and returns payoffs to various players.

Game theory is popularly used in CRNs since each CR in a CRN interacts with a dynamic environment composed of other *rational* agents that sense, act, and learn while aiming to maximize personal utility. For games specific to CRNs, individual CRs typically represent

the players, and the actions may include the choice of various system or design parameters such as, e.g., the modulation scheme, transmit power level, flow control parameter, etc. One of the main goal of game theory is to determine *equilibria* points for a given game. These are sets of stable strategies in which individuals are unlikely to unilaterally change their behavior. To gauge their efficiency, these equilibria points are often contrasted with some notion of socially *optimal* point which produces the 'best' outcome when interests of all the players is taken into accounts.

In recent years, game theory has provided deep insights into how to design decentralized algorithms for resource sharing in networks particularly through the theory known as *mechanism design* sometimes known as reverse game theory. While traditional game theory focuses on analyzing how rational players would play a given game, in mechanism design, we are interested in engineering or designing a game which rational players will play into a desired equilibrium point. Intuitively, mechanism design aims to set up the game such that players do what the designers want them to do but because the players themselves want to do it (Keshav 2012).

4.3.1 Representation of games

There are two common ways of representing non-cooperative games. The *normal-form* representation of a game explicitly lists the payoff for each player of every conceivable outcome. This representation, also known as the *standard-* or *strategic-form*, is appropriate for static games of complete, and perfect information. For two player games, this can be depicted in a matrix form either as a pair of payoff matrices (one each for the *row player* and *column player*) or as a single payoff matrix (with an entry containing payoffs for both players). On the other hand, an *extensive-form* game is a representation that allows, unlike the normal-form games, explicit representation of temporal aspects of dynamic games such as the sequencing of players' possible moves and their choices at every decision point along with payoffs for all possible game outcomes. It also allows representation of the (possibly imperfect) information each player has about the other player's moves when making a decision, and of incomplete information (about the nature of the game) in the form of chance events encoded as moves by the player 'nature'. More details about representation of the games can be seen at MacKenzie and DaSilva (2006).

4.3.2 Solution concepts

In game theory, a *solution concept* formalizes the concept of 'solving' a game by predicting how rational players would play a specified game. These predictions, called *solutions*, describe what strategies would be chosen by players and, therefore, it also describes the predicted result of the game. The most commonly utilized solution concepts are the *optimality* concepts and the *equilibrium* concepts.

While *optimality* has a well-defined unambiguous meaning in optimal control problems (one-player games), optimality—in settings of multi-player decision making—is a difficult concept to define precisely. Equilibrium points are not necessarily optimal since equilibria points may not be *socially optimum* [e.g., as in the classical Prisoner's dilemma game (Nisan 2007)] where a strategy profile is considered as socially optimal if and only if it results in the highest *sum* of expected payoffs. A common notion of optimality in game-theory is that of *Pareto-optimality*. A strategy profile is stated to be a *Pareto-optimal* solution if no other joint decision of the players can improve the performance of at least one of them without

degrading the performance of another. It must be noted that achieving Pareto optimality does not imply equality nor fairness. Another optimality concept is the *Minimax* solution concept useful for non-zero-sum games in which it is aimed to minimize the maximum loss a player will face in the worst-case scenario (Basar et al. 1995).

We shall now discuss four concepts of equilibrium that are relevant to our subject.

Nash equilibrium The Nash equilibrium (NE) is a solution concept of a non-cooperative game involving two or more players. A NE is a stable equilibrium point of a game representing the situation where no player can benefit by changing its strategy unilaterally (i.e., by the player changing its strategy while other players keep their unchanged). In other words, a NE implies that each player's strategy is the *best response* against those of the others. It is noted that it is possible for games to have multiple NE. While NE is a very useful concept, analysis based solely on NE has many drawbacks as pointed out in Haykin (2005) and Halpern (2008). Also, the significant complexity of computing NEs has prompted development of alternative solution concepts.

Correlated equilibrium This is an intuitive solution concept that generalizes the Nash equilibrium and is much easier to compute.[5] The idea is that each player chooses its action after observing a common public signal. The player's strategy assigns an action to every possible observation. If no player has any incentive to deviate from the devised strategy, assuming that others do not deviate, the game is in correlated equilibrium.

Wardrop equilibrium This is a common solution concept useful for modeling selfish routing in transportation and telecommunication networks with congestion. It is assumed that in the study of transportation and telecommunication networks that the players (travelers or packets, respectively) choose the shortest perceived routes given the current traffic conditions. For a network in Wardrop equilibrium, all the flow paths in use for a source-destination pair have an equal delay. No other unutilized path has a lower delay in the Wardrop equilibrium.[6] A wireless routing analogue of this was explored in Raghunathan and Kumar (2009) where a flow-avoiding routing protocol was proposed.

Stackelberg equilibrium This solution concept applies to Stackelberg games. Stackelberg games aim to address the inefficiency of non-cooperative games (the equilibrium point of non-cooperative network, or the Nash equilibrium, typically displays suboptimal network performance) by employing a network manager/network agent which acts a *leader* which imposes its strategy on the individual selfish users that then behave as followers. Stackelberg strategy has previously been investigated for achieving network optimal routing (Korilis et al. 1997) and congestion control (Shenker 1995) in networks. The Stackelberg strategy is an important tool that can be leveraged for the design of network optimal cognitive routing strategies.

Game theory predicts the agents' equilibrium behavior typically without specifying by itself how to reach such a state. Algorithms for computing equilibria and determining the dynamics of games towards it is a subject studied in the fledgling discipline of *algorithmic game theory* which is at the intersection of game theory and algorithms (Nisan 2007). It

[5] Roger Myerson has pithily remarked that: "If there is intelligent life on other planets, in a majority of them, they would have discovered correlated equilibrium before Nash equilibrium."

[6] If this property was not met, the system would not be in equilibrium intuitively, for it would have been possible for a flow to reduce its latency by switching to an unutilized path.

has been shown that equilibrium points do not have necessarily have to socially optimal. An interesting question then is to quantify how inefficient the equilibria points (which are reached through self interested behavior) are with reference to the idealized 'optimal' situation (where the agents collaborate selflessly in a bid to maximize total utility. Since there can be multiple NE with varying overall payoffs, the comparison of the worst NE with the ideal is known as the 'price of anarchy' while the comparison of the best NE with the ideal is known as the 'price of stability' (Nisan 2007).

We have covered only the most basic solution concepts that are relevant to our subject. For a discussion on advanced solution concepts such as rationalizability, ϵ-Nash equilibrium, trembling-hand perfect equilibrium, we refer the interested reader to standard game theory texts (Leyton-Brown and Shoham 2008).

4.3.3 Categories of games

We introduce the various ways to categorize games through the following contrasting categories:

- *Cooperative versus noncooperative* In all game theoretic models, a basic primitive is the concept of a *player*. A player may be either be interpreted as an individual or alternatively as a group of individuals. After defining the set of players in a game, we may distinguish between two kinds of models: (1) in which we are dealing with the possible actions of individual players; (2) in which we are dealing with possible joint actions of groups of players. Models of the former kind (individual-based) are sometimes known as 'noncooperative', while those of the latter kind are correspondingly known as 'cooperative'. The difference can be summarized in that in a cooperative game, players can make binding commitments, while in noncooperative game, they cannot. A game in which the players are groups of individuals that can make binding commitments is also known as a coalition game (Saad et al. 2012).
- *Sequential versus simultaneous* In a *sequential game* (also known as a *dynamic game*), one player chooses his action before the others choose theirs—the latter player can utilize knowledge about the previous move to decide on its action. On the other hand, in *simultaneous games*, also known as *static games*, players choose their moves without being aware of other player's moves.
- *Static versus dynamic* In *static games*, alternatively known as *single-stage games* or *one-shot games*, it is assumed that there exists only a single time step implying that the players only have one move. In contrast, players in a *dynamic game* interact with each other sequentially over multiple time steps. *Repeated games*, also known as supergames, are a subclass of dynamic games in which the same stage game is played numerous times. Dynamic games have a strong connection with MDPs which are also models for sequential decision making. Repeated games (also known as matrix games) generalize MDPs and can be envisioned as single-state multi-user MDPs: multi-user since it is a game theoretic model that captures the interaction between multiple users and single-state since the same game is played at every stage. Stochastic games (also, known as Markov games) generalize MDPs differently: they can be considered as multi-state multi-user MDPs in which the game being played at each stage is stochastically dependent on the game played previously, and the action adopted therein, and thus may change at every stage. The relationship between dynamic games and MDPs can be observed in Fig. 4. Dynamic games are interesting since they capture the temporal and sequential nature of decision making involved in routing in CRNs. The study of dynamic game is taken in

a subfield of game theory known as *dynamic game theory* which can be envisioned as child discipline of game-theory and optimal control theory (Basar et al. 1995).

- *Finite versus infinite-horizon games* Depending on the number of stages, we can classify dynamic games into *finite-horizon games* and *infinite-horizon game*—the strategies for such games can hugely vary. Players in a repeated game, unlike those in simultaneous games, have the benefit of historic information which they can utilize to adapt their strategy. If players in a finite-horizon game are not aware of the duration of the game (which is clearly a common situation in practical interactions particularly in a networking setting), then infinite-horizon games with *discounting* can be used an appropriate model. As explained in Sect. 4.2, discounting entails decreasing the value of future stage payoffs in order to cater for the potentially abrupt end to the game thereby preferring payoffs in nearer-by time.
- *Complete versus incomplete information* A game with complete information is a game in which each player knows the exact game being played. The game is represented by 3-tuple $G = (N, S, U)$ with N representing the set of players, S the set of strategies, and U the set of payoff functions. This complete information is not known in games of incomplete information. We typically employ the model of a *Bayesian game* to model situations in which some of the parties are not certain of the characteristics of some of the other parties. Games with incomplete information should not be confused with games with imperfect information (in which the history of the game is not available to all players).
- *Perfect versus imperfect information* We refer to a game as a *perfect-information game* if the players have perfect knowledge of all previous moves in the game at any moment they have to make a new move. Since players in simultaneous games (which includes practical games like poker and bridge) do not know the actions of other players, simultaneous games are *imperfect-information games*. Only sequential games, therefore, can be games of perfect information, with an example sequential perfect-information game being chess. In games of imperfect information, while the actual moves of agents are not common knowledge, the game itself is. This is in contrast to Bayesian games where at least one player is unsure of the type (and therefore the payoff function) of another player.
- *Symmetric versus asymmetric* If the game is *symmetric*, the identities of the players may be changed without changing the payoff to the strategies. In other words, even if the role of the two players in a two-player symmetric game is reversed, the same payoffs would be observed. This condition does not hold for *asymmetric* games.
- *Zero-sum versus non-zero-sum* In a *zero-sum game*, the sum of payoffs of all the players must be zero—in other words, a player cannot get better off without affecting some other player's utility. A game which is not zero-sum is called *nonzero-sum game* or *variable-sum game*.
- *Flat versus hierarchical* In the usual game models reviewed up till now, it is assumed that all the players have the same status without any hierarchy. However, DSA based CRNs are essentially hierarchical due to the priority of PUs over SUs for the purpose of spectrum access. In game theory, Stackelberg game model is a way of modeling competition in a hierarchical setting in which there is a leader and several followers competing over certain resources.

4.3.4 Modeling routing with game theory

An important aspect of tackling routing problems through game theory is precisely how the game is modeled (i.e., how are the players defined, what are the utilities, etc.). As an

illustrative example (from MacKenzie and DaSilva 2006), consider a simple *source routing* setup in which the end-to-end path is specified by the source node. In this game, the source nodes may be considered as the *players*; to allow for the existence of multiple flows from a single source, it is also possible, and more convenient, to view a source-destination pair as a player instead. The *action* set available to each player is possibly the set of all possible paths from the source to the destination. Depending on how the game is formulated, a node may choose a single path from all the possible paths or even choose multiple paths and also how much of their flow to send on each route. *Preferences* in a routing game can take several forms just like many routing metrics exist for routing protocols to determine a route's quality. A simple way to formulate preferences can be to base it on end-to-end delay for a packet to traverse the chosen route with a short delay being preferable to longer delay. While such a simple example can be solved through optimization techniques (especially, if we consider a single source and destination pair or if the available routes are completely disjoint), the benefit of using game theory kicks in when we consider the interaction between multiple flows using common paths through the network.

4.3.5 Uncertainty in games

Uncertainty can come into games in three distinct ways: (1) a player may use chance to determine which strategy to use (such a strategy is known as mixed strategy), (2) the game itself can include random events, and (3) you may not be exactly sure what game you're playing—i.e., you may not know what strategies other players are capable of, or their payoffs precisely. The latter two points refer to the *incomplete information* nature of the game. In addition, the game may have *imperfect information* where the players do not know previous history or have *asymmetric information*. We note here that simultaneous games are always imperfect information games since players choose their moves without being aware of other player's moves.

Stochastic games introduced by Lloyd Shapley in 1950s, are games in which (potentially multiple) agents take decisions in a sequence of stages (i.e., in a dynamic game) and each player receives a payoff that depends probabilistically on the current state and the chosen actions (Haykin 2005). Intuitively speaking, the agents in a stochastic game repeatedly play games from a collection of games—the particular game played at any given iteration depends probabilistically on the previous game played and on the actions taken by all agents therein (Leyton-Brown and Shoham 2008). Stochastic games have been applied in wireless networks in areas such as flow control, routing, and scheduling (Hossain et al. 2009).

Stochastic games, also known as Markov games, generalize the concepts of MDPs, Markov chains, and repeated games. In particular, MDPs can be viewed as the special case of a single-agent stochastic game, Markov chains as a single agent stochastic game where each player has a single action in each stage, while repeated games can be viewed as a single state (or, single stage) stochastic game (Neyman and Sorin 2003). Stochastic games can be viewed as a bridge between game-theoretic models and MDPs. Stochastic games generalize the MDP model to permit a pair of agents to control state transitions (either jointly or in alternation). The relationship between Markov Chains, MDPs, POMDPs, and Markov (or stochastic) games can be seen in Fig. 4 where it can be noted that a one-state stochastic game is equivalent to an (infinitely) repeated game, while the special case of an one-agent stochastic game is equivalent to an MDP.

We have seen previously that MDP are appropriate models for reinforcement learning techniques that address the problem of a single agent learning through experience and interaction with an environment (assumed stationary). Stochastic games extend the concept of

Table 2 Summary of the various decision and planning techniques discussed in this paper

Decision techniques	Applications to routing	General application to CRNs
Optimization theory	Shortest-path graph-theoretic optimization problems	Optimal spectrum sharing (Hou et al. 2008)
	Gallager's minimum delay routing (Gallager 1977)	Power control and scheduling (Shi et al. 2008)
	Optimization based routing in CRNs (Shi et al. 2008; Ma and Tsang 2008)	Resource allocation (Shakkottai et al. 2008)
Markov decision processes	Routing in ad-hoc CRNs (Di Felice et al. 2010)	Opportunistic spectrum access: Choi and Hossain (2011)
	Routing in communication networks: see ref. in Altman (2002)	Medium access control (MAC): Zhao et al. (2007)
	MDP-based routing formulation (Friend 2009; Rathnasabapathy and Gmytrasiewicz 2003; Peshkin and Savova 2002; Nurmi 2007)	Cooperative spectrum selection: Di Felice et al. (2011)
Game theory	Routing games (Roughgarden 2007; Pavlidou and Koltsidas 2008; Han et al. 2011)	Resource allocation: see references in Maharjan et al. (2011) and Zhang et al. (2013)
	Mitigating selfish routing (Felegyhazi et al. 2006; Eidenbenz et al. 2005; Wang et al. 2004)	Spectrum sharing: Han et al. (2012); Van der Schaar and Fu (2009)
	Modeling routing: see references in MacKenzie and DaSilva (2006)	Medium access control (MAC): Akkarajitsakul et al. (2011)
		Security: see references in Liu and Wang (2010)
		Channel assignment: Duarte et al. (2012) and Farooq et al. (2013)

MDPs for multi-agent environments. In multi-agent environments, the other agents are also learning and adapting and thus the environment can no longer be assumed stationary. Stochastic games, also called competitive MDPs, allow us to model uncertainty in the players' operating environment by allowing probabilistic state transitions in a dynamic game.

MDPs are observable stochastic environments in which a single agent takes a decision by choosing an action given knowledge of the current state. A POMDP models partially observable stochastic environments in which a *single agent* takes a decision while being provided with partial knowledge of the current state. In incomplete information games, on the other hand, *multiple agents* control the transitions in the environment while having incomplete knowledge of the environment's state (Fig. 4). As pointed out earlier game theory, MDP, and game theory are closely related. The application of these techniques for CRNs generally, and for routing in particular is shown and summarized in Table 2.

4.3.6 Application of game theory in CRNs

In literature, there has been a lot of work in applying game-theoretic ideas to the design and analysis of general wireless networks including the works presented in MacKenzie and DaSilva (2006), Han et al. (2012), Srivastava et al. (2005) and Naserian and Tepe (2009).

Game theoretic ideas have been applied in CRNs widely in problems such as resource allocation (Maharjan et al. 2011; Zhang et al. 2013), spectrum sharing (Han et al. 2012; Van der Schaar and Fu 2009), medium access control (Akkarajitsakul et al. 2011), security (Liu and Wang 2010; Attar et al. 2012), etc. Van der Schaar and Fu (2009) presented a survey of spectrum-access games that are relevant to DSA CRN. Wang et al. (2010) a more general survey paper on the application of game-theoretic ideas in CRNs is presented. The book by Liu and Wang (2010) is a comprehensive game-theoretic treatment of cognitive radio networking and security. We will present the application of game-theoretic ideas specifically for routing in a later subsection. Interested readers are referred to the references Van der Schaar and Fu (2009), Wang et al. (2010) and Liu and Wang (2010), and the references therein, for more details.

The dynamism of the overall wireless ecosystem in CRNs has led researchers to explore utilizing models from other complex domains such as economics (Maharjan et al. 2011). In particular, CRNs—in their distributed nature, complexity and heterogeneity—have become analogous to real-world markets (Zhang et al. 2013) and are amenable to incorporation of market mechanisms and incentives. *Auction theory* is an interdisciplinary field that has shown itself to be particularly useful for CRN applications. Traditional static methods of managing spectrum have been shown to be grossly inadequate for modern CRNs, and the market mechanism of auctions seems to be a promising approach for distributed allocation of network resources. The concept of market equilibrium—which comprises of (1) the supplier and consumers both achieving maximum utility in the Pareto sense; (2) total demand being equal to the total supply; (3) the budgets of consumers being totally spent—has been applied for spectrum markets in multi-channel DSA based CRNs (Byun et al. 2014). In this work, Byun et al. formulated the problem of sharing of multiple channels in such settings is as a spectrum market and proposed both a centralized algorithm and a distributed algorithm to yield the equilibrium. We note that while most of the application of auction theory and market mechanisms in CRNs have been in the domain of resource allocation with a detailed survey provided in Zhang et al. (2013), auction theory can also be useful for the problem of routing since it is intertwined with resource allocation, although it remains to be seen if auction theory and market based techniques from economics will play a more direct role in AI-based routing.

4.3.7 Application of game theory for routing

In algorithmic game theory, routing in networks is a well-studied problem both in a general network setting (e.g., of transportation networks) (Roughgarden 2007) and also for Internet-like networks (Qiu et al. 2006). In general, centralized calculation of optimal routes is infeasible for a majority of network routing problems, leading to interest in distributed algorithms. Distributed algorithms can be viewed as 'selfish routing' since each agent intends to optimize for itself. Researchers have vigorously pursued questions that aim to quantify the performance degradation due to lack of coordination between the various 'players' of this *routing game*. In this regard, concepts of price of anarchy and price of stability, discussed earlier, have been proposed. It has been shown that while the price of anarchy is unbounded for the case of selfish routing in networks with general latency functions (Roughgarden 2007), results are much more encouraging for networks with linear latency functions (Roughgarden 2007) and for actual Internet-like networks (Qiu et al. 2006). Selfish routing in networks and their equilibria was first formally defined by Wardrop in 1952, and it has been a popular topic for researchers since.

Broadly speaking, there are two popular models of selfish routing games: *nonatomic selfish routing* in which there are very large number of players each controlling a negligible fraction

of overall traffic, and *atomic selfish routing* in which each player controls a non-negligible amount of traffic. Nonatomic selfish routing was first studied for transportation networks by Wardrop, and equilibrium in such games is known as Wardrop equilibria. It has been shown that for nonatomic selfish routing, the price of anarchy is the same as the price of stability. Nonatomic selfish routing has been applied to routing in communication networks where it is relevant to the 'source routing' paradigm in which the source node specifies a complete route for its traffic and in a distributed setting (Roughgarden 2007). The paradigm of distributed shortest-path routing, that is typically used on Internet-like networks, cannot be addressed by selfish routing unless the 'length' used to define the shortest paths coincide with the edge cost functions (Roughgarden 2007). Atomic selfish routing games were first considered by Rosenthal in 1973 who also introduced the concept of congestion games and potential games. The price of anarchy is also well understood for atomic selfish routing game (Roughgarden 2007).

A characteristic of a typical routing game is that each player is interested in finding a minimum cost path from the origin to the destination in a *congested* network, where the delay of an edge on some path depends on its congestion which in turn depends on the total of players using that edge in their path. Such a dependence on congestion is seen in a class of games known as *congestion games*, first proposed by Rosenthal in 1973. In a congestion game, the payoff of each player depends not only on the resource it chooses, but also on the number of players choosing the same resource. Congestion games are a special case of *potential games*. Fortunately, the equilibria points are guaranteed to be approximately optimal under best response dynamics (Nisan 2007) for potential games in general.

Repeated games and *potential games* have been shown to be especially relevant to the routing problem. In previous work, repeated games have been used to address the problem of selfish routing with punishment for unsocial behavior (Felegyhazi et al. 2006; Eidenbenz et al. 2005; Wang et al. 2004). The usage of potential games for routing has been well-explored (Roughgarden 2007). Potential games encompass many of the well-studied network routing and congestion games. Potential games have many desirable properties including (1) pure equilibria always exists, (2) the best response dynamics is guaranteed to converge, and (3) the price of stability (or, the ratio of the best NE to the optimal solution) can be bounded using a technique named the potential function method. Potential games are especially attractive from the point of view of analysis, since the incentives of all the players are mapped onto a single function, called the potential function, whose local optima correspond to the set of pure NE. There has been a lot of work in modeling wireless networking problems as potential games [see the references in Hossain et al. (2009) for more details] with most applications being in the domain of power control, waveform adaptation, and routing and congestion games.

In game-based routing optimization, a common goal is to minimize $C + D$ where C, representing congestion, is the maximum edge congestion while D, called dilation, represents the maximum path length. This optimization problem is known to be NP-complete. An alternative type of routing games is "quality of routing" (QoR) games that are similar to C + D-routing games but always have a Nash equilibria with the price of anarchy being small for most interesting instances of the game. In addition, outcomes of the QoR game provide approximation to the solution of the $C + D$-routing games. Busch et al. (2012) have studied the problem of QoR games as an approximate way of solving $C + D$ routing games. It was shown that QoR games always have a pure Nash equilibria that can be obtained with players utilizing best response dynamics to greedily improve their paths.

Interested readers are referred to a detailed survey of game-theoretic methodologies for routing models at Pavlidou and Koltsidas (2008), details about *routing games* and the analysis of the efficiency of its equilibria points at Roughgarden (2007), and a survey of application of various networking games in telecommunications in Altman et al. (2006).

4.3.8 Pitfalls and challenges

It is difficult to structure a game so that players converge to a desired equilibrium. Since a problem is a game only when multiple agents are involved in making decisions, a common pitfall is using game theoretic techniques where optimization techniques would have sufficed (MacKenzie and DaSilva 2006). Another common pitfall while using game theoretic techniques is to mix unduly tools from cooperative and non-cooperative games which can make the analysis unsound. Most dynamic game models require statistical information about the environment based on which a dynamic game formulation (repeated game or stochastic game) is devised. However, in the case of CRNs, the statistical information about the environment is not always known a priori making the straight forward application of game theoretic models problematic. In addition, as stated earlier for optimization based methods, we must consider the non-stationarity of the CRN environment due to rapid changes in the environment, which can lead to the shifting of the optimal operating point before the convergence of the game. The convergence speed for game theoretic models is another important area of concern. The convergence results, in cases where they exist, only apply asymptotically and do not inform about the speed of convergence. It is important to focus on the speed of convergence since it is possible for a model with no convergence guarantees to outperform an asymptotically convergent model in practical time frames.

5 Background: machine learning

Machine learning is a field of research that formally studies learning systems and algorithms and provides an ability to "adapt to new circumstances and to detect and extrapolate patterns" (Russell and Norvig 1995). Machine learning techniques are useful in diverse domains—such as pattern recognition, robotics, natural language processing, autonomous control systems—and are particularly suited to domains like CRNs where the agents must dynamically adapt to changing conditions.

Previous work on applying machine learning to CRNs Bkassiny et al. (2013) provide a comprehensive survey of applications of machine-learning techniques in CRNs, and divide learning applications for CRNs into two broad categories of *feature classification* and *decision making*. Feature classification mainly has applications in spectrum sensing and signal classification. Decision making has diverse applications in CRNs including adaptive modulation, power control, routing and transport-layer applications (Bkassiny et al. 2013). Decision making problems can be further classified into policy making and decision rules problems. In a policy making problem, an agent determines an optimal *policy* (or an optimal *strategy* in game theory terminology) to determine what actions it should perform over a certain time duration. In a decision rule problem, on the other hand, the problem is formulated as hypothesis testing problem and the aim is to directly learn the optimal values of certain design and operation parameters (Bkassiny et al. 2013). Bkassiny et al. also establish the relationship between learning and optimization and show that many learning algorithms converge towards the *optimal* solution concept in their respective applications (whenever it exists). Applications of machine learning to CRNs are vast (Clancy et al. 2007; Rondeau 2007), and we shall develop a more complete picture gradually as we proceed in this paper. Interested readers are referred to the surveys (He et al. 2010; Bkassiny et al. 2013), and the references therein, for a comprehensive complementary treatment of general applications of machine learning to CRNs.

Some challenges that confront learning algorithms in CRNs, as identified in Bkassiny et al. (2013), are as follows:

(1) Learning algorithms have to operate in certain cases in unknown RF environments without any supervision.
(2) Learning algorithms have to operate in environments that are only partially observable.
(3) Learning algorithms for CRNs require distributed algorithms due to the decentralized nature of CRNs and are properly envisioned in multi-agent learning which are more challenging that single-agent learning scenario.

Machine learning concerns itself with a learner using a set of observations to uncover the underlying process (Abu-Mostafa et al. 2012). There are principally three variations to this broad definition and machine learning can be classified into three broad classes with respect to the sort of feedback that the learner can access: (1) supervised learning, (2) unsupervised learning, and (3) reinforcement learning. We will cover these three kinds of learning next in Sects. 5.1, 5.2 and 5.3, respectively.

5.1 Supervised learning

In supervised learning, algorithms are developed to learn and extract knowledge from a set of training data which is composed of inputs and corresponding outputs assumed to be labelled correctly by a 'teacher' or a 'supervisor'. To understand supervised learning, imagine a machine that experiences a series of inputs: x_1, x_2, x_3, and so on. The machine is also given the corresponding desired outputs y_1, y_2, y_3, and so on, and the goal is to learn the general function $f(\mathbf{x})$ through which correct output can be determined given a new input x_i (not necessarily seen in the training examples provided).

The output can be a continuous value for a regression problem, or can be a discrete value for a classification problem. The objective of supervised learning is to predict the output given any valid input. In other words, the task in supervised learning is to discover the function through which an input is transformed into output. This contrasts with 'unsupervised learning' in which the example of objects are available in an unlabelled or unclassified fashion.

There are essentially two types of supervised learning problems—classification and regression (or estimation). Classifiers itself can be further classified into *statistical classifiers* such as linear classifiers (e.g., Naive Bayes classifier or logistic regression), hidden Markov model (HMM) and Bayesian networks, or *connectionist* classifiers such as artificial neural networks (ANN) or *computational classifiers* such as support vector machines (SVM). We will discuss ANNs as a representative supervised learning technique later these techniques in detail later in a dedicated section on learning techniques (Sect. 6).

A central result in 'supervised learning theory' is the 'no free lunch theorem' which informs that there is no single learning method that will outperform all others regardless of the problem domain and the underlying distributions. For this reason, a variety of domain and application specific techniques have emerged to deal with diverse applications with varying degrees of success. The design of practical learning algorithms is therefore a mixture of art and science (Kulkarni and Harman 2011).

The major issue with supervised learning is the need to generalize a function from the learned data so that the technique may be able to conjure up the correct output even for inputs it has not explicitly seen in the training data. This task of generalization cannot be solved exactly without some additional assumptions[7] being made about the nature of the target function as it is possible for the yet unseen inputs to have arbitrary output values. Potential

[7] These assumptions are subsumed in the phrase *inductive bias*. See Mitchell (1997) for more details.

problems arise in supervised learning of creating a model that is underfitted (perhaps due to limited amounts of training data) or overfitted (in which a unnecessarily complex model is built to model the spurious and uncharacteristic noisy attributes of data). Depending on the application, huge amounts of training data may be necessary for the supervised learning algorithm to work.

5.1.1 Application of supervised learning to CRNs

Supervised learning techniques (such as HMM, ANN, etc.) have been extensively applied in CRNs. The application of individual learning techniques to CRNs and to the routing problem under respective headings in Sect. 6. The main applications of supervised learning techniques have been in the domain of signal classification—which although not directly relevant to the problem of routing can be used for solving concomitant problems that accompany routing such as PU detection, spectrum modeling, etc. (see Sect. 6.9 for more details of such tasks).

5.1.2 Application of supervised learning to routing

Due to the need of training, supervised learning techniques have not been used much directly for the routing task which requires online learning in potentially unknown environments. Supervised learning techniques are expected to play a minor role in such environments.

5.1.3 Pitfalls and challenges

Since supervised learning typically makes use of historical data as the training data, an underlying assumption of the stationarity of the environment is made. In CRN, where the environment is typically non-stationary, the use of supervised learning must be used with caution while considering the potential effect of 'concept drift' which refers to the changing of the statistical properties of the target variable over time in unanticipated ways.

5.2 Unsupervised learning

In supervised learning, it was assumed that a labeled set of training data consisting of some inputs and their corresponding outputs was provided. In contrast, in unsupervised learning, no such assumption is made. The objective of unsupervised learning is to identify the structure of the input data. To understand unsupervised learning, again imagine the machine that experiences a series of inputs: x_1, x_2, x_3, and so on. The goal of the machine in unsupervised learning is to build a model of $\mathbf{x}$ that can be useful for decision making, reasoning, prediction, communication, etc.

The basic method in unsupervised learning is clustering (which can be thought of as the unsupervised counterpart of the supervised learning task of classification). This clustering is used to find the groups of inputs which have similarity in their characteristics. In this paper, we will discuss Gaussian mixture models, non-parametric Bayesian clustering techniques and hidden Markov models (HMM) in Sect. 6 all of which can used for unsupervised clustering.

5.2.1 Application of unsupervised learning in CRNs

An application to which unsupervised learning is particularly suited is the extraction of knowledge about primary signals on the basis of measurements (Bkassiny et al. 2013). A

prominent (non-parametric) unsupervised classification technique that has been applied to CRNs particularly for this problem is the Dirichlet process mixture model (DPMM). The DPMM is a Bayesian non-parametric model which makes very few assumptions about the distribution from which the data are drawn by using a Dirichlet process prior distribution (Teh et al. 2006). The benefit of Dirichlet process based learning is that training data is not needed anymore, thus allowing this approach to be used for identification of unknown signals in an unsupervised setting. Dirichlet process has been proposed in literature (Shetty et al. 2009) for identifying and classifying spectrum usage by unidentified systems in CRNs.

5.2.2 Application of unsupervised learning for routing

Unsupervised learning techniques (such as learning with game theory, online learning, etc.) provide a promising platform for tackling routing problems in CRNs since these techniques do not require training data and can learn without direct supervision. These techniques and their applications in the context of routing are later discussed in Sect. 6.

5.2.3 Pitfalls and challenges

Although unsupervised learning has the desired property of being able to learn without any training data, unsupervised learning typically takes a long time to converge. Overfitting is also a known problem with unsupervised techniques.

5.3 Reinforcement learning

Reinforcement learning (RL) is inspired from how learning takes place in animals. It is well known that an animal can be taught to respond in a desired way by rewarding and punishing it appropriately; conversely, it can be said that the animal *learns* how it must act so as to maximize positive *reinforcement* or reward. RL is distinct from supervised learning in that instead of being presented with training examples of how to select the correct output for an input, the system has to learn indirectly from reinforcements (called reward for positive reinforcement and punishment for negative reinforcement) on actions taken. RL is also distinct from supervised and unsupervised learning in that it focuses on online performance (*learning through taking actions*) rather than on *planning* and offline performance. Since RL can be used without training data and because it aims to maximize the long-term online performance, it is particularly suitable for CRNs.

To understand RL, we consider Fig. 5 in which the CR acts as an agent and interacts with the RF environment by taking certain actions. RL, like MDP, is used to model sequential decision making where the agent takes a decision a_t in each time step t (also known as time epoch) while being in environmental state s_t. One time-step later, the agent receives a numerical reward or reinforcement[8] r_{t+1} as a consequence of the action taken a_t and finds itself in a new state s_{t+1}. The mapping from the actions to rewards is probabilistic in general. The objective of a reinforcement learner is to discover a *optimal policy* (i.e., a mapping from situations to actions) such that *expected* long-term reward is maximized in an *unknown stochastic* environment. We note here that MDPs, on the other hand, address this planning problem for *known stochastic* environments.

Since RL agents work in a stochastic environment, they have to balance two potentially conflicting considerations: on the one hand, it needs to *explore* the feasible actions and their

[8] The reinforcement is a scalar value that can be negative to express a punishment or positive to indicate a reward.

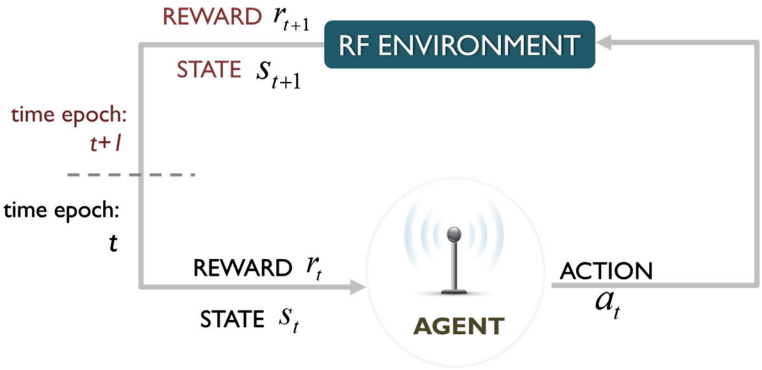

Fig. 5 The interaction of the CR node as an RL agent with the RF environment [adapted from Sutton and Barto (1998)]

consequences (to ensure that it does not get stuck in a rut) while on the other hand, it needs to *exploit* the knowledge, attained through past experience, of favorable actions which received the most positive reinforcement. We will discuss RL in considerable detail when we study RL as a specific learning technique in Sect. 6.2.

5.3.1 Applications, pitfalls, and challenges

We will describe the applications of RL in CRNs and in routing along with a description of major challenges and common pitfalls after a more thorough discussion on RL algorithms and techniques later when we discuss it as a learning technique in Sect. 6.2.

6 Learning techniques

Learning is especially crucial when dealing with unknowns or unplanned scenarios and is especially relevant to CRNs (Haykin 2005). Learning, for the purpose of our discussion, will focus on computational processes employed by CRs that can improve their behavior through diligent study of their own interactions with the environment. Learning can also be envisioned in the perspective of search. In this context, we can envision learning as searching through a space of possible hypotheses to determine which hypothesis best fits the available training examples and prior knowledge and constraints (Mitchell 1997). Learning techniques are needed in the context of cognitive networks to address tasks **T**3 and **T**4 described in Sect. 3. In the remainder of this section, we will discuss hidden Markov models, reinforcement learning, online learning algorithms, learning with game theory, metaheuristic learning, artificial neural networks, and finally methods of Bayesian inference.

6.1 Hidden Markov models

Hidden Markov model (HMM) are stochastic models of great utility, especially in domains where we wish to analyze temporal or dynamic processes such as speech recognition, PU arrival pattern in CRNs, etc. HMMs are highly relevant to CRNs since many environmental parameters in CRNs are not directly observable.

An HMM-based approach can analytically model Markovian stochastic processes whose actual states are hidden, but which emit observations from states per some probability dis-

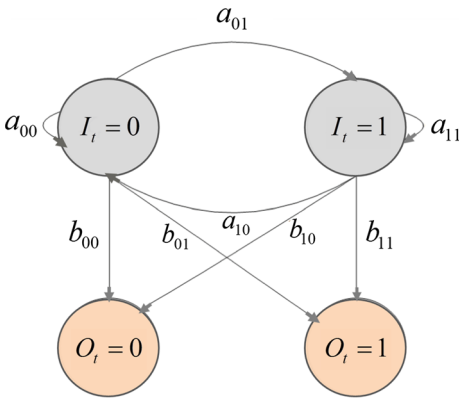

Fig. 6 A depiction of an extremely simple HMM with 2 hidden states and 2 observable states. $A = \{a_{00}, a_{01}, a_{10}, a_{11}\}$ is the (hidden) state transition probability, while $B = \{b_{00}, b_{01}, b_{10}, b_{11}\}$ is the observation symbol probability distribution

tribution. It is for this reason that an HMM is defined to be a doubly stochastic process: first, the underlying stochastic process that is not observable, and second, the set of stochastic processes, dependent on the embedded underlying stochastic process, that produce the sequence of observed symbols (Rabiner and Juang 1986).

Intuitively, HMMs can be visualized as a Markov chain observed in noise (Cappé et al. 2005). In a simple Markov model like a Markov chain, the state is directly visible to the observer, and the model is completely specified by describing the parameters defined through state transition probabilities. In an HMM, on the other hand, a more elaborate model is needed. The relationship of HMM with other Markov models is depicted in Fig. 4.

To represent an HMM, we use the notation $\lambda = (A, B, \pi)$ to represent an HMM where A, B and π are three probability distributions. A is the *state* transition probability, B is the *observation symbol* probability distribution from various states (Rabiner and Juang 1986), while π is the initial state distribution. Specifying an HMM completely requires, in addition to A, B and π, information about the number of states N and the number of discrete output symbols M. A simple example HMM with 2 hidden states and 2 observable states is presented in Fig. 6 with the state transition probability A and the observation symbol probability B shown.

6.1.1 Key problems in HMMs

Having defined the notation for HMMs above, we can talk about the three key problems that must be solved for the HMM to be useful in real world applications (He et al. 2010; Rabiner and Juang 1986). The listing of these three keys problems below assumes an observation sequence $O = O_1, O_2, O_3, \ldots O_T$.

- *Learning or training problem* Given an observation sequence O, this problem deals with learning the most appropriate model $\lambda = (A, B, \pi)$ that 'best' explains the observed sequence. In other words, we have to learn the most likely set of state transition A and observation symbol probabilities B from the training data. For many applications, this is the most important task since it allows us to optimally adapt model parameters to the training data. The *Baum–Welch* expectation-maximization algorithm solves this problem.
- *Evaluation problem* Given the parameters of the model λ, this problem deals with how to compute the probability of a particular observation sequence $Pr(O|\lambda)$. The forward algorithm, backward algorithm, and the forward-backward algorithm solve this problem.[9]

[9] While the forward-backward algorithm solve the evaluation problem (i.e., it can estimate the most likely state for any point in time), it cannot solve the decoding problem (of finding the most likely *sequence* of states) for which the Viterbi algorithm is used.

The evaluation problem in HMMs is intuitively related to learning problem in the following way. The evaluation problem computed $Pr(O|\lambda)$ which represented the probability of a particular observation sequence given a model. $Pr(O|\lambda)$ is also the likelihood function for λ given the observations O. The learning problem is determining the HMM parameters λ that maximize the likelihood function. The Baum–Welch algorithm is an iterative algorithm which solves the learning problem by expectation-maximization to produce maximum likelihood, or maximum a posteriori, estimates of HMM parameters given only observation sequence as training data.

- *Decoding problem* Given the observation sequence O and the parameters of the model λ, this problem deals with decoding or inferring about the sequence of hidden states $I = I_1, I_2, I_3, \ldots I_T$ that most likely produced the observation sequence O. This task aims at decoding, or uncovering, the hidden part of the HMM and is essentially an estimation problem. The *Viterbi algorithm* solves this problem by providing the most likely sequence and its probability.

We have already noted that HMM is a strong generic temporal model for dynamic signals and systems. Prediction/inference techniques have been widely deployed in networking [e.g., for prediction-based data aggregation in WSNs (Wei et al. 2011)]. To hone onto the important problem of inference in such temporal models, we note that there are four basic inference tasks that may be performed with HMMs (Russell and Norvig 1995). (We use the notation I_t and O_t to indicate respectively the hidden state and the observation during time step t. It is assumed that observations $O_0, O_1, \ldots, O_{t-1}$ have been observed till date.)

- *Filtering or monitoring* This is the task of computing the posterior distribution over the *current state*, given all evidence to date. Mathematically, this is calculating $P(I_{t-1}|O_0, \ldots, O_{t-1})$
- *Prediction* This is the task of computing the posterior distribution over the *future state*, given all evidence to date. Mathematically, this is calculating $P(I_t|O_0, \ldots, O_{t-1})$.
- *Smoothing or hindsight* This is the task of computing the posterior distribution over *past states*, given all evidence up to present. Mathematically, this is calculating $P(I_k|O_0, \ldots, O_{t-1})$ for $0 \leq k \leq t-1$
- *Most likely explanation* This is the task mentioned earlier as the *decoding* task. The aim is to find the most likely sequence of states that generated the observed sequence. Mathematically, this is $\arg\max_{I1:t} P(I_{1:t}|O_{1:t})$

6.1.2 Application of HMMs in CRNs

All the inference tasks listed above are potentially very useful for CRNs. HMMs have been extensively used in CRNs for a wide range of problems. They can be used for spectrum prediction, PU detection, signal classification, etc. (He et al. 2010). A potential drawback when using HMMs is that a training sequence is needed, with the training process being potentially computationally complex. Other AI techniques such as genetic algorithms are used to improve the model training efficiency (Rondeau et al. 2004b).

6.1.3 Application of HMMs for routing

HMMs have been, or can be, used for solving various modeling, planning and prediction tasks that relate to cognitive routing in CRNs. In particular, HMMs have been popularly used for *spectrum occupancy prediction* (Akbar and Tranter 2007; Park et al. 2007). Akbar

and Tranter (2007) utilized HMM models for predicting spectrum occupancy of the licensed radio bands for CRNs in their proposal of an HMM-based DSA algorithm. Choi and Hossain (2011) proposed a channel learning scheme based on HMM and also proposed a partially observable Markov decision process (POMDP) based framework for channel access to opportunistically exploit frequency channels a primary network operates on. Choi and Hossain (2013) have another follow up work on using HMM to model the traffic pattern on PUs.

6.1.4 Pitfalls and challenges

HMM makes many big assumptions that can be problematic for CRNs. The Markovian assumption, fundamental to HMM, presupposes that the emission and transition probabilities depend only on the current state (or that the system is memoryless). While this simplifies analysis, the modeled system in CRNs may have significant memory. HMM is a supervised learning technique which limits its usage in scenarios where the environment is unknown and no training sequence is available. Furthermore, efficient performance of HMM requires training with a sufficiently large training sequence which can be computationally complex. Even after HMM has learnt of the model, the underlying assumption is that the system is stationary and will not change is not matched by real systems which are dynamic and non-stationary. In CRN environments, we often have to work in unfamiliar environments in an online fashion which does not match the niche of HMM entirely.

6.2 Reinforcement learning

In this section, we will discuss initially the relationship of RL with MDPs, and will then discuss major categories of RL algorithms. We will then study specific RL techniques including Q-learning and Learning Automata. We will then discuss some central issues in RL and will finally discuss applications of RL in CRNs and for routing.

6.2.1 Relationship with MDPs

An interesting way to conceptualize RL is to think of it as a simulation-based technique for solving large-scale and complex MDPs. We refer to Sect. 4.2 for an earlier discussion on the relationship between MDPs and RL. We also discussed in Sect. 4.2 that classical DP techniques are ineffective at solving large-scale complex MDPs (Gosavi 2009; Szepesvári 2010). Practical RL algorithms that can deal with large-scale complex MDPs (having large state and action spaces) essentially bank upon two key ideas: firstly, to use samples to compactly represent the dynamics of the control problem, and secondly, to use powerful function approximation methods, including bootstrap methods that build estimates on other estimates, to compactly represent value functions (Talk on Deconstructing Reinforcement Learning' by Richard Sutton at ICML 2013; Szepesvári 2010). It has been stated that understanding the interplay between dynamic programming, samples and function approximation is at the heart of design, analysis and application of modern RL algorithms (Szepesvári 2010; Busoniu et al. 2011). We note here that RL is also known by alternate monikers such as neuro-dynamic programming (NDP) (Bertsekas and Tsitsiklis 1995), adaptive dynamic programming (Bertsekas 2011; Gosavi 2009), and approximate dynamic programming (Powell 2007).

6.2.2 Categories of RL algorithms

It is noted that RL is best understood as a class of learning problems rather than as a fixed set of algorithms or techniques. There is great diversity in the various approaches taken by different RL algorithms and techniques.

Most RL algorithms can be broadly classified into being either *model-free* or *model-based* (Sutton and Barto 1998). A model intuitively is an abstraction that an agent can use to predict how the environment will respond to its actions: i.e., given a state and the action performed therein by the agent, a model can predict the (expected) resultant next state and the accompanying reward. We will be mostly interested in stochastic models which can predict probabilistically possible next states and rewards given the current state and action.

In the *model-based approach*, the agent builds a model of the environment through interaction with it typically in the form of a MDP analogous to the approach taken in adaptive control (Kumar 1985). With a model in hand, given a state and action, the resultant next state and next reward can be predicted allowing *planning* through which a future course of action can be contemplated by considering possible future situations before they are actually experienced. Based on the MDP model in the model-based approach, a planning problem is solved to find the optimal policy function with techniques from the related field of *dynamic programming* (Russell and Norvig 1995; Sutton and Barto 1998). Commonly used algorithms used to solve MDPs include the celebrated dynamic programming algorithms of value iteration (Bellman 1957) and policy iteration (Howard 1960) (discussed earlier in Sect. 4.2).

In the *model-free approach*, on the other hand, the agent aims to *directly* determine the optimal policy by mapping environmental states to actions without constructing a MDP model of the environment. Early RL systems were explicitly trial-and-error learners and were generally devoid of planning. Popular model-free RL techniques include temporal difference (TD) learning (in which a guess is updated on the basis of another guess) and Q-learning (Sutton and Barto 1998). Modern reinforcement learning spans the whole gamut of approaches from low-level, trial-and-error learning to high-level, deliberative planning (Sutton and Barto 1998).

RL tasks can be also be categorized into two types depending on whether the decision making tasks are sequential or not. In *non-sequential tasks*, expected immediate payoff is more important, and the objective is to learn a mapping from situations to actions that maximizes the expected immediate payoff. Such learning has been studied extensively in the field of *learning automata*. In *sequential tasks*, the objective now is to maximize the expected long-term payoffs. Sequential tasks are considered more difficult since the chosen action may influence future trajectory of situations and payoffs. There are two major challenges in sequential RL: the first challenge is the *temporal credit assignment problem* which arises from the fact that the reward in a sequential RL tasks is received in response to a series of actions (or moves) which makes it challenging to attribute credit appropriately to particular actions; the second challenge is the *structural credit assignment problem* in which the problem space is too large for complete exploration and generalization must be performed so that the learning agent may guess about new situations based on previous experience in similar situations. Sequential RL learning has been the subject of fields such as *dynamic programming* (DP)—where the environment is completely known and the state/action spaces are not too large—and *approximate dynamic programming* (ADP) for incompletely known environment or prohibitively large state/action spaces.

6.2.3 Major RL techniques

We can broadly categorize RL techniques into two main categories of *value iteration* and *policy iteration* techniques. In *value iterating* learning techniques, the optimal policy is calculated on the basis of optimal value function calculated as described in Sect. 4.2.1. In *policy iterating* learning techniques, on the other hand, the learning is directly in the policy space as described earlier also in Sect. 4.2.1. We will present representative techniques that belong to these two categories next. In particular, we will discuss *Q-learning* as an example value-iterating model-free technique, and will then discuss *learning automata* as an example technique that is policy-iterating.

Q-learning Q-learning, proposed by Watkins and Dayan (1992), is a popular value-iteration model-free technique with limited computational requirements that enables agents to learn how to act optimally in controlled Markovian domains. The implication of being model-free is that Q-learning does not explicitly model the reward transition probabilities of the underlying process. Q-learning proceeds instead by estimating the value of an action by compiled over experienced outcomes using an idea known as *temporal-difference (TD) learning*.

The TD learning idea has been referred to as the central key idea in the theory of RL. TD learning combines ideas from *Monte Carlo* (*MC*) *methods* and dynamic programming (DP). Like MC methods, TD method is a simulation based model-free method that can learn directly from raw experience without a model of the environment's dynamics. Like dynamic programming, TD method used bootstrapping to update estimates based in part on other learned estimates. The concepts of TD, DP and MC are central recurring themes in RL literature.

Q-learning proceeds by incrementally improving its evaluations of the *Q-values* that incorporate the quality of particular actions at particular states. The evaluation of the action-value pair, or the Q-value, is done by learning the *Q-function* that gives the expected utility of taking a given action in a given state and following the optimal policy thereafter. The Q-function is defined as follows:

$$Q(s,a) = \sum_{s'} P_a(s,s') \left(R_a(s,s') + \gamma V(s') \right) \tag{3}$$

The array Q is updated directly with experience in the following way. The core of the update algorithm below is based on value iteration (discussed earlier in Sect. 4.2.1). r_{t+1} is the reward observed after performing a_t in s_t, and where $\alpha_t(s,a)$ $(0 < \alpha \leq 1)$ is the learning rate (may be the same for all pairs). The discount factor γ $0 \leq \gamma \leq 1)$ trades off the importance of sooner versus later rewards. The Q-function estimate is refined in every learning step and a new policy is generated on its basis which drives the next action to execute.

$$\begin{aligned} Q_{t+1}(s_t,a_t) = & \underbrace{(1-\alpha_t(s_t,a_t))}_{\text{inverse learning rate}} \times \underbrace{Q_t(s_t,a_t)}_{\text{old value}} \\ & + \underbrace{\alpha_t(s_t,a_t)}_{\text{learning rate}} \times \underbrace{(r_{t+1} + \gamma \max_a Q_t(s_{t+1},a))}_{\text{learned value}} \end{aligned} \tag{4}$$

Q-learning in its simplest setting stores data in tables. This quickly becomes impractical for complex systems. In such cases, Q-learning can be combined with function approximation: in particular, (adapted) artificial neural networks (ANNs) have been proposed for function approximation for large-scale RL problems (Tesauro 2002). While Q-learning does not systematically handle the tradeoff between exploration and exploitation and relies instead on heuristic explorations, it has been shown that fortunately Q-learning does eventually find the

optimal value of an action [the proof relies on infinitely many observations for every action and state (Watkins and Dayan 1992)]. The Markovian environment of MDPs is crucial for guaranteed convergence, and the convergence guarantee is lost if this assumption is not valid.

In its basic setting, Q-learning is intended for single-agent environments, although *multi-agent Q-learning*, also known as *Q-learning with games*, have also been proposed recently. Multi-agent learning is especially challenging since it operates in non-Markovian environments (as the output of an action no longer only depends on the current state and agent's personal action). As such, the convergence guarantees of MDP do no extend to multi-agent RL environments due to their non-Markovian nature.

Application of Q-learning in CRNs Q-learning is perhaps the most popular model-free reinforcement learning technique which has been applied to CRNs extensively (Bkassiny et al. 2013). For example, Reddy (2008) has used Q-learning for detecting the PU to ensure efficient utilization of spectrum. We refer the interested reader to a survey paper for more details and references (Al-Rawi et al. 2013).

Application of Q-learning for routing Boyan et al. in 1994 proposed their 'Q-routing' algorithm (Boyan and Littman 1994) that learned a routing policy that minimizes total delivery time by learning through experimentation with different routing policies. The presented RL based algorithm had the desirable features that: (1) its learning is continual and online, (2) it uses local information only, and (3) it is robust in the face of dynamic network conditions. This early paper showed that adaptive routing is a natural domain for reinforcement learning. In a follow-up paper (Kumar and Miikkulainen 1997), another adaptive routing algorithm DRQ-routing was presented which combines Q-routing and dual reinforcement learning which learns a better routing policy (better average packet delivery time at high loads and faster learning of policy) compared to Q-routing due to increased exploration. In another Q-learning based routing work, Zeng et al. (2013) have presented a Q-learning based directional routing and scheduling scheme for green vehicular DTNs optimized for energy efficiency. The proposed algorithm explores multiple possible strategies, and adapts the strategy in an online manner according to the knowledge obtained through prior actions.

Learning automata Learning automata (LA) is an AI technique that subscribes to the policy iteration paradigm of RL (Nicopolitidis et al. 2011; Akbari Torkestani and Meybodi 2010a, b; Vasilakos and Papadimitriou 1995). In contrast to other RL techniques, policy iterators operate by directly manipulating the policy π. Another example of policy iterators are evolutionary algorithms such as genetic algorithms (which we will discuss later in Sect. 6.5).

A learning automaton is a finite state machine that interacts with a stochastic environment and attempts to learn the optimal action (that has the maximum probability to be rewarded) offered by the environment so that it can ultimately choose this action more frequently than other actions. Since wireless networks operate in dynamic time-varying environments with possibly unknown characteristics (e.g., variable link qualities, dynamic topologies, changing traffic patterns, etc.), the application of LA techniques for building adaptive protocols in such networks is particularly appealing. In this regard, LA has been used in the design of wireless MAC, routing and transport-layer protocols (Nicopolitidis et al. 2011).

Application of LA to routing and CRNs We will now present some *example LA based routing protocols*. Misra and Oommen (2005) presented a learning-automata based solution to the dynamic shortest path problem in which there are continuous probabilistic updates in the cost of edges of a single-source stochastic graph topology. Torkestani et al. have proposed using LA for multicast routing in mobile ad-hoc networks or MANETs[10] to find

[10] MANETs share an important characteristic with CRNs in that both of them have highly dynamic topology. The dynamically changing topology in MANETs is due to node mobility while in CRNs it is due PU arrivals.

routes with expected higher lifetimes through prediction of node mobility (Akbari Torkestani and Meybodi 2010a). Another LA-based distributed broadcast solutions can be seen at Akbari Torkestani and Meybodi (2010b).

6.2.4 Central issues in reinforcement learning

Some pressing issues in RL research have been highlighted in Kaelbling et al. (1996) to be: (1) trading off exploration and exploitation, (2) learning from delayed reinforcement, (3) making use of generalization, (4) dealing with multiple agent reinforcement learning, (5) constructing empirical models to accelerate learning, and (6) coping with hidden state. Out of these problems, the issues of exploration and exploitation and that of multi-agent reinforcement learning are most relevant to our work, and we discuss these next.

Issue of exploration and exploitation Exploitation would entail favoring immediate payoff while exploration would require tolerating momentary *regret* of not using the best currently known policy for the opportunity of potential information about better policies. It should be apparent after some reflection that neither exploration nor exploitation can be pursued exclusively without failing at the task of selection of the optimal action. The tension between exploitation and exploration is typified in the so-called *multi-armed bandit problems*. The k-armed bandit problem is the simplest possible RL problem (Kaelbling et al. 1996) and represent an MDP with a single state (see Fig. 4) in which k actions are available. The problem is called a k-armed bandit in a metaphorical reference to predicament of a gambler who must select from k slot machines, colloquially called a 1-armed bandit, in a casino. Interestingly, the conflict between delayed versus immediate gratification is a dilemma unique not only to RL, the conflict it arises can be experienced in our own humanness.[11] Fortunately, a method has been devised by Gittins in 1979 for optimally solving the exploration and exploitation tradeoff for the simple case of k-armed bandit problem (Gittins 1989) assuming a discounted expected reward criterion. This method entails providing a dynamic 'allocation index' to each action for each step in k-armed bandit problems. Gittins showed that it is guaranteed that choosing the action with the largest index value will lead to optimal balance between exploration and exploitation (Gittins 1989).

For the general case of MDPs, the optimal balance between exploration and exploitation is known to be an intractable problem to solve (Sutton and Barto 1998). Therefore, a lot of interest has focused on development of heuristic or approximate methods to handle the trade-off between exploration or exploitation. To manage the exploration or exploitation dilemma, the ϵ-*greedy* strategy is to select the greedy action (one that exploits prior knowledge and provides the best value) all but ϵ of the time, and to select an action randomly for the remaining ϵ of the time. The value ϵ ranges between 0 and 1 and it is possible to change this value over time. Intuitively, it would be prudent for an agent to be more of an explorer initially (by having a higher ϵ) since it has no knowledge to exploit it. With passing time, as good states and actions are learnt, the agent can benefit more by being an exploiter and taking the greedy approach (with smaller ϵ) which chooses good actions more often. It makes intuitive sense that during explorations, the choice of actions is not completely random but based on some estimation of their potential value. In this regard, a *soft-max action selection* technique can be used which uses the *Gibbs or Boltzmann distribution* for selecting the action to explore where the probability of selecting an action is proportional to its perceived value (e.g., its

[11] It has been said by a mathematician Peter Whittle that "bandit problems embody in essential form a conflict evident in all human action: information versus immediate payoff".

Q-value). We note here the question of exploration versus exploitation is central not only to reinforcement learning, but also to genetic algorithms, and to evolutionary algorithms in general (Črepinšek et al. 2013).

Multi-agent reinforcement learning Multi-agent RL are more challenging than single-agent RL problems mainly since the Markov property does not hold in such environments as an agent's reinforcement depends not only on its current state but also on the action taken by the other agents. Accordingly, convergence guarantees that apply to MDP RL tasks do not extend in such non-Markovian multi-agent RL settings. Learning automata based tools have been quite popular in multi-agent RL environments. A detailed survey of multi-agent reinforcement learning algorithms is presented in Busoniu et al. (2008).

Multi-objective reinforcement learning Multi-objective (or, multi-criteria) reinforcement learning has also been proposed by Gábor et al. (1998) to apply RL to problems with multiple objectives and where the RL reward is a vector rather than a scalar. Zheng et al. (2012) have proposed a multi-objective RL-based (more specifically, Q-learning based) routing algorithm for CRNs that integrates multiple desirable routing performance metrics—they have proposed to minimize transmission delay under a constraint on packet loss rate—while taking into account network dynamics triggered by PU arrivals.

6.2.5 Application of RL to CRNs

RL methods are especially appropriate for online control of CRN parameters where the optimal behavior for the dynamic environment is not known a priori due to the unavailability of a teacher or trainer. Since CRs often have to work in unknown environments, RL seems to be a promising solution to the various learning problems in CRNs and it looks set to become a popular tool for future CRN designers. Applications of reinforcement learning to CRNs in general are explored in Yau et al. (2010) and Di Felice et al. (2010) while RL techniques for context awareness and intelligence in wireless networks are reviewed in Yau et al. (2012). The main benefits of applying RL in CRNs are adaptivity and network awareness while the main drawback is slow convergence (Di Felice et al. 2010).

6.2.6 Application of RL to routing

There have been many applications of RL techniques for the routing problem. We have already seen applications of the specific RL techniques of Q-learning and Learning Automata to routing earlier in this section. In addition, techniques likes the multi-armed bandit problem (covered next under the heading of online learning algorithms) are also closely related to RL and have been applied in the context of routing. Xia et al. (2009) have proposed RL based spectrum-aware routing algorithms inspired by Q-learning and dual reinforcement learning for CRNs. In another work, Bhorkar et al. (2012) have proposed a distributed adaptive RL-based opportunistic routing algorithm (d-AdaptOR) for ad-hoc networks which is optimal with respect to the average expected per-packet reward and does not require any explicit knowledge of the network environment. Numerous other works have utilized RL formulation in their routing solution, and the interested readers are referred to the following two papers, and the references therein, for an exhaustive treatment of RL applications for routing in wireless networks. In the first reference (Di Felice et al. 2010), existing RL schemes in the context of ad-hoc CRNs are surveyed and modifications are proposed from the viewpoint of routing and link-layer spectrum-aware operations. The second reference (Al-Rawi et al.

2013) presents a detailed survey of applications of RL to routing in distributed wireless networks.

6.2.7 Pitfalls and challenges in applying RL in CRNs

We have noted earlier that RL techniques such as Q-learning can solve MDPs without requiring an explicit specification of the transition probabilities, or the reward function, in contrast to classical dynamic programming solutions of value and policy iteration. A common conception of RL is that it obviates the need to explicitly specify the transition probabilities by accessing it through a simulator that typically is restarted from a uniformly random initial state many times (Busoniu et al. 2011). While such a conception of RL is useful in the AI/control community where it is feasible to perform such simulations repeatedly (resetting the state whenever desired), it is not as applicable in wireless communications where the learning has to be performed online without any generative model that can be reset (e.g., it is not feasible to reset the channel state or the number of backlogged packets on demand).

The main drawback of RL techniques, especially in the wireless communications context applicable to CRNs, is their slow convergence. Although wireless networks are highly dynamic and non-stationary, RL techniques only have asymptotic guarantees of convergence to optimal policy (i.e., only if each action is executed in each state an infinite number of times, which is clearly not realistic in practical systems) even if we make the simplifying (non-realistic) assumption of stationary networks. The non-stationary nature of wireless networks also implies a shifting optimal policy that changes with network dynamics thus making successful implementation of RL techniques in wireless networking challenging. Unfortunately, in realistic wireless networking scenarios, learning the network dynamics through RL techniques is not straight forward and can lead to fairly complex implementations (Fu and Van Der Schaar 2010). More research is needed to develop RL techniques that are amenable to efficient implementation in the highly dynamic CRNs.

The utility of RL may also depend on environmental dynamics: e.g., pattern and structure of PU activity. For a learning process to be useful, there should be enough structure in the observable environment over a suitable time scale. It has been observed that the learning performance of RL is highly correlated with the level of PU activity and the amount of structure in spectrum usage (Macaluso et al. 2013). In particular, RL performs no better than random channel selection for low levels of PU activity (or high Lempel–Ziv complexity in channel utilization).

6.3 Online learning algorithms

Online learning algorithms address the task of performing sequential decision-making online with partial information. For example, consider the problem of determining what route to use to drive to work everyday in an uncertain environment where the congestion pattern on the various paths is both stochastic and unknown (Blum and Monsour 2007). The basic setting is we have a space of N actions, from which the algorithm chooses an action (in our example, selecting the route to take) one time step after the other. The environment then makes its 'move' (in our example, by setting the path congestions for that time step). The algorithm then incurs the 'loss' for its action chosen (in our example, this is how long the route took). Online learning algorithms aim to perform well in such tasks of repeated decision making. While our example relates to routing in a transportation network, it is analogous and directly extensible to the problem of routing in a CRN. Online learning algorithms typically aim to provide efficient online solutions to complex problems (typically NP-hard) that are unlikely

to admit algorithms with provably good worst-case performance. Due to the intractability of the problem, the problem has to be addressed through heuristics whose performance can be difficult to predict in advance.

A key technique for analyzing the performance of online learning algorithms is *regret analysis* which quantifies the suboptimality of the online learning algorithm compared to the optimal fixed policy. This captures our sentiment that we want our sophisticated online algorithm (which may be choosing different actions at different times) to be at least as good as some simple fixed alternative policy λ that sticks with just one action at the time of all decisions—this will minimize our *regret* of not choosing the alternative policy λ. Regret is formally defined to be the difference between the loss of our learning algorithm and the loss using the alternative policy λ. This regret is more properly called *external regret* when the alternative policy is a *static* policy (i.e., a policy of performing the same action in all time steps). External regret allows us a general methodology for developing online algorithms whose performance is comparable to that of an optimal static online algorithm. Stronger notions of regret include *internal* or *swap regret* which allow comparison of online action sequences in which every occurrence of a given action i is changed by an alternative action j (Blum and Monsour 2007). There has been a lot of work by the learning theory and the game-theory communities in the area of no-regret learning, and online learning algorithms have been shown to have strong performance guarantees (Blum and Monsour 2007) with decision-making algorithms (such as the *weighted majority algorithm* (Littlestone and Warmuth 1989)] available that approach zero regret even against a fully adaptive adversary.

The multi-armed bandit (MAB) problem, discussed earlier in Sect. 6.2 as a special case of MDPs that has a single state with certain actions and associated rewards, is a powerful online learning tool. The MAB formulation is a widely applied tool in the fields of manufacturing, industry, decision theory, and is applicable wherever there is choice between alternatives in an uncertain settings. The MAB problem deals with a player (known as a bandit) who has to choose one or more resources (known as arms) amongst several plausible candidates each having unknown statistical properties with the aim of maximizing some notion of long-term reward. In bandit problems, partial information in the form of only the payoff of the selected action is observed. This partial information setting is most relevant to the routing problem in CRNs. The bandit setting is distinguished from the setting of online learning with full information in which the action chosen by the adversary is revealed after each time step. A MAB is said to have *rested arms* if the state of the stochastic process representing an arm stays frozen unless played. A MAB has *restless arms*, on the other hand, when the state of the stochastic process representing all arms continues to evolve (accordingly to a possibly different law) regardless of which arm is played per the player's action. From the perspective of CRs, the most important family of bandit problems is the restless MAB problem since the rewards associated with various routes cannot be predicted unless those routes are adopted and these rewards do keep changing.

Online MAB problems can be thought of as a repeated game between the player and the environment. There are two fundamental formalization of the online MAB problems relevant to the routing problem depending on the assumed nature of the reward process: *stochastic* and *adversarial*. In *stochastic bandit problems*, the rewards are assumed to be sampled from an unknown distribution. The classic UCB algorithm (Auer et al. 2002a) is essentially optimal for such a setting. In *adversarial bandit problems*, also known as non-stochastic bandit problems, the rewards are assumed to be chosen by an adversary. The stochastic model of the reward process lends itself to a framework where the algorithm are designed to perform well in expectation for the average case (in the stochastic sense) while the adversarial (or the non-stochastic) model of the reward process allows the design of algorithms to be robust in

the worst-case sense. If it is assumed in an adversarial setting that the rewards do not depend on this history of arms selected so far then we have an *oblivious adversary*, otherwise (i.e., if the rewards do depend on the history of selected arms), we have an *adaptive adversary*. The adaptive adversary makes stronger assumptions and it tries to make the online algorithm as poorly as possible. The Exp3 algorithm (Auer et al. 2002b) is an effective algorithm for the adversarial bandit problem. Adversarial online bandit algorithms have been proposed both for routing in oblivious settings (Bansal et al. 2003) and in adaptive settings (Aspnes et al. 1997). While adaptive algorithms generally return 'better' performance, oblivious algorithms have the advantage of lower implementation overhead and can be computed in advance. In addition to stochastic and non-stochastic bandit problems, we also have Markovian bandit problems in which it is assumed that the bandit arms are associated with K Markov processes each with its own state space; for the Markovian bandit problem, the use of Gittin's indices is an effective algorithmic solution.

6.3.1 Application of online learning algorithms in CRNs

The MAB formulation has also been extensively applied in CRNs for problems such as opportunistic spectrum access (in which a node is considered as a bandit with the set of channels that be accessed considered as the bandit's arms) (Xu et al. 2013). In general, the framework of 'restless bandits' fits best with the dynamic environment of CRNs in which each arm may correspond to links on different frequency channels and the PU activity on each channel can change even if it that link is not chosen (Goldsmith et al. 2012). The restless bandit approach has been adopted in literature for spectrum sensing but no work using this approach has been proposed for routing in CRNs according to the best of our knowledge. In other work, Han et al. proposed a using the solution concept of correlated equilibrium for opportunistic spectrum access in CRNs using a distributed no-regret online learning algorithm. It was shown in their work that their correlated equilibrium based solution returns fairer results with better performance (Han et al. 2007).

6.3.2 Application of online learning algorithms for routing

While the centralized problem can be directly formulated as the classic MAB problem with each path from the source to destination being considered as an arm. After such a formulation, standard MAB solutions can be applied to yield centralized learning of the shortest path in a source-routing setting. The drawback of such a naive approach is poor performance with the regret growing linearly with the number of paths and thus exponentially with the network size (in terms of the number of edges) and the difficulty of adopting a distributed learning approach (MAB policies typically rely on the knowing the number of times an arm has been played to balance the exploration/exploitation tradeoff; this information is not available to individual nodes in a distributed setting). Tehrani and Zhao (2013) have proposed a distributed online learning based shortest path algorithm, which is called distributed Bellman–Ford with learning (DBFL) which utilizes dependencies between paths sharing common edges to solve the problem of regret growing exponentially with the number of edges and also local information exchange to detect the least traversed edge which is then explored. The DBFL algorithm achieves regret logarithmic in time and polynomial in network size. To reiterate the DBFL algorithm is distinct from conventional Bellman–Ford algorithm in that the edge weights are not static and known a priori but are considered as random variables with unknown distributions which have to be learnt. In other works, Awerbuch and Kleinberg

(2008) have formulated the problem of determining a sequence of routing paths in a network with unknown link delays varying unpredictably over time as a generalization of the online MAB problem. The sequential decision-making under partial information in this MAB problem is handled through the framework of a repeated game with two players (algorithm and adversary) interacting over time. Avramopoulos et al. (2008) have proposed using online learning algorithms as a framework for adding adaptivity to routing decisions in realistic Internet-like environments and still return with stable outcome that has a small optimality gap with respect to the network wide optimum. Recently, Bhorkar and Javidi (2010) have presented a no-regret routing algorithm for wireless ad-hoc networks.

6.3.3 Pitfalls and challenges

MAB problems essentially deal with single user problems, only incorporating the agent's interaction with the environment and not with other users, thus limiting its use to single-user environments. Since CRNs are characterized by the interaction of multiple agents, MABs cannot model the multi-user interaction that may be present in CRNs. Also, existing works in CRN literature using MABs (e.g., Gai et al. 2012) have mostly assumed i.i.d. rewards. The problem of getting regret results for MAB with restless Markovian rewards is still an open research issue. The restless MAB problems in general are known to be computationally intractable being in the class of PSPACE (or polynomial space problems) (Papadimitriou and Tsitsiklis 1999). PSPACE-problems, include all decision problems that can be solved in a polynomial amount of space by a Turing machine, are highly challenging to solve optimally.

To summarize information covered till now about common analytical models of routing, we refer the reader to Table 3.

6.4 Learning with game theory

While game theory is essentially concerned with the decisions made by individuals in their interactions with other decision makers and their environment, researchers have long recognized the need to guide future decisions from the history of past experience. There is a lot of work on the important relationship between game theory and learning (Fudenberg 1998). A branch of game theory known as 'learning game theory' studies the dynamics of individuals who repeatedly play a game, and adjust their behavior over time as a result of their experience (through, e.g., reinforcement, imitation, or belief updating) (Izquierdo et al. 2012).

It is worth highlighting the work that has been done in identifying the similarities between inference and learning in the fields of machine learning and game theory (Rezek et al. 2008). In the field of game theory, learning is used implied to mean inference of the correct strategy to play against an opponent within a dynamic game (repeated game, stochastic game, or evolutionary game). Some of the models that have been used for learning in game theory include reinforcement learning, learning by imitation, myopic response, fictitious play, and Bayesian learning (see Sect. 6.7). As examples, we discuss myopic response, fictitious play, and Q-learning with game theory.

Myopic response In myopic adaptation, a SU does not update its belief about other SU's action but instead maximizes the utility based on the observation of other SU's actions in the previous round of the game.

Fictitious play The main idea in fictitious play is that each player would choose their best strategy in each period, based on the predicted strategy that each opponent player would choose in that period, to maximize expected payoff.

Table 3 Analytical formulation of the routing problem in wireless networks

Analytical model used	References	Comments
Markov decision process		
MDP	Lott and Teneketzis (2006)	MDP-theoretic formulation for the opportunistic routing approach
DEC-MDP	Friend (2009)	Optimal 'minimum-expected-cost' routing in cognitive networking setting
POMDP	Nurmi (2007)	Models routing as a sequential decision making processing with incomplete information using the POMDP framework
Game theory		
Static game	Urpi et al. (2003)	Static Bayesian game is used to model routing
Repeated game	Srinivasan et al. (2003)	Repeated prisoners' dilemma is used to model the conflict nodes face (i.e., should the node forward cooperatively or drop selfishly?)
Repeated game	Urpi et al. (2003)	Infinitely repeated games are used to model routing
Dynamic Bayesian game	Nurmi (2004)	Dynamic Bayesian games are used to incorporate non-simultaneous decision making and incorporating history information into the decision making process
Reinforcement learning		
Q-learning	Boyan and Littman (1994)	Proposed a Q-learning based distributed Q-routing scheme in which a RL module is embedded in each routing node
Multi-armed bandit	Avramopoulos et al. (2008)	Formulated routing as a regret minimizing online MAB problem and proposed a bandit-based routing algorithm which used the Exp3 (Auer et al. 2002b) learning algorithm

Q-learning with game-theory Although Q-learning in its basic form is used in a single-agent RL setting, it can also be used in a multi-agent RL setting. In this multi-agent Q-learning algorithm, the Q-value is updated with the future payoff so that each agent can observe and estimate the payoff for using a particular strategy (not only for itself but also for the other players).

6.4.1 Application of learning with game theory in CRNs

As an example of the use of myopic adaptation, Meshkati et al. (2007) has used this approach as the learning approach in a power control game. It was shown that this approach converges to a Nash equilibrium but with a lower system performance than the collaborative case. As an example of fictitious play, Shiang and Van Der Schaar (2008) have presented a multi-

agent learning approach for delay-sensitive resource management in multi-hop CRNs. As an example of using Q-learning with game theory, Hu and Wellman (2003) have presented a Nash Q-learning algorithm for multi-agent RL setting based on the concept of stochastic games. A detailed survey of strategic learning in CRNs, and various spectrum access games, is presented in Van der Schaar and Fu (2009). In another related work, Khan et al. (2012) have studied the game dynamics and the cost of learning in heterogeneous 4G wireless networks. The authors proposed a new RL based learning scheme named 'cost-to-learn' that incorporated the cost of switching to a new channel and a new action.

6.4.2 Application of learning with game theory to routing

There is limited work on using learning based game theory in CRNs. As an example of work in this domain, a distributed routing framework based on multi-stage fictitious play learning has been proposed by Zhu et al. (2010) for the dynamic interference minimization routing game. Using learning-based game theory for routing is a promising direction in need of further exploration.

6.4.3 Pitfalls and challenges

The aim of the approach of learning with game theory is for agents to use learning techniques to reach the same strategy that rational players would use in a game of complete information. They key challenge in learning with game theory in CRNs is incomplete information and observability of the players, their strategies, and the environment. Learning in multi-agent environments is in general a very hard problem due to the inherent dynamism in the system which arises as all agents continually adapt as they learn.

6.5 Learning with metaheuristic algorithms

Most of the optimization problems of interest in CRNs are difficult to solve optimally in reasonable time thus motivating interest in heuristic optimization techniques that provide 'good enough' solutions in reasonable time. Metaheuristics are generic design patterns or algorithmic ideas based on heuristics that can be applied to a broad range of optimization and learning problems (Glover and Kochenberger 2003). There are diverse number of metaheuristic techniques such as simulated annealing, swarm optimization, hill climbing, that can be applied in CRNs but we will, in the interest of brevity, focus on the two metaheuristic techniques that have been widely applied in CRNs: *genetic algorithms* and *ant-colony-optimization*.

6.5.1 Genetic algorithms

A genetic algorithm (GA) is a particular class of evolutionary algorithm which uses techniques inspired from evolutionary biology (Goldberg and Holland 1988).[12] Evolutionary algorithms are a set of machine learning techniques that aim to imitate the robust procedures and structures that various biological organisms have used for adaptation and learning in their evolution. Evolutionary algorithms are similar to reinforcement learning algorithms in that they also depend on exploration and exploitation (Črepinšek et al. 2013). GAs constitute a very general meta-heuristic technique which can be thought of as the sledgehammer of

[12] We note here that GA is not the only biology-inspired technique and there are a variety of other biology-inspired optimization techniques (Liu et al. 2015; Song et al. 2014).

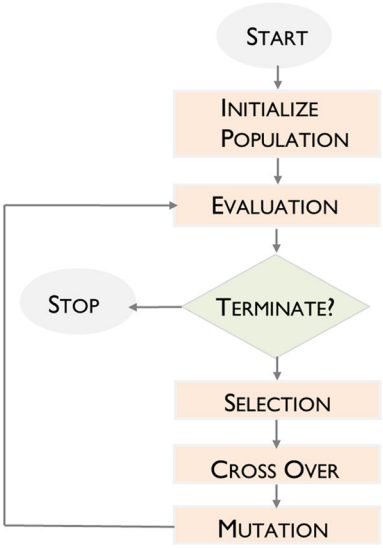

Fig. 7 A flow chart of a typical genetic algorithm

the craft of algorithms, much like the technique of ANNs, which can be readily invoked when more specialized methods fail. GAs are very good at navigating through huge search spaces to heuristically find near optimal solutions in quick time. More specifically, GA fundamentally relies on the genetic operators of random *mutation* and recombination through *crossover* to improve the current solution. Apart from these operators, the design of GAs also includes other crucial components such as population initialization, genetic representation, fitness function, and a mechanism for selection.

The flow chart of a typical GA is shown in Fig. 7. GAs operate by initially defining a *population* of candidate solutions (called *individuals*). Individuals are encoded in an abstract representation known as a *chromosome* (which may be problem specific although representation in strings of 1s and 0s is common). Various evolutionary techniques (mutation, crossover, etc.) are applied on the population thereafter in a computer simulation. The evolution can start from a population of completely random individuals and can evolve to better solutions through *survival of the fittest* after application of genetic operators in every *generation*. In every generation, multiple individuals are stochastically selected from the current population with fitter individuals more likely selections and are genetically modified (mutated or recombined) to form the next generation of the population. The usage of genetic operators and stochastic selection allow a gradual improvement in the 'fitness' of the solution and allow GAs to keep away from local optima. The GA completes execution when the population (representing the solution) meets a predefined fitness condition or when a predefined number of iterations have been performed.

We note here that evolutionary algorithms, and by extension GA, are related to reinforcement learning in that both depend on exploration and exploitation (Črepinšek et al. 2013). Evolutionary algorithms based learning also illustrates how learning can be viewed as a special case of optimization. These algorithms pursue the 'optimization problem' of finding the optimal hypothesis according to a predefined fitness function (Mitchell 1997). With the insight that learning is ultimately related to optimization, we can apply other optimization and heuristic techniques to machine learning problems. For a discussion about heuristic

optimization techniques (such as simulated annealing, tabu search, hill climbing) and their application to CRNs, readers are referred to He et al. (2010).

Application of GAs to CRNs GAs have been extensively deployed in CRNs and more generally in wireless networks (Mehboob et al. 2014). Apart from playing a direct role in routing, GAs can also be used for bandwidth allocation for paths and for QoS based routing as used by Pitsillides et al. (2000) in their work. The benefits of GA is that it is conceptually easy to understand, inherently amenable to parallel solutions and therefore can be easily distributed. GA also lends support to multi-objective optimization and performs well in noisy environments. An early application of GA techniques to CRNs is documented in a paper authored by Rondeau et al. (2004a). This paper presented the adaptation mechanism of a cognitive engine implemented by the authors which used GAs to evolve a radio's parameters to a set of parameters that optimize the radio for the user's current needs. This paper also proposed a GA approach, called the 'wireless system genetic algorithm' (WSGA), to realize cross-layer optimization and adaptive waveform control (Rondeau et al. 2004a). Genetic algorithms have also been used for building distributed *parallel* solutions, using the technique of *island genetic algorithms*, to the problem of channel allocation in cognitive networks (Friend et al. 2008). An island GA divides the overall population into subpopulation known as islands which interact by migrating individuals to the other island. More details of parallel genetic algorithms, and of general parallel metaheuristics, can be found in the survey Alba and Troya (1999) and Alba (2005), respectively.

Application of GAs to routing Ahn and Ramakrishna (2002) proposed tackling the shortest path routing problem through GAs. The paper discussed the issues of path-oriented encoding, and path-based crossover and mutation, which are relevant to the routing problem (Ahn and Ramakrishna 2002).

Pitfalls and challenges There are numerous challenges in devising an appropriate GA based solution for optimization problems in CRNs including suitable definitions of population size and evaluation function (which is typically non-trivial to define). The policy for deletion and operators of mutation/crossover also have to be suitably defined. A big challenge with using GAs in CRNs is the risk of slow convergence and settling for local minimas especially with inappropriate choice of model parameters. In light of potentially slow convergence, it is very important to devise a suitable termination criteria which defines a 'good enough' solution.

6.5.2 Ant colony optimization

While typical 'shortest path' routing protocols may have significant computational and message complexity, the humble biological ants, in a marvel of nature, are able to shortest routes to food sources in the dynamics of ant colony with extremely modest resources. Ant colony optimization can be used to solve shortest-path problems in the following way. Initially, ants wander around in search of food, but when food is found, ants return to their colony while laying down a pheromone trail that is then used by subsequent ants who follow this trail instead of wandering randomly. The pheromone trail that takes the ants to the food successfully via the shortest path is continually reinforced and is kept updated by the fact that pheromone trails starts to evaporate over time.

Application of ACO to CRNs As a form of an evolutionary algorithm, ACO is suited for heuristic control of dynamic environments. ACO can be used, like GA, for developing cog-

nitive radio engine (Zhao et al. 2012) and for various other tasks in CRNs such as routing (Li et al. 2009).

Application of ACO to routing With biological ants well-known for self-learning routes to food sources in dynamic environments robustly and adaptively, it is natural to explore ACO for the problem of routing in the context of networking. A lot of research effort has been focused on imitating the performance of biological ants to produce optimized and efficient distributed routing behavior for mobile ad-hoc networks (Caro et al. 2005) as well as for dynamic CRNs (Li et al. 2009). Jie and Kamal (2014) have also proposed an ACO-based *multi-objective* optimization algorithm that aims to compute optimized multicast trees that minimize the worst-case delay and the number of transmission links while simultaneously maximizing the multicast rate. A detailed description of the design parameters, and possible choices for these parameters, for ACO-based routing protocols is provided in the survey paper of Zheng and Sicker (2013) on biologically inspired algorithms for networking. In a previous work, it has been shown that ACO can outperfrom GA in routing and motion planning especially for dynamic topologies (Purian et al. 2013).

Pitfalls and challenges While ACO is an efficient metaheuristic that is useful for routing in dynamically changing environments such as those present in CRNs, it also has a few disadvantages. It is difficult to theoretically analyze ACO, with the time to convergence being potentially problematic for large problems. Also, since ACO is a metaheuristic technique, there is no guarantees regarding convergence to the globally optimal solution.

6.6 Artificial neural networks

Artificial neural networks (ANNs) are composed of artificial 'neurons' interconnected together in a programming structure that aims to mimic the neural processing (organization and learning) of biological neurons and its behavior (Tsagkaris et al. 2008). More specifically, ANNs involve a network of simple elements that can exhibit complex global behavior determined through: (1) the way these elements are connected together into a network, and (2) the adaptive element parameters which are tuned by a learning algorithm. ANNs are mostly used in supervised learning settings but can also be used in reinforcement learning environments [e.g., it can be used along with dynamic programming (Bertsekas and Tsitsiklis 1995), in what is known as neuro-dynamic programming, to solve RL problems] and in unsupervised learning environments (e.g., a self-organizing map (SOM) is a type of an ANN that works under the unsupervised learning paradigm to produce a low-dimensional map of the input space of the training samples, called a map). ANNs are essentially "a network of weighted, additive values with nonlinear transfer functions" although its coined name seems to elicit a grander impression.[13]

The simplest kind of ANN is a single-layer *perceptron* network which is a simple kind of a feed-forward network (i.e., a network in which connections between the units do not form a directed cycle). In such a network, there exists a single layer of output nodes which is provided the input directly via a series of weights. The sum of the weighted input is calculated at each node to calculate an overall value which is then matched against a threshold (typically 0). If the calculated value is greater than the threshold, the neuron is fired and it takes an activated value (typically 1), otherwise, the neuron takes a deactivated value (typically -1). Despite

[13] It has been claimed that the selection of the name "neural network" was one of the great PR successes of the twentieth century since it sounds much more exciting by eliciting a comparison with an actual neural network (i.e., the brain) (A brief history of neural networks 2013).

having a simple and efficient learning algorithm, single-layer networks are of limited utility since they have limited expressive power (i.e., they can not express complex functions) and can only learn linear decision boundaries in the input. Multi-layer networks, on the other hand, are far more expressive and can represent non-linear functions. In multi-layer ANNs, processing elements are arranged in multiple layers (typically interconnected in a feed-forward fashion) with each neuron in a layer having directed connections to the neurons of the subsequent layer. Such networks have a downside that they are hard to train because of high dimensionality of the weight-space and the abundance of local minima (Russell and Norvig 1995).

ANN is essentially a black-box statistical modeling technique that does not utilize the domain's subject knowledge but learns feature from the data itself. Despite the black-box modeling style of ANN, it is a remarkably versatile tool and applies to a wide range of problems and performs fairly well in general. This has led to John Denker to famously remark that "neural networks are the second best way of doing just about anything." (Russell and Norvig 1995). Notwithstanding this claim, for certain types of tasks (e.g., pattern recognition, speech recognition, etc.), ANN is arguably the most effective learning method known currently (Mitchell 1997). The price of the generality of ANNs, though, can be the need of large amounts of training data and in its greater convergence time.

6.6.1 Application of ANNs to CRNs

ANNs have been successfully applied to various problems in CRNs such as spectrum sensing, spectrum prediction (Tumuluru et al. 2010), and dynamic channel selection (Baldo et al. 2009)—the last two applications being especially relevant to our focused topic of routing in CRNs. For more details about application of ANNs to CRNs, the interested reader is referred to these survey papers (He et al. 2010; Bkassiny et al. 2013; Tsagkaris et al. 2008).

6.6.2 Application of ANNs to routing

Since ANNs are used mostly in supervised learning setting, they are mostly used in tasks that complement routing but there are two notable works in which ANNs have been directly employed for the problem of routing in Ju and Evans (2010) and Barbancho et al. (2006). Barbancho et al. (2006) have proposed a QoS-driven routing algorithm named Sensor Intelligence Routing (SIR) for WSNs. The proposed framework incorporates AI by building an ANN on Kohonen SOMs that is implemented on every node to build a distributed AI-based system. SIR builds up the network backbone by using a modification of Dijkstra's algorithm that connects the base station or root to every other node in the network through minimum cost paths. The edge weight parameter is then defined using the *qos* variable obtained from the output of the SOM ANN. In the SOM employed in this work, the first layer has four input neurons (corresponding to the metrics of latency, throughput, error rate and duty cycle) while the second layer has twelve output neurons forming a 3×4 matrix. In another work, Ju and Evans (2010) have proposed scalable cognitive routing protocol (SCRP) for MANETs. SCRP employs a novel approach of scalable flooding that also uses ANNs to provide knowledge to network nodes about history. The use of ANN in flooding reduces routing overhead significantly since nodes now flood RREQ selectively over links and frequencies that are predicted to be strong using historical information.

6.6.3 Pitfalls and challenges

ANN is mainly a supervised learning technique (although ANNs may also be used for non-supervised learning using SOMs in some settings). The supervised learning model limits the usage of ANNs in CRNs since the environment in CRNs is typically unknown and no training sequence is available. Training an ANN model can take a lot of time and computational resources depending on the network size. It is also possible that ANN may over-train, i.e., overfit the training data, with overfitting one of the most cited problems with ANNs (Zhang 2007). However, this lengthy training provides the benefit of simpler computation of the output with very small overhead. Furthermore, ANN is essentially a black-box modeling technique for capturing the non-linear relationship between the network input and the network performance which does not provide any domain specific insights about the model/network developed.

6.7 Bayesian learning

Bayesian learning can be viewed as a form of uncertain reasoning from observations (Russell and Norvig 1995). Bayesian learning is used to calculate the probability of each hypothesis, given the data, and to make predictions on that basis. It has been shown that the true hypothesis eventually dominates in Bayesian prediction (Russell and Norvig 1995). Bayesian analysis accords significant importance to the *prior distribution* which is supposed to represent knowledge of unknown parameters before the data becomes available. While it is a common assumption that the agent has no prior knowledge about what it is trying to learn, this is not an accurate reflection of reality in many cases. Frequently, an agent will have some prior information, and the learning process should ideally exploit this available information.

While it was noted by Rondeau and Bostian (2009) that little research attention has focused on using Bayesian methods of statistical inference in CRNs, a lot of Bayesian inference based work have recently been proposed for CRNs. *Bayesian networks* can be used for computing how much a set of mutually exclusive prior events contributes to a posterior condition, which can be a prior to yet another posterior, and so on. Bayesian networks can be used for reasoning and for tracing chain of conditional causation back from the final condition to the initial causes (Russell and Norvig 1995). Previous work on using Bayesian networks for reasoning in CRNs has been summarized in Adamopoulou et al. (2008).

Parametric models (such as the Gaussian distribution, k-means, HMM, etc) typically assume some finite set of parameters θ and assume that θ captures everything there is know about the data. Non parametric models (such as the *Bayesian non-parametric* (*BNP*) *models* Gershman and Blei 2012), on the other hand, do not assume that the data distribution can be explained on the basis of a finite set of parameters—instead, an infinite dimensional set of parameters θ, envisioned as a function,[14] is assumed. There also has been increasing interest in applying BNP models to CRNs (Saad et al. 2012; Han et al. 2011) due to their desirable characteristics such as its ability to flexibly model an unknown environment with model complexity growing as warranted by new data. The term 'non-parametric' in BNP should not be construed to mean a total lack of parameters but as a lack of non-fixed parameters. Indeed, non-parametric models not only have parameters, but typically have infinitely many parameters. To illustrate, let us take a look at the example of clustering data. The parametric mixture modeling approach requires the number of clusters to be specified a priori, whereas

[14] While the parameter set in BNP models is infinite dimensional, only a finite subset of the available dimensions are used to explain a given finite sample of observations with the dimension size depending, and growing, with sample size.

the non-parametric Bayesian approach is a purely observation-based approach that allows the number of clusters to grow with more data.

BNP models typically exploit in their formulation decades of research on Gaussian processes (which defines a distribution on functions) and Dirichlet process (which defines a distribution on distributions). Popular BNP techniques that use these processes include Gaussian process regression, in which the correlation structure is improved as the sample size increases, and Dirichlet process mixture models for clustering, which adapt the number of clusters to the complexity of the data. As an example, we can compare and contrast the parametric HMM—which makes an assumption (that often does not hold in practice) that the number of states are known a priori—with the non-parametric Bayesian clustering techniques of infinite Gaussian Mixture Models (IGMMs) which can be used to dynamically learn the number of states without a prior knowledge of the number of clusters.

6.7.1 Applications of BNP methods in CRNs

Saad et al. (2012) have proposed using cooperation between CR devices that are observing the availability pattern of PUs, and the use of BNP techniques to estimate PU activity pattern's distributions. In another work, spectrum access in CRNs was modeled as a repeated auction game and a Bayesian nonparametric belief update scheme was constructed based on the Dirichlet process (Han et al. 2011). In practice, different PUs can have different traffic patterns which will provide time-varying spectrum opportunities for SUs. In previous work, BNP inference model with unknown number of traffic types (based on the Dirichlet process mixture model) has been used for clustering PU traffic patterns with the cluster parameters being utilized by SUs to determine the PU channel idle time distribution and optimizing the transmission strategy accordingly.

6.7.2 Applications of BNP methods for routing

Since BNP techniques are primarily tools for inferencing, clustering, planning and prediction, BNP techniques are useful for a variety of tasks (such as learning and predicting PU activity) that complement the algorithmic problem of routing in CRNs. Their use in this context in CRNs have already been covered in previous paragraph. They have not bee used directly for routing according to the best of our knowledge.

6.7.3 Pitfalls and challenges

Bayesian analysis is appealing since it provides a mathematical formulation of how previous knowledge can be incorporated with fresh evidence to create new knowledge. However, choosing the right prior distribution is not trivial with the prior being chosen mostly in practice for computational ease (e.g., for conjugacy). A major pitfall is making an incorrect assumption about the prior which can significantly skew inference. It is for this reason that some statisticians feel uneasy about the use of prior distributions fearing that it may distort "what the data are trying to say." (Box and Tiao 2011). In addition, another challenge to successfully employing non-parametric learning techniques in CRNs is that such techniques, including DPMM, typically require a larger number of iterations compared to parametric methods.

To conclude this section on learning algorithms, a representative summary of this section on learning techniques for CRNs is captured in Table 4. In addition, the comparison the main techniques, along with their pros and cons, is presented in Table 5.

Table 4 Summary of the various learning techniques discussed in Sect. 6

Learning techniques	Applications to routing	General applications to CRNs
Hidden Markov model	Can indirectly utilize spectrum occupancy and channel quality predictions	Spectrum occupancy prediction: Akbar and Tranter (2007), Park et al. (2007) and Choi and Hossain (2013); spectrum sensing, primary signal detection (see references in He et al. 2010)
Reinforcement learning	Q-routing algorithm (Boyan and Littman 1994); learning automata (Akbari Torkestani and Meybodi 2010a, b); RL-based routing for CRNs (Xia et al. 2009), MANETs (Bhorkar et al. 2012; Di Felice et al. 2010), see further references in survey papers (Bkassiny et al. 2013; Al-Rawi et al. 2013; Yau et al. 2010)	Dynamic channel selection and topology management (Yau et al. 2010); spectrum sensing and efficient spectrum utilization through PU detection (Reddy 2008); security (He et al. 2010)
Learning with games	Evolutionary game theory; dynamic Bayesian games (Pavlidou and Koltsidas 2008); congestion games (Pavlidou and Koltsidas 2008); quality of routing games (Busch et al. 2012)	Spectrum access games (Van der Schaar and Fu 2009)
Online learning	No regret routing for adhoc networks (Avramopoulos et al. 2008; Bhorkar and Javidi 2010); Bandit routing (Tehrani and Zhao 2013); online adaptive routing (Awerbuch and Kleinberg 2008)	Opportunistic spectrum access (Han et al. 2007)
Genetic algorithms	Shortest path routing (Ahn and Ramakrishna 2002)	Modeling wireless channel: (Rondeau et al. 2004b)
Ant colony optimization	Routing with ACO in CRNs (Zhao et al. 2012) and MANETs (Caro et al. 2005)	Cognitive engine design (Zhao et al. 2012)
Artificial neural networks	Routing with ANNs Ju and Evans (2010) and Barbancho et al. (2006)	Spectrum occupancy prediction (Tumuluru et al. 2010); dynamic channel selection (Baldo et al. 2009); radio parameter adaptation (see ref. in He et al. 2010)
Bayesian learning	Bayesian routing in delay-tolerant-networks (Ahmed and Kanhere 2010)	Establishing PU's activity pattern (Saad et al. 2012; Han et al. 2011); channel estimation (Haykin 2005); channel quality prediction (Xing et al. 2013)

6.8 Relationship of learning with reasoning

For a radio to be deemed a *cognitive* radio, it is necessary for it to be equipped with the ability of learning and reasoning (Haykin 2005). Reasoning is an important aspect of CRN behavior and is necessary for cognitive behavior. Reasoning techniques are needed in the context of cognitive networks to address tasks **T**5 described in Sect. 3. We have already seen some learning techniques such as Bayesian networks as well as metaheuristic techniques that can also

Table 5 Comparison of the main AI techniques presented in this paper

Technique	Class	Pros	Cons
Optimization techniques	Single-player optimization	Well-developed theory that has been extensively applied. Many interesting practical problems can be formulated as convex optimization problem whose efficient solution methods exist	Requires complete knowledge of the environment which can be impractical in many situations; The theory does not incorporate multiple decision makers or changing environment
Markov decision processes (MDP)	Single-player optimization (with controllable states)	MDP is a useful tool for sequential planning, or control, of dynamic CRN processes. It can model and find optimal actions for situations where the outcome does not follow deterministically from actions	MDPs make two assumptions (the Markovian assumption and assumption of a stationary environment) that may not be realistic in CRNs. MDPs also assume a single player and cannot model the presence of multiple users in CRNs
Hidden Markov models (HMM)	Single-player optimization (with hidden states)	HMM are excellent models of temporal processes and can be used to model and analyze CRN processes such as PU arrivals	The Markovian assumption, fundamental to HMM, is often not satisfied by temporal processes with memory. Also, HMM is a supervised learning technique that is entirely suited for routing optimization
Reinforcement learning (RL)	Unsupervised learning	Can be applied in unknown environments. Can optimally solve Markov decision processes (MDPs)	The main drawback of the RL technique is its slow convergence. RL techniques may not converge towards optimal solution. More efficient methods exist in supervised settings/or when the environment is known
Genetic algorithms (GA)	Metaheuristic optimization	Excellent for parameter optimization and search of solutions for complex problems for which optimal solutions are unavailable/too expensive	The main problem of GAs is slow convergence. Can suffer from local minimas. Genetic algorithms are not well suited to real-time applications

Table 5 continued

Technique	Class	Pros	Cons
Ant colony optimization (ACO)	Metaheuristic optimization	ACO is an efficient simple metaheuristic that is useful for routing in dynamically changing environments (such as those present in CRNs)	The disadvantages of ACO stem from the heuristic nature of ACO. There is no guarantee that ACO will converge towards an optimal solution, and the time of convergence may be high for large problems
Artificial neural networks (ANNs)	Supervised learning/unsupervised learning	Excellent for classification; does not require prior knowledge of the distribution of observed process; applies very generally to a wide variety of problems; easy to scale; can identify new patterns	Training of ANNs can be slow and can lead to overfitting; No underlying theory to link application with required network
Game theory	Multi-player optimization	Can model interactive optimal decision making between multiple decision makers; can be used to study competition	Requires complete knowledge of the environment which can be impractical in many situations; Convergence to optimal strategy cannot be guaranteed for many games
Bayesian non-parametric (BNP) methods	Supervised learning/unsupervised learning	BNP techniques are primarily tools for inferencing, clustering, planning and prediction. BNP techniques have been widely applied in this context in CRNs	While BNP techniques are useful for a variety of tasks (such as learning and predicting PU activity), their direct application for the direct algorithmic problem of routing in CRNs is limited

be used for the task of reasoning (Gavrilovska et al. 2013). While reasoning is an important part of cognition, a comprehensive treatment of various reasoning tools and techniques useful for CRNs is outside the scope of this work. For illustrative purposes, we will discuss fuzzy cognitive maps as a representative reasoning technique useful for cognitive networking and will refer the interested readers to a recent survey for a comprehensive account of methods, techniques, issues and challenges in implementing reasoning in CRNs (Gavrilovska et al. 2013).

6.8.1 Fuzzy cognitive maps

Fuzzy logic is a formalism that is useful for reasoning in systems and situations having inherent uncertainty or ambiguity. Since complete environmental knowledge is difficult, or even impossible, to obtain in CRNs. Fuzzy logic is a natural fit to the CRN environment where there is limited or no information about certain environment factors. Fuzzy logic based reasoning has been used commonly in CRNs (Erman et al. 2009; Shatila 2012). Fuzzy techniques have also been deployed for strategic QoS routing in networks (for instance, in the work Vasilakos et al. (1998) that uses evolutionary-fuzzy prediction for routing in broadband networks).

Fuzzy cognitive maps (FCMs) are fuzzy-logic based graph structures used for representing causal reasoning (Kosko 1986). FCMs represent a reasoning formalism much like neural networks, Bayesian and Markov networks which can be used for representing cause-effect relationships between variables of a given problem.

FCMs can be compared with traditional expert systems and Bayesian network which are both frameworks that allow reasoning based on causal knowledge or relationships. Traditional expert systems consist of a knowledge base comprising of rules of the form of IF *some condition* THEN some action. The inference engine can then determine the system state and action depending on the input. Despite some early success, expert systems are primarily limited to a settings that do not have uncertainty and are relatively simple (and can be described using explicit rules). Bayesian networks offer an alternate framework for reasoning based on causal relationships that involve uncertainty. FCMs improve upon the capability of Bayesian networks in that FCMs can also handle complex feedback loops which Bayesian networks cannot accommodate. An advantage of FCM framework is that it allows effective handling of different uncertainties by allowing merging of several FCMs into one FCM.

Application of FCMs to CRNs and routing FCMs can be used to facilitate fuzzy-logic based reasoning in CRNs (Gavrilovska et al. 2013). FCMs work based on cause-effect dependencies have already been proposed for cognitive networking applications by Facchini et al. (2013). FCM can be embedded in radio network controllers in the capacity of cognitive engines. FCMs' use in cognitive networking is mainly motivated by the fact that FCMs facilitate cross-layered optimization and can enable reasoning within the cognition cycle. FCM has already been used for cognitive networking in the context of cognitive rate adaptation of WLANs (Facchini et al. 2011) and for ensuring dynamic green (energy-efficient) self-configuration of 3G base stations (Facchini et al. 2013). According to the best of our knowledge, FCM has not been directly used to solve the problem of routing in networks.

Pitfalls and challenges Inference of causality between events only based on observational data is not readily accessible without a priori knowledge. Also, the problem of abductive reasoning, which guesses the cause responsible for a given effect, is an NP-hard problem.

6.8.2 Other reasoning techniques

Knowledge can be represented using an *ontology* which provides shared vocabulary useful for modeling a domain, e.g., it can be used to model the type of objects and concepts existing in a system or domain, and their mutual relationship and properties (Gavrilovska et al. 2013). A rule based system can make use of a knowledge base and some means of inference through an *inference engine*. It is also possible to reason by analogy. This involves the transferring of knowledge from a past analogous situation to another similar present situation. Case-based reasoning (CBR) is a well-known kind of analogy making which has been exploited in CRN research (He et al. 2010). In case-based reasoning a database of existing cases is maintained and used to draw conclusions about new cases. The CBR reasoning method can utilize procedures like pattern matching and various statistical techniques to find which historical case to relate to the current case. More details about other prominent reasoning techniques can be seen in a recent survey paper on this topic (Gavrilovska et al. 2013).

6.9 Some inferential tasks in cognitive routing

Future cognitive routing protocols can benefit from the following inferential (i.e., prediction-based) tasks: (1) channel quality modeling and prediction, and (2) spectrum occupancy modeling (including the modeling and prediction of PUs). We will first highlight the importance of these prediction-based tasks in the context of cognitive routing, and will then discuss the listed tasks under their respective headings. We note that these tasks relate closely to the task **T**6 described in Sect. 3.

Relationship to cognitive routing We will now discuss how these inferential predictions can improve cognitive routing. The prime motivation of learning PU dynamics is to incorporate these predictions into the utility/reward functions into the various decision-theoretic and learning frameworks we have studied in this paper (such as optimization theory, MDP, game-theory, HMM, RL, etc.) so that more stable routes (that are disrupted less) are preferred. In particular, a CR that manages to learn the behavioral patterns of a primary user by modeling it can optimize its performance by exploiting the learned model. For example, a SU can exploit information, potentially gleaned from spectrum sensing data, and select white spaces (that emerge due to the absence of PUs) that tend to be longer lived at certain times of day and at certain locations. Knowledge of PU patterns can also be helpful for advanced planning when a SU has to decide the channel to switch to on the arrival of a PU (which will help reduce the temporal connection loss faced by SUs and potential interference faced by PUs due to any delays in vacation of channel by SUs) (Doyle 2009). Accurate prediction of PU arrivals and network dynamics can also help in improving the accuracy of the model adopted in model-based approaches. While model-based approaches are computationally expensive, it has been shown that they are more effective than model-free approaches provided an accurate model is available (Sammut and Webb 2010). To be reminded about the distinction between model-based and model-free approaches, the reader is referred to the Sect. 6.2.2. In CRNs, both model-based and model-free approaches can be used and both have their pros and cons. It is in the context of model-based approaches that PU activity prediction is useful for cognitive routing.

6.9.1 Channel quality modeling and prediction

It is useful in the context of cognitive routing to be able to estimate an accurate model of the current and future (predicted) quality of a channel. Rondeau et al. (2004b) proposed using HMM to model the wireless channel online with the HMM being trained using a genetic

algorithm. Akbar (2007) techniques for modeling wireless network channel using Markov models are presented along with techniques for efficient estimation of Markov model parameters (including the number of states) to aid in reproducing and/or forecasting channel statistics accurately. In another work, Xing et al. (2013) have proposed to perform channel quality prediction using Bayesian inference. Channel estimation problem has also been addressed in Haykin (2005) in which the use of particle filters, rooted in Bayesian estimation, were proposed as a device for tracking statistical variations in a wireless channel. Researchers have also proposed using an ANN-based cognitive engine for learning how various channel's quality status affects performance and thereby dynamically selecting a channel that improves performance. The dynamic selection of channels has an obvious implication for network-layer functionality and the routing algorithm for such networks should be able to keep up with the channel changes so that the best performing routes are selected.

6.9.2 Spectrum occupancy modeling

The cognitive routing task can benefit from modeling the spectrum occupancy. A satisfactory model of spectrum occupancy (or, of spectrum white spaces) should incorporate: (1) states of the channel along with their transition behavior, and (2) the *sojourn time* or the time duration the system resides in each of the states (Geirhofer et al. 2007).

Since many DSA environments (e.g., contention based protocols such as IEEE 802.11) do not have a slotted structure, it is more appropriate to use a continuous-time (CT) model. A CT model that is especially relevant to DSA, and one that is popularly used for modeling spectrum occupancy, is the semi-Markov model (SMM) which generalizes the concept of CT Markov chains (CTMCs). Although both the semi-Markov and CTMC models have the Markovian property and they describe the transition behavior in the same way, a SMM allows for specifying the occupancy periods, or the sojourn time, for each state arbitrarily. In particular, the occupancy time does need have to be necessarily exponentially distributed as must be the case for CTMCs by definition (Kleinrock 1975; Ross 1970). Specification of a SMM therefore requires both the statistical specification of the transition behavior and of the sojourn time within each state (Kleinrock 1975; Ross 1970).

It has been posited that for practical purposes of analyzing DSA/CRNs, a simple two-state semi-Markov ON–OFF model is adequate for modeling spectrum usage (López-Benítez and Casadevall 2011) (Table 1 may be referred to see the popularity of this model). The OFF state represents an idle channel, while the ON state indicates a busy channel not available for opportunistic access, with the length of ON and OFF periods being random variables (RVs) following some specified distribution. Such a model is also known as a stochastic duty cycle model (Wellens and Mähönen 2010). The use of this simple semi-Markov ON–OFF model is quite popular (Geirhofer et al. 2007), although other more elaborate models are also available (e.g., Jiang et al. 2012 modify the ON/OFF Markov model by also incorporating a priority service queue to provide QoE-driven channel allocation). Geirhofer et al. (2006) showed that such a basic Markov model can be used to empirically model the spectrum use in IEEE 802.11b WLAN-systems. It was noted that their results should also extend to other systems having multi-access protocols similar to CSMA/CA.

An important aspect of using such semi-Markov models is specifying the state sojourn or stay times, and to study if successive period lengths are correlated. The simplest approach is to assume the state sojourn time is exponentially distributed and that successive stay times are not correlated. Such an approach is interesting due to its simplicity and tractability. Unfortunately, studies have shown that this simple approximation does not tally up well with empirical studies on actual systems (Geirhofer et al. 2007). Nonetheless, exponential

distribution is still used heuristically (Lee and Akyildiz 2008), although such an approach is not entirely justified statistically, since this assumption makes the model earlier to apply in practice. Empirical studies have shown that state sojourn times typically have larger variability than suggested by the exponential distribution. In fact, the distributions of the ON and in particular the OFF period were often found to be heavy-tailed (Geirhofer et al. 2007).

While focusing on spectrum occupancy modeling, it is extremely important for cognitive routing in DSA CRN networks that we are able to model, in particular, the activities of PUs. Various models for traffic pattern prediction for PU are presented in Li and Zekavat (2008). Wang et al. (2009) have proposed modeling the interaction between the PUs and SUs through continuous-time Markov chains (CTMC). In addition, Bayesian nonparametric techniques (Saad et al. 2012) and ANNs (Tumuluru et al. 2010) have also been proposed to estimate PU activity pattern's distributions. A lot of studies have focused on empirical modeling of spectrum usage and have proposed various models for PU traffic pattern (Wellens and Mähönen 2010; Wellens et al. 2009b, a; Harrold et al. 2011; Ghosh et al. 2010; Hoyhtya et al. 2010). For further details, interested readers are referred to the survey papers (López-Benítez and Casadevall 2011; Masonta et al. 2013) in which various statistical models proposed in literature for modeling temporal and spatial spectrum occupancy are reviewed in detail. For more details about spectrum prediction techniques, the interested readers are referred to detailed survey papers on this topic (Xing et al. 2013; Saleem and Rehmani 2014) and the references therein.

7 Open issues and future work

In this section, we will outline some of the major open research issues in building cognitive networks and in developing AI-enabled cognitive routing protocols. We will also discuss potential future work.

7.1 Multi-agent decision-making in complex environments

The cognitive radio networking environment is naturally amenable to distributed *multi-agent* decision making rather than centrally controlled optimization. We have seen earlier how multi-agent environments are much more challenging to design than their single-agent counterparts. Ideas from game-theory and economic market design will become increasingly important as multi-agent learning becomes commonplace in CRN design. With AI-based cognitive networks becoming mainstream, it will be important to understand the behavior of the overall CRN system in terms of equilibria and dynamics for large distributed networks with multiple learning CRs, each taking self-serving decisions with access to limited information. This is because emergent behavior of CRNs, composed of multiple self-interested CR servicing users with distinct context, can be *complex*. This can manifest itself when slight changes in one or more of the system parameters result in dramatic changes in system behavior (Haykin 2005). The task of cognitive routing is made complex by the fact that two cognitive loops (device level and network level) operate at different time scales and their interdependence is largely unexplored (Haigh and Partridge 2011). Researchers can exploit advances in the study of complexity to understand the dynamics of such CRNs (Waldrop 1992). Also, while it is quite common to use simplistic assumptions (such as the Markovian assumption or the perfect knowledge assumption) to keep our models tractable, real systems are in fact quite complex with CRs often operating in unknown RF environments. There is a lot of scope of new research in areas of decision making and learning in non-Markovian,

partially observed, or unknown environments. In such unknown environments, the usage of model-free and online methods seem promising.

7.2 Application of varied machine-learning techniques

Game theory, reinforcement learning, neural networks, and genetic algorithms, due to their natural fit to the kind of problems faced in CRNs, are understandably the most used AI techniques in CRNs. However, as listed in this survey paper, there are various other machine-learning techniques that can be plausibly applied to tasks much more diverse than their current application. In particular, it is anticipated that Bayesian techniques will find increasing use in CRNs. It is an open research question that which machine-learning techniques, apart from the current popular approaches, would prove to be most successful in solving problems in CRNs.

7.3 Interworking with other modern technologies

The interplay of cognitive radios with the software defined networking (SDN) architecture, which allows a standards based interface (McKeown et al. 2008) between a centralized 'network controller'[15] and networking devices, should be explored. It is possible that interesting use cases will emerge that will synergize the mainly centralized operational paradigm of SDNs with the mainly distributed operational paradigm of CRs. While the emphasis of SDN architecture has been on the separation of control and data planes, it is worth exploring if a combined SDN and CR architecture can help realize the vision of having a 'knowledge plane' for networks as envisioned by Clark et al. (2003). Also, it is worth exploring how cognitive networks may seamlessly integrate modern technologies like internet-of-things, pervasive and ubiquitous technology, and cloud computing. In particular, cloud computing can be used to aggregate state information from various wireless nodes and for performing optimization computations centrally in datacenters rather than on constrained wireless devices.

7.4 Cognitive traffic engineering and congestion control

In this paper, our particular focus has been on cognitive routing in the settings of cognitive networks. Apart from routing, there are also other correlated problems such as traffic engineering that focuses on network robustness (by minimizing network congestion by changing what happens inside a network) and congestion control that focuses on user performance (by maximizing user utility by controlling the edge of the network) (Fortz et al. 2002) that are important from the perspective of network-wide optimization. The relationship of traffic engineering and congestion control with routing is as follows. In traffic engineering, the focus is on minimizing network congestion by devising appropriate routing given the traffic. In congestion control, on the other hand, we are interested in maximizing user utility given the routing state by adapting the end transmitting rate. More research is needed on the interaction between routing, traffic engineering, and congestion control in the setting of cognitive networking.

8 Conclusion

Learning lies at the core of the vision of cognitive radio and cognitive radio networks. While a lot of previous research attention has focused on general AI techniques for optimizing

[15] The centralized SDN network controller can itself be built as a distributed system to be scalable and avoid a single point of failure.

PHY and MAC layer parameters for CRs, scant attention has been given to utilizing learning techniques at the network layer particularly for the problem of routing. We have argued in this paper that incorporating learning from the past and present conditions can be very productive and can lead to improved CRN performance. In this paper, we have surveyed the set of techniques that can be used to embed learning in the routing framework of CRNs. Open research issues and potential directions for future work have also been identified.

References

A brief history of neural networks [online]. http://www.dtreg.com/mlfn.htm. Accessed 17 Aug 2013

Abbagnale A, Cuomo F (2010) Gymkhana: a connectivity-based routing scheme for cognitive radio ad hoc networks. In: INFOCOM IEEE conference on computer communications workshops, 2010. IEEE, pp 1–5

Abu-Mostafa YS, Magdon-Ismail M, Lin H-T (2012) Learning from data. AMLBook, USA

Adamopoulou E, Demestichas K, Demestichas P, Theologou M (2008) Enhancing cognitive radio systems with robust reasoning. Int J Commun Syst 21(3):311–330

Ahmed S, Kanhere SS (2010) A Bayesian routing framework for delay tolerant networks. In: Wireless communications and networking conference (WCNC), 2010 IEEE. IEEE, pp 1–6

Ahn CW, Ramakrishna RS (2002) A genetic algorithm for shortest path routing problem and the sizing of populations. IEEE Trans Evol Comput 6(6):566–579

Ahuja RK, Magnanti TL, Orlin JB (1993) Network flows: theory, algorithms, and applications. Prentice-Hall, Englewood Cliffs, NJ

Akbar IA (2007) Statistical analysis of wireless systems using Markov models. Ph.D. thesis, Virginia Polytechnic Institute and State University

Akbar IA, Tranter WH (2007) Dynamic spectrum allocation in cognitive radio using hidden Markov models: poisson distributed case. In: SoutheastCon, 2007. Proceedings of the IEEE. IEEE, pp 196–201

Akbari Torkestani J, Meybodi MR (2010a) Mobility-based multicast routing algorithm for wireless mobile ad-hoc networks: a learning automata approach. Comput Commun 33(6):721–735

Akbari Torkestani J, Meybodi MR (2010b) An intelligent backbone formation algorithm for wireless ad hoc networks based on distributed learning automata. Comput Netw 54(5):826–843

Akkarajitsakul K, Hossain E, Niyato D, Kim DI (2011) Game theoretic approaches for multiple access in wireless networks: a survey. IEEE Commun Surv Tutor 13(3):372–395

Akyildiz IF, Lee W-Y, Vuran MC, Mohanty S (2006) Next generation/dynamic spectrum access/cognitive radio wireless networks: a survey. Comput Netw 50(13):2127–2159

Alba E (2005) Parallel metaheuristics: a new class of algorithms, vol 47. Wiley, New York

Alba E, Troya JM (1999) A survey of parallel distributed genetic algorithms. Complexity 4(4):31–52

Almasaeid HM, Kamal AE (2013) Exploiting multichannel diversity for cooperative multicast in cognitive radio mesh networks. IEEE ACM Trans Netw 22(3):770–783. doi:10.1109/TNET.2013.2258035

Almasaeid HM, Jawadwala TH, Kamal AE (2010) On-demand multicast routing in cognitive radio mesh networks. In: Global telecommunications conference (GLOBECOM 2010), 2010 IEEE. IEEE, pp 1–5

Al-Rawi HA, Yau K-LA (2013) Routing in distributed cognitive radio networks: a survey. Wirel Pers Commun 69(4):1983–2020

Al-Rawi HA, Ng MA, Yau K-LA (2013) Application of reinforcement learning to routing in distributed wireless networks: a review. Artif Intell Rev 43(3):381–416

Altman E (2002) Applications of Markov decision processes in communication networks. In: Handbook of Markov decision processes. Springer, Berlin, pp 489–536

Altman E, Boulogne T, El-Azouzi R, Jiménez T, Wynter L (2006) A survey on networking games in telecommunications. Comput Oper Res 33(2):286–311

Aspnes J, Azar Y, Fiat A, Plotkin S, Waarts O (1997) On-line routing of virtual circuits with applications to load balancing and machine scheduling. J ACM (JACM) 44(3):486–504

Attar A, Tang H, Vasilakos AV, Yu FR, Leung VC (2012) A survey of security challenges in cognitive radio networks: solutions and future research directions. Proc IEEE 100(12):3172–3186

Auer P, Cesa-Bianchi N, Fischer P (2002a) Finite-time analysis of the multiarmed bandit problem. Mach Learn 47(2–3):235–256

Auer P, Cesa-Bianchi N, Freund Y, Schapire RE (2002b) The nonstochastic multiarmed bandit problem. SIAM J Comput 32(1):48–77

Avramopoulos IC, Rexford J, Schapire RE (2008) From optimization to regret minimization and back again. In: SysML
Awerbuch B, Kleinberg R (2008) Online linear optimization and adaptive routing. J Comput Syst Sci 74(1):97–114
Baldo N, Tamma BR, Manojt B, Rao R, Zorzi M (2009) A neural network based cognitive controller for dynamic channel selection. In: Communications, 2009. ICC'09. IEEE international conference on. IEEE, pp 1–5
Bansal N, Blum A, Chawla S, Meyerson A (2003) Online oblivious routing. In: Proceedings of the fifteenth annual ACM symposium on parallel algorithms and architectures. ACM, pp 44–49
Barbancho J, León C, Molina J, Barbancho A (2006) SIR: a new wireless sensor network routing protocol based on artificial intelligence. In: Advanced web and network technologies, and applications. Springer, Berlin, pp 271–275
Basar T, Olsder GJ, Clsder G, Basar T, Baser T, Olsder GJ (1995) Dynamic noncooperative game theory, vol 200. SIAM, Philadelphia, PA
Bellman R (1957) Dynamic programming. Princeton University Press, Princeton, NJ
Beltagy I, Youssef M, El-Derini M (2011) A new routing metric and protocol for multipath routing in cognitive networks. In: Wireless communications and networking conference (WCNC), 2011 IEEE, pp 974–979
Bertsekas DP (1979) Dynamic models of shortest path routing algorithms for communication networks with multiple destinations. In: Decision and control including the symposium on adaptive processes, 1979 18th IEEE Conference on, vol 18. IEEE, pp 127–133
Bertsekas DP (2011) Approximate dynamic programming. In: Dynamic programming and optimal control, 4th edn, vol II. Approximate dynamic programming, 4th edn, 2012, Athena Scientific, USA
Bertsekas DP, Tsitsiklis JN (1995) Neuro-dynamic programming: an overview. In: Decision and control. Proceedings of the 34th IEEE conference on vol 1. IEEE, pp 560–564
Bertsekas DP, Gallager RG, Humblet P (1992) Data networks, vol 2. Prentice-Hall International, Englewood Cliffs, NJ
Bhorkar A, Javidi T (2010) No regret routing for ad-hoc wireless networks. In: Signals, systems and computers (ASILOMAR), 2010 conference record of the forty fourth Asilomar conference on. IEEE, pp 676–680
Bhorkar AA, Naghshvar M, Javidi T, Rao BD (2012) Adaptive opportunistic routing for wireless ad hoc networks. IEEE/ACM Trans Netw (TON) 20(1):243–256
Biswas S, Morris R (2005) Exor: opportunistic multi-hop routing for wireless networks. In: ACM SIGCOMM computer communication review, vol 35. ACM, pp 133–144
Bkassiny M, Li Y, Jayaweera S (2013) A survey on machine-learning techniques in cognitive radios. IEEE Commun Surv Tutor 15 :1136–1159
Blum A, Monsour Y (2007) Learning, regret minimization, and equilibria. Cambridge University Press, Cambridge, MA
Box GE, Tiao GC (2011) Bayesian inference in statistical analysis, vol 40. Wiley, New York
Boyan JA, Littman ML (1994) Packet routing in dynamically changing networks: a reinforcement learning approach. Adv Neural Inf Process Syst 671–671
Boyd SP, Vandenberghe L (2004) Convex optimization. Cambridge University Press, Cambridge, MA
Busch C, Kannan R, Vasilakos AV (2012) Approximating congestion+dilation in networks via" quality of routing games. IEEE Trans Comput 61(9):1270–1283
Busoniu L, Babuska R, De Schutter B (2008) A comprehensive survey of multiagent reinforcement learning. IEEE Trans Syst Man Cybern Part C Appl Rev 38(2):156–172
Busoniu L, Ernst D, De Schutter B, Babuska R (2011) Approximate reinforcement learning: an overview. In: Adaptive dynamic programming and reinforcement learning (ADPRL), 2011 IEEE symposium on. IEEE, pp 1–8
Byun S-S, Balashingham I, Vasilakos AV, Lee H-N (2014) Computation of an equilibrium in spectrum markets for cognitive radio networks. IEEE Trans Comput 63(2):304–316
Cacciapuoti AS, Calcagno C, Caleffi M, Paura L (2010) Caodv: routing in mobile ad-hoc cognitive radio networks. In: Wireless Days (WD), 2010 IFIP. IEEE, pp 1–5
Cacciapuoti AS, Caleffi M, Paura L (2012) Reactive routing for mobile cognitive radio ad hoc networks. Ad Hoc Netw 10(5):803–815
Cadger F, Curran K, Santos J, Moffett S (2013) A survey of geographical routing in wireless ad-hoc networks. IEEE Commun Surv Tutor 15(2):621–653
Caleffi M, Akyildiz IF, Paura L (2012) Opera: optimal routing metric for cognitive radio ad hoc networks. IEEE Trans Wirel Commun 11(8):2884–2894
Campista MEM, Esposito PM, Moraes IM, Costa LHMK, Duarte OCMB, Passos DG, de Albuquerque CN, Saade DCM, Rubinstein MG (2008) Routing metrics and protocols for wireless mesh networks. IEEE Netw 22(1):6–12

Cappé O, Moulines E, Rydén T (2005) Inference in hidden Markov models. Springer, Berlin
Cesana M, Cuomo F, Ekici E (2011) Routing in cognitive radio networks: challenges and solutions. Ad Hoc Netw 9(3):228–248
Chiang M (2005) Geometric programming for communication systems. Now Publishers Inc, Boston
Chiang M, Low SH, Calderbank AR, Doyle JC (2007) Layering as optimization decomposition: a mathematical theory of network architectures. Proc IEEE 95(1):255–312
Choi KW, Hossain E (2011) Opportunistic access to spectrum holes between packet bursts: a learning-based approach. IEEE Trans Wirel Commun 10(8):2497–2509
Choi KW, Hossain E (2013) Estimation of primary user parameters in cognitive radio systems via hidden Markov model. IEEE Trans Signal Process 61(3):782–795. doi:10.1109/TSP.2012.2229998
Chowdhury KR, Akyildiz IF (2011) Crp: a routing protocol for cognitive radio ad hoc networks. IEEE J Sel Areas Commun 29(4):794–804
Chowdhury KR, Felice MD (2009) Search: a routing protocol for mobile cognitive radio ad-hoc networks. Comput Commun 32(18):1983–1997
Clancy C, Hecker J, Stuntebeck E, O'Shea T (2007) Applications of machine learning to cognitive radio networks. IEEE Wirel Commun 14(4):47–52
Clark DD, Partridge C, Ramming JC, Wroclawski JT (2003) A knowledge plane for the internet. In: Proceedings of the 2003 conference on applications, technologies, architectures, and protocols for computer communications. ACM, pp 3–10
Črepinšek M, Liu S-H, Mernik M (2013) Exploration and exploitation in evolutionary algorithms: a survey. ACM Comput Surv (CSUR) 45(3):35
De Couto DS, Aguayo D, Bicket J, Morris R (2005) A high-throughput path metric for multi-hop wireless routing. Wirel Netw 11(4):419–434
Deng S, Chen J, He H, Tang W (2007) Collaborative strategy for route and spectrum selection in cognitive radio networks. In: Future generation communication and networking (FGCN 2007), vol 2. IEEE, pp 168–172
Di Caro G, Ducatelle F, Gambardella LM (2005) Anthocnet: an adaptive nature-inspired algorithm for routing in mobile ad hoc networks. Eur Trans Telecommun 16(5):443–455
Di Felice M, Chowdhury KR, Bononi L (2011) Learning with the bandit: a cooperative spectrum selection scheme for cognitive radio networks. In: Global telecommunications conference (GLOBECOM 2011), 2011 IEEE. IEEE, pp 1–6
Di Felice M, Chowdhury KR, Wu C, Bononi L, Meleis W (2010) Learning-based spectrum selection in cognitive radio ad hoc networks. In: Wired/wireless internet communications. Springer, Berlin, pp 133–145
Ding L, Melodia T, Batalama S, Medley MJ (2009) Rosa: distributed joint routing and dynamic spectrum allocation in cognitive radio ad hoc networks. In: Proceedings of the 12th ACM international conference on modeling, analysis and simulation of wireless and mobile systems. ACM, pp 13–20
Ding L, Melodia T, Batalama SN, Matyjas JD, Medley MJ (2010) Cross-layer routing and dynamic spectrum allocation in cognitive radio ad hoc networks. IEEE Trans Veh Technol 59(4):1969–1979
Doyle L (2009) Essentials of cognitive radio. Cambridge University Press, Cambridge, MA
Duarte PB, Fadlullah ZM, Vasilakos AV, Kato N (2012) On the partially overlapped channel assignment on wireless mesh network backbone: a game theoretic approach. IEEE J Sel Areas Commun 30(1):119–127
Dvir A, Vasilakos AV (2011) Backpressure-based routing protocol for dtns. ACM SIGCOMM Comput Commun Rev 41(4):405–406
Eidenbenz S, Resta G, Santi P (2005) Commit: a sender-centric truthful and energy-efficient routing protocol for ad hoc networks with selfish nodes. In: Parallel and distributed processing symposium, 2005. Proceedings. 19th IEEE international, 10 pp. doi:10.1109/IPDPS.2005.142
Ephremides A, Verdu S (1989) Control and optimization methods in communication network problems. IEEE Trans Autom Control 34(9):930–942
Ephremides A, Hajek B (1998) Information theory and communication networks: an unconsummated union. IEEE Trans Inf Theory 44(6):2416–2434
Erman M, Mohammed A, Rakus-Andersson E (2009) Fuzzy logic applications in wireless communications. In: IFSA/EUSFLAT conference, pp 763–767
Facchini C, Granelli F, da Fonseca NL (2011) Cognitive rate adaptation in wireless lans. In: ICC, pp 1–5
Facchini C, Holland O, Granelli F, Da Fonseca NL, Aghvami H (2013) Dynamic green self-configuration of 3G base stations using fuzzy cognitive maps. Comput Netw 57(7):1597–1610
Fahad M, Qadir J, Baig A (2010) Broadcasting in cognitive wireless mesh networks with dynamic channel conditions. In: Emerging technologies (ICET), 2010 6th international conference on. IEEE, pp 400–404

Farooq MJ, Hussain M, Qadir J, Baig A (2013) A game-theoretic spectrum allocation framework for mixed unicast and broadcast traffic profile in cognitive radio networks. In: Local computer networks (LCN), 2013 IEEE 38th Conference on, IEEE, pp 425–432
Felegyhazi M, Hubaux J-P (2006) Game theory in wireless networks: a tutorial. Technical Report, Technical Report LCA-REPORT-2006-002, EPFL
Felegyhazi M, Hubaux J-P, Buttyan L (2006) Nash equilibria of packet forwarding strategies in wireless ad hoc networks. IEEE Trans Mobile Comput 5(5):463–476
Fette BA (2009) Cognitive radio technology, 2nd edn. Academic Press, USA
Filippini I, Ekici E, Cesana M (2009) Minimum maintenance cost routing in cognitive radio networks. In: Mobile adhoc and sensor systems, 2009. MASS'09. IEEE 6th international conference on, IEEE, pp 284–293
Fortuna C, Mohorcic M (2009) Trends in the development of communication networks: cognitive networks. Comput Netw 53(9):1354–1376
Fortz B, Rexford J, Thorup M (2002) Traffic engineering with traditional IP routing protocols. IEEE Commun Mag 40(10):118–124
Friend DH (2009) Cognitive networks: foundations to applications. Ph.D. thesis. Virginia Polytechnic Institute and State University
Friend DH, EINainay M, Shi Y, MacKenzie AB (2008) Architecture and performance of an island genetic algorithm-based cognitive network. In: Consumer communications and networking conference, 2008. CCNC 2008. 5th IEEE. IEEE, pp 993–997
Fudenberg D (1998) The theory of learning in games, vol 2. MIT Press, Cambridge, MA
Fu F, Van Der Schaar M (2010) A systematic framework for dynamically optimizing multi-user wireless video transmission. IEEE J Sel Areas Commun 28(3):308–320
Gábor Z, Kalmár Z, Szepesvári C (1998) Multi-criteria reinforcement learning. ICML 98:197–205
Gai Y, Krishnamachari B, Jain R (2012) Combinatorial network optimization with unknown variables: multi-armed bandits with linear rewards and individual observations. IEEE/ACM Trans Netw (TON) 20(5):1466–1478
Gallager RG (1977) A minimum delay routing algorithm using distributed computation. IEEE Trans Commun 25(1):73–85
Ganesan D, Govindan R, Shenker S, Estrin D (2001) Highly-resilient, energy-efficient multipath routing in wireless sensor networks. ACM SIGMOBILE Mobile Comput Commun Rev 5(4):11–25
Gao C, Shi Y, Hou YT, Sherali HD, Zhou H (2011) Multicast communications in multi-hop cognitive radio networks. IEEE J Sel Areas Commun 29(4):784–793
Gavrilovska L, Atanasovski V, Macaluso I, DaSilva L (2013) Learning and reasoning in cognitive radio networks. IEEE Commun Surv Tutor 15(4):1761–1777. doi:10.1109/SURV.2013.030713.00113
Geirhofer S, Tong L, Sadler BM (2006) Dynamic spectrum access in wlan channels: empirical model and its stochastic analysis. In: Proceedings of the first international workshop on technology and policy for accessing spectrum. ACM, p 14
Geirhofer S, Tong L, Sadler BM (2007) Cognitive radios for dynamic spectrum access-dynamic spectrum access in the time domain: modeling and exploiting white space. IEEE Commun Mag 45(5):66–72
Gershman SJ, Blei DM (2012) A tutorial on Bayesian nonparametric models. J Math Psychol 56(1):1–12
Ghosh C, Roy S, Rao MB, Agrawal DP (2010) Spectrum occupancy validation and modeling using real-time measurements. In: Proceedings of the 2010 ACM workshop on cognitive radio networks. ACM, pp 25–30
Gittins J (1989) Multi-armed bandit allocation indices. Wiley, New York
Glover F, Kochenberger GA (2003) Handbook of metaheuristics. Springer, Berlin
Goldberg DE, Holland JH (1988) Genetic algorithms and machine learning. Mach Learn 3(2):95–99
Goldsmith AJ, Greenstein LJ, Mandayam NB (2012) Principles of cognitive radio. Cambridge University Press, Cambridge, MA
Gosavi A (2009) Reinforcement learning: a tutorial survey and recent advances. INFORMS J Comput 21(2):178–192
Haigh KZ, Partridge C (2011) Can artificial intelligence meet the cognitive networking challenge? Dayton, OH. http://www.cs.cmu.edu/~khaigh/2011-haigh-EURASIP-JWCN.pdf
Halpern JY (2008) Beyond nash equilibrium: solution concepts for the 21st century. In: Proceedings of the twenty-seventh ACM symposium on principles of distributed computing. ACM, pp 1–10
Han Z, Pandana C, Liu KR (2007) Distributive opportunistic spectrum access for cognitive radio using correlated equilibrium and no-regret learning. In: Wireless communications and networking conference, 2007. WCNC 2007. IEEE. IEEE, pp 11–15
Han Z, Zheng R, Poor HV (2011) Repeated auctions with Bayesian nonparametric learning for spectrum access in cognitive radio networks. IEEE Trans Wirel Commun 10(3):890–900

Han Z, Niyato D, Saad W, Basar T, Hjorungnes A (2012) Game theory in wireless and communication networks. Cambridge University Press, Cambridge, MA
Harrold T, Cepeda R, Beach M (2011) Long-term measurements of spectrum occupancy characteristics. In: New Frontiers in dynamic spectrum access networks (DySPAN), 2011 IEEE symposium on. IEEE, pp 83–89
Haykin S (2005) Cognitive radio: brain-empowered wireless communications. IEEE J Sel Areas Commun 23(2):201–220
He A, Bae KK, Newman TR, Gaeddert J, Kim K, Menon R, Morales-Tirado L, Neel JJ, Zhao Y, Reed JH et al (2010) A survey of artificial intelligence for cognitive radios. IEEE Trans Veh Technol 59(4):1578–1592
Hillier FS, Lieberman GJ (2001) Introduction to operations research. McGraw Hill, New York
Hossain E, Niyato D, Han Z (2009) Dynamic spectrum access and management in cognitive radio networks. Cambridge University Press, Cambridge, MA
Hou YT, Shi Y, Sherali HD (2008) Spectrum sharing for multi-hop networking with cognitive radios. IEEE J Sel Areas Commun 26(1):146–155
How KC, Ma M, Qin Y (2011) Routing and QoS provisioning in cognitive radio networks. Comput Netw 55(1):330–342
Howard R (1960) Dynamic programming and Markov processes. MIT Press, Cambridge, MA
Hoyhtya M, Pollin S, Mammela A (2010) Classification-based predictive channel selection for cognitive radios. In: Communications (ICC), 2010 IEEE international conference on. IEEE, pp 1–6
Htike Z, Hong CS (2013) Broadcasting in multichannel cognitive radio ad hoc networks. In: Wireless communications and networking conference (WCNC), 2013 IEEE. IEEE, pp 733–737
Huang X, Lu D, Li P, Fang Y (2011) Coolest path: spectrum mobility aware routing metrics in cognitive ad hoc networks. In: Distributed computing systems (ICDCS), 2011 31st international conference on, IEEE, pp 182–191
Hu J, Wellman MP (2003) Nash Q-learning for general-sum stochastic games. J Mach Learn Res 4:1039–1069
Izquierdo LR, Izquierdo SS, Vega-Redondo F (2012) Learning and evolutionary game theory. In: Encyclopedia of the sciences of learning. Springer, Berlin, pp 1782–1788
Jain R, Puri A, Sengupta R (2001) Geographical routing using partial information for wireless ad hoc networks. IEEE Pers Commun 8(1):48–57
Jiang T, Wang H, Vasilakos AV (2012) Qoe-driven channel allocation schemes for multimedia transmission of priority-based secondary users over cognitive radio networks. IEEE J Sel Areas Commun 30(7):1215–1224
Jie Y, Kamal AE (2014) Multi-objective multicast routing optimization in cognitive radio networks. http://home.engineering.iastate.edu/kamal/Docs/jk14.pdf. Accessed 08 June 2014
Ju S, Evans JB (2010) Scalable cognitive routing protocol for mobile ad-hoc networks. In: Global telecommunications conference (GLOBECOM 2010), 2010 IEEE. IEEE, pp 1–6
Kaelbling LP, Littman ML, Moore AW (1996) Reinforcement learning: a survey. arXiv preprint cs/9605103
Keshav S (2012) Mathematical foundations of computer networking. Addison-Wesley, Reading, MA
Khan MA, Tembine H, Vasilakos AV (2012) Game dynamics and cost of learning in heterogeneous 4G networks. IEEE J Sel Areas Commun 30(1):198–213
Kim W, Oh SY, Gerla M, Park J-S (2009) Cocast: multicast mobile ad hoc networks using cognitive radio. In: Military communications conference, 2009. MILCOM 2009. IEEE. IEEE, pp 1–7
Kleinrock L (1975) Queueing systems: theory, vol 1. Wiley-Interscience, New York
Knuth DE (2006) Art of computer programming, volume 4, Fascicle 4, The generating all trees-history of combinatorial generation. Addison-Wesley Professional, Reading, MA
Korilis YA, Lazar AA, Orda A (1997) Achieving network optima using Stackelberg routing strategies. IEEE/ACM Trans Netw (TON) 5(1):161–173
Kosko B (1986) Fuzzy cognitive maps. Int J Man Mach Stud 24(1):65–75
Kulkarni S, Harman G (2011) An elementary introduction to statistical learning theory, vol 853. Wiley, New York
Kumar P (1985) A survey of some results in stochastic adaptive control. SIAM J Control Optim 23(3):329–380
Kumar S, Miikkulainen R (1997) Dual reinforcement Q-routing: an on-line adaptive routing algorithm. In: Proceedings of the artificial neural networks in engineering conference, pp 231–238
Lee W-Y, Akyildiz IF (2008) Optimal spectrum sensing framework for cognitive radio networks. IEEE Trans Wirel Commun 7(10):3845–3857
Leyton-Brown K, Shoham Y (2008) Essentials of game theory: a concise multidisciplinary introduction. Synth Lect Artif Intell Mach Learn 2(1):1–88
Li B, Li D, Wu Q-H, Li H (2009) Asar: ant-based spectrum aware routing for cognitive radio networks. In: Wireless communications & signal processing, 2009. WCSP 2009. International Conference on. IEEE, pp 1–5

Li P, Guo S, Yu S, Vasilakos AV (2012) Codepipe: an opportunistic feeding and routing protocol for reliable multicast with pipelined network coding. In: INFOCOM, 2012 Proceedings IEEE. IEEE, pp 100–108
Littlestone N, Warmuth MK (1989) The weighted majority algorithm. In: Foundations of Computer Science, 30th Annual symposium on. IEEE, pp 256–261
Liu KR, Wang B (2010) Cognitive radio networking and security: a game-theoretic view. Cambridge University Press, Cambridge, MA
Liu Y, Cai LX, Shen X (2012) Spectrum-aware opportunistic routing in multi-hop cognitive radio networks. IEEE J Sel Areas Commun 30(10):1958–1968
Liu L, Song Y, Zhang H, Ma H, Vasilakos A (2015) Physarum optimization: a biology-inspired algorithm for the steiner tree problem in networks. IEEE Trans Comput 64:819–832
Li X, Zekavat SA (2008) Traffic pattern prediction and performance investigation for cognitive radio systems. In: Wireless communications and networking conference, 2008. WCNC 2008. IEEE. IEEE, pp 894–899
Lo BF (2011) A survey of common control channel design in cognitive radio networks. Phys Commun 4(1):26–39
López-Benítez M, Casadevall F (2011) An overview of spectrum occupancy models for cognitive radio networks. In: NETWORKING 2011 workshops. Springer, Berlin, pp 32–41
Lott C, Teneketzis D (2006) Stochastic routing in ad-hoc networks. IEEE Trans Autom Control 51(1):52–70
Macaluso I, Finn D, Ozgul B, DaSilva L (2013) Complexity of spectrum activity and benefits of reinforcement learning for dynamic channel selection. IEEE J Sel Areas Commun 31(11):2237–2248
MacKenzie AB, DaSilva LA (2006) Game theory for wireless engineers. Synth Lect Commun 1(1):1–86
Maharjan S, Zhang Y, Gjessing S (2011) Economic approaches for cognitive radio networks: a survey. Wirel Pers Commun 57(1):33–51
Mähönen P (2004) Cognitive trends in making: future of networks. In: Personal, indoor and mobile radio communications, 2004. PIMRC 2004. 15th IEEE international symposium on, vol 2. IEEE, pp 1449–1454
Mähönen P, Petrova M, Riihijärvi J, Wellens M (2006) Cognitive wireless networks: your network just became a teenager. In: Proceedings of IEEE INFOCOM
Masonta M, Mzyece M, Ntlatlapa N (2013) Spectrum decision in cognitive radio networks: a survey. IEEE Commun Surv Tutor 15(3):1088–1107. doi:10.1109/SURV.2012.111412.00160
Ma M, Tsang DH (2008) Joint spectrum sharing and fair routing in cognitive radio networks. In: Consumer communications and networking conference, 2008. CCNC 2008. 5th IEEE, IEEE, pp 978–982
McKeown N, Anderson T, Balakrishnan H, Parulkar G, Peterson L, Rexford J, Shenker S, Turner J (2008) Openflow: enabling innovation in campus networks. ACM SIGCOMM Comput Commun Rev 38(2):69–74
Mehboob U, Qadir J, Ali S, Vasilakos A (2014) Genetic algorithms in wireless networking: techniques, applications, and issues. arXiv:1411.5323
Meshkati F, Poor HV, Schwartz SC (2007) Energy-efficient resource allocation in wireless networks. Sig Process Mag IEEE 24(3):58–68
Mir AK, Akram A, Ahmed E, Qadir J, Baig A (2012) Unified channel assignment for unicast and broadcast traffic in cognitive radio networks. In: Local computer networks workshops (LCN Workshops), 2012 IEEE 37th conference on, IEEE, pp 799–806
Misra S, Oommen BJ (2005) Dynamic algorithms for the shortest path routing problem: learning automata-based solutions. IEEE Trans Syst Man Cybern Part B Cybern 35(6):1179–1192
Mitchell TM (1997) Machine learning. McGraw Hill
Mitola J III (2006) Cognitive radio architecture: the engineering foundations of radio XML. Wiley, New York
Naserian M, Tepe K (2009) Game theoretic approach in routing protocol for wireless ad hoc networks. Ad Hoc Netw 7(3):569–578
Nasipuri A, Das SR (1999) On-demand multipath routing for mobile ad hoc networks. In: Computer communications and networks, 1999. Proceedings of the eight international conference on, IEEE, pp 64–70
Nemirovski A (2006) Advances in convex optimization: conic programming. In: Proceedings of the international congress of mathematicians: Madrid: invited lectures, pp 413–444
Neyman A, Sorin S (2003) Stochastic games and applications, vol 570. Springer, Berlin
Nicopolitidis P, Papadimitriou GI, Pomportsis AS, Sarigiannidis P, Obaidat MS (2011) Adaptive wireless networks using learning automata. IEEE Wirel Commun 18(2):75–81
Nisan N (2007) Algorithmic game theory. Cambridge University Press, Cambridge, MA
Nurmi P (2004) Modelling routing in wireless ad hoc networks with dynamic bayesian games. In: Sensor and ad hoc communications and networks, 2004. IEEE SECON 2004. 2004 First annual IEEE communications society conference on. IEEE, pp 63–70
Nurmi P (2007) Reinforcement learning for routing in ad hoc networks. In: WiOpt. Citeseer, pp 1–8

Palomar DP, Chiang M (2006) A tutorial on decomposition methods for network utility maximization. IEEE J Sel Areas Commun 24(8):1439–1451
Papadimitriou CH, Steiglitz K (1998) Combinatorial optimization: algorithms and complexity. Courier Dover Publications, New York
Papadimitriou CH, Tsitsiklis JN (1999) The complexity of optimal queuing network control. Math Oper Res 24(2):293–305
Park C-H, Kim S-W, Lim S-M, Song M-S (2007) HMM based channel status predictor for cognitive radio. In: Microwave conference, 2007. APMC 2007. Asia-Pacific. IEEE, pp 1–4
Pavlidou F-N, Koltsidas G (2008) Game theory for routing modeling in communication networks—a survey. J Commun Netw 10(3):268–286
Pefkianakis I, Wong SH, Lu S (2008) Samer: spectrum aware mesh routing in cognitive radio networks. In: New Frontiers in dynamic spectrum access networks, 2008. DySPAN 2008. 3rd IEEE symposium on. IEEE, pp 1–5
Peshkin L, Savova V (2002) Reinforcement learning for adaptive routing. In: Neural networks, 2002. IJCNN '02. Proceedings of the 2002 international joint conference on, vol 2, pp 1825–1830
Pitsillides A, Stylianou G, Pattichis CS, Sekercioglu A, Vasilakos A (2000) Bandwidth allocation for virtual paths (BAVP): investigation of performance of classical constrained and genetic algorithm based optimisation techniques. In: INFOCOM 2000. Nineteenth annual joint conference of the IEEE computer and communications societies. Proceedings. IEEE, vol 3. IEEE, pp 1501–1510
Powell WB (2007) Approximate dynamic programming: solving the curses of dimensionality, vol 703. Wiley, New York
Purian FK, Farokhi F, Nadooshan RS (2013) Comparing the performance of genetic algorithm and ant colony optimization algorithm for mobile robot path planning in the dynamic environments with different complexities. J Acad Appl Stud 3(2):29–44
Puterman ML (2009) Markov decision processes: discrete stochastic dynamic programming, vol 414. Wiley, New York
Qadir J, Baig A, Ali A, Shafi Q (2014) Multicasting in cognitive radio networks: algorithms, techniques and protocols. arXiv preprint arXiv:1406.xxx
Qiu L, Yang YR, Zhang Y, Shenker S (2006) On selfish routing in internet-like environments. IEEE/ACM Trans Netw (TON) 14(4):725–738
Rabiner L, Juang B (1986) An introduction to hidden Markov models. IEEE ASSP Mag 3(1):4–16
Raghunathan V, Kumar P (2009) Wardrop routing in wireless networks. IEEE Trans Mobile Comput 8(5):636–652
Rahman MA, Caleffi M, Paura L (2012) Joint path and spectrum diversity in cognitive radio ad-hoc networks. EURASIP J Wirel Commun Netw 1:1–9
Raniwala A, Chiueh T-C (2005) Architecture and algorithms for an IEEE 802.11-based multi-channel wireless mesh network. In: INFOCOM 2005. 24th Annual joint conference of the IEEE Computer and communications societies. Proceedings IEEE, vol 3, IEEE, pp 2223–2234
Rathnasabapathy B, Gmytrasiewicz P (2003) Formalizing multi-agent POMDP's in the context of network routing. In: System sciences, 2003. Proceedings of the 36th annual Hawaii international conference on, IEEE
Reddy YB (2008) Detecting primary signals for efficient utilization of spectrum using Q-learning. In: Information technology: new generations, 2008. ITNG 2008. Fifth international conference on. IEEE, pp 360–365
Ren W, Xiao X, Zhao Q (2009) Minimum-energy multicast tree in cognitive radio networks. In: Signals, systems and computers, 2009 Conference record of the forty-third Asilomar conference on. IEEE, pp 312–316
Resende MG, Pardalos PM (2006) Handbook of optimization in telecommunications. Springer, Berlin
Rezek I, Leslie DS, Reece S, Roberts SJ, Rogers A, Dash RK, Jennings NR (2008) On similarities between inference in game theory and machine learning. J Artif Intell Res (JAIR) 33:259–283
Rondeau TW (2007) Application of artificial intelligence to wireless communications. Ph.D. thesis, Virginia Polytechnic Institute and State University
Rondeau TW, Bostian CW (2009) Artificial intelligence in wireless communications. Artech House, London
Rondeau TW, Le B, Rieser CJ, Bostian CW (2004a) Cognitive radios with genetic algorithms: intelligent control of software defined radios. In: Software defined radio forum technical conference. Citeseer, pp C3–C8
Rondeau TW, Rieser CJ, Gallagher TM, Bostian CW (2004b) Online modeling of wireless channels with hidden Markov models and channel impulse responses for cognitive radios. In: Microwave symposium digest, 2004 IEEE MTT-S international, vol 2. IEEE, pp 739–742

Ross SM (1970) Applied probability models with optimization applications. Courier Dover Publications, New York
Roughgarden T (2007) Routing games. Algorithmic Game Theory 18:459–484
Rozner E, Seshadri J, Mehta Y, Qiu L (2009) Soar: simple opportunistic adaptive routing protocol for wireless mesh networks. IEEE Trans Mobile Comput 8(12):1622–1635
Russell S, Norvig P (1995) Artificial intelligence: a modern approach, vol 74. Prentice-Hall Englewood Cliffs, NJ
Saad W, Han Z, Poor HV, Basar T, Song JB (2012) A cooperative bayesian nonparametric framework for primary user activity monitoring in cognitive radio networks. IEEE J Sel Areas Commun 30(9):1815–1822
Saleem Y, Rehmani MH (2014) Primary radio user activity models for cognitive radio networks: a survey. J Netw Comput Appl 43:1–16
Sammut C, Webb GI (2010) Encyclopedia of machine learning. Springer, Berlin
Sampath A, Yang L, Cao L, Zheng H, Zhao BY (2008) High throughput spectrum-aware routing for cognitive radio networks. In: Proceedings of the IEEE Crowncom
Sekercioğlu Y, Pitsillides A, Vasilakos A (2001) Computational intelligence in management of atm networks. Soft Comput 5(4):257–263
Shakkottai S, Shakkottai SG, Srikant R (2008) Network optimization and control. Now Publishers Inc, Boston
Shatila HS (2012) Adaptive radio resource management in cognitive radio communications using fuzzy reasoning. Ph.D. thesis, Virginia Polytechnic Institute and State University
Shenker SJ (1995) Making greed work in networks: a game-theoretic analysis of switch service disciplines. IEEE/ACM Trans Netw (TON) 3(6):819–831
Shetty N, Pollin S, Pawelczak P (2009) Identifying spectrum usage by unknown systems using experiments in machine learning. In: Wireless communications and networking conference, 2009. WCNC 2009. IEEE. IEEE, pp 1–6
Shiang H-P, Van Der Schaar M (2008) Delay-sensitive resource management in multi-hop cognitive radio networks. In: New Frontiers in dynamic spectrum access networks, 2008. DySPAN 2008. 3rd IEEE symposium on. IEEE, pp 1–12
Shi Y, Hou YT (2008) A distributed optimization algorithm for multi-hop cognitive radio networks. In: INFOCOM 2008. The 27th conference on computer communications, IEEE. IEEE, pp 1292–1300
Song Y, Xie J (2012) A distributed broadcast protocol in multi-hop cognitive radio ad hoc networks without a common control channel. In: INFOCOM, 2012 proceedings IEEE. IEEE, pp 2273–2281
Song Y, Liu L, Ma H, Vasilakos AV (2014) A biology-based algorithm to minimal exposure problem of wireless sensor networks. IEEE Trans Netw Serv Manag 11(3):417–430. doi:10.1109/TNSM.2014.2346080
Srinivasan V, Nuggehalli P, Chiasserini C-F, Rao RR (2003) Cooperation in wireless ad hoc networks. In: INFOCOM 2003. Twenty-second annual joint conference of the IEEE computer and communications. IEEE Societies, vol 2. IEEE, pp 808–817
Srivastava V, Neel JO, MacKenzie AB, Menon R, DaSilva LA, Hicks JE, Reed JH, Gilles RP (2005) Using game theory to analyze wireless ad hoc networks. IEEE Commun Surv Tutor 7(1–4):46–56
Subramanian AP, Buddhikot MM, Miller S (2006) Interference aware routing in multi-radio wireless mesh networks. In: Wireless mesh networks, 2006. WiMesh 2006. 2nd IEEE workshop on, IEEE, pp 55–63
Sun L, Zheng W, Rawat N, Sawant V, Koutsonikolas D (2013) Performance comparison of routing protocols for cognitive radio networks. In: MASCOTS 2013. IEEE, IEEE
Sutton RS, Barto AG (1998) Reinforcement learning: an introduction. Cambridge University Press, Cambridge, MA
Szepesvári C (2010) Algorithms for reinforcement learning. Synth Lect Artif Intell Mach Learn 4(1):1–103
Talk on Deconstructing Reinforcement Learning' by Richard Sutton at ICML (2009) http://videolectures.net/icml09_sutton_itdrl/. Accessed 17 Aug 2013
Tassiulas L, Ephremides A (1992) Stability properties of constrained queueing systems and scheduling policies for maximum throughput in multihop radio networks. IEEE Trans Autom Control 37(12):1936–1948
Teh YW, Jordan MI, Beal MJ, Blei DM (2006) Hierarchical dirichlet processes. J Am Stat Assoc 101(476):1566–1581
Tehrani P, Zhao Q (2013) Distributed online learning of the shortest path under unknown random edge weights. In: Acoustics, speech and signal processing (ICASSP), 2013 IEEE international conference on. IEEE, pp 3138–3142
Tesauro G (2002) Programming backgammon using self-teaching neural nets. Artif Intell 134(1):181–199
Thomas RW, Friend DH, DaSilva LA, MacKenzie AB (2006) Cognitive networks: adaptation and learning to achieve end-to-end performance objectives. IEEE Commun Mag 44(12):51–57
Thomas RW, Friend DH, DaSilva LA, MacKenzie AB (2007) Cognitive networks. Springer, Berlin

Tsagkaris K, Katidiotis A, Demestichas P (2008) Neural network-based learning schemes for cognitive radio systems. Comput Commun 31(14):3394–3404
Tuggle RE (2010) Cognitive multipath routing for mission critical multi-hop wireless networks. In: System theory (SST), 2010 42nd Southeastern symposium on. IEEE, pp 66–71
Tumuluru VK, Wang P, Niyato D (2010) A neural network based spectrum prediction scheme for cognitive radio. In: Communications (ICC), 2010 IEEE international conference on. IEEE, pp 1–5
Urpi A, Bonuccelli M, Giordano S et al (2003) Modelling cooperation in mobile ad hoc networks: a formal description of selfishness. In: WiOpt'03: modeling and optimization in mobile, ad hoc and wireless networks
Van der Schaar M, Fu F (2009) Spectrum access games and strategic learning in cognitive radio networks for delay-critical applications. Proc IEEE 97(4):720–740
Vasilakos AV, Papadimitriou GI (1995) A new approach to the design of reinforcement schemes for learning automata: stochastic estimator learning algorithm. Neurocomputing 7(3):275–297
Vasilakos A, Ricudis C, Anagnostakis K, Pedryca W, Pitsillides A (1998) Evolutionary-fuzzy prediction for strategic QoS routing in broadband networks. In: Fuzzy systems proceedings, 1998. IEEE World Congress on computational intelligence. The 1998 IEEE international conference on, vol 2, IEEE. pp 1488–1493
Waldrop MM (1992) Complexity: the emerging science at the edge of order and chaos, vol 12. Simon & Schuster, New York
Wang W, Li X-Y, Wang Y (2004) Truthful multicast routing in selfish wireless networks. In: Proceedings of the 10th annual international conference on mobile computing and networking. ACM, pp 245–259
Wang B, Ji Z, Liu KR, Clancy TC (2009) Primary-prioritized Markov approach for dynamic spectrum allocation. IEEE Trans Wirel Commun 8(4):1854–1865
Wang B, Wu Y, Liu K (2010) Game theory for cognitive radio networks: an overview. Comput Netw 54(14):2537–2561
Watkins CJ, Dayan P (1992) Q-learning. Mach Learn 8(3–4):279–292
Wei G, Ling Y, Guo B, Xiao B, Vasilakos AV (2011) Prediction-based data aggregation in wireless sensor networks: combining grey model and Kalman filter. Comput Commun 34(6):793–802
Wellens M, Mähönen P (2010) Lessons learned from an extensive spectrum occupancy measurement campaign and a stochastic duty cycle model. Mobile Netw Appl 15(3):461–474
Wellens M, RiihijäRvi J, Mähönen P (2009a) Empirical time and frequency domain models of spectrum use. Phys Commun 2(1):10–32
Wellens M, Riihijarvi J, Mähönen P (2009b) Modelling primary system activity in dynamic spectrum access networks by aggregated on/off-processes. In: Sensor, mesh and ad hoc communications and networks workshops, 2009. SECON Workshops' 09. 6th Annual IEEE communications society conference on, IEEE, pp 1–6
Xiang L, Luo J, Vasilakos A (2011) Compressed data aggregation for energy efficient wireless sensor networks. In: Sensor, mesh and ad hoc communications and networks (SECON), 2011 8th annual IEEE communications society conference on. IEEE, pp 46–54
Xia B, Wahab MH, Yang Y, Fan Z, Sooriyabandara M (2009) Reinforcement learning based spectrum-aware routing in multi-hop cognitive radio networks. In: Cognitive radio oriented wireless networks and communications, 2009. CROWNCOM'09. 4th International conference on. IEEE, pp 1–5
Xing X, Jing T, Huo Y, Li H, Cheng X (2013) Channel quality prediction based on Bayesian inference in cognitive radio networks. In: IEEE INFOCOM
Xu Y, Anpalagan A, Wu Q, Shen L, Gao Z, Wang J (2013) Decision-theoretic distributed channel selection for opportunistic spectrum access: strategies, challenges and solutions. IEEE Commun Surv Tutor 15(4):1689–1713
Yang Z, Cheng G, Liu W, Yuan W, Cheng W (2008) Local coordination based routing and spectrum assignment in multi-hop cognitive radio networks. Mobile Netw Appl 13(1–2):67–81
Yau K-L, Komisarczuk P, Teal PD (2010) Applications of reinforcement learning to cognitive radio networks. In: Communications workshops (ICC), 2010 IEEE international conference on. IEEE, pp 1–6
Yau K-LA, Komisarczuk P, Teal PD (2012) Reinforcement learning for context awareness and intelligence in wireless networks: review, new features and open issues. J Netw Comput Appl 35(1):253–267
Youssef M, Ibrahim M, Abdelatif M, Chen L, Vasilakos AV (2014) Routing metrics of cognitive radio networks: a survey. IEEE Commun Surv Tutor 16(1):92–109
Yuan Y, Yang H, Wong SH, Lu S, Arbaugh W (2005) Romer: resilient opportunistic mesh routing for wireless mesh networks. In: IEEE workshop on wireless mesh networks (WiMesh), vol 6
Zeng Y, Xiang K, Li D, Vasilakos AV (2013) Directional routing and scheduling for green vehicular delay tolerant networks. Wirel Netw 19(2):161–173
Zhang GP (2007) Avoiding pitfalls in neural network research. IEEE Trans Syst Man Cybern Part C Appl Rev 37(1):3–16

Zhang Y, Lee C, Niyato D, Wang P (2013) Auction approaches for resource allocation in wireless systems: a survey. IEEE Commun Surv Tutor 15(3):1020–1041. doi:10.1109/SURV.2012.110112.00125

Zhao Q, Tong L, Swami A, Chen Y (2007) Decentralized cognitive MAC for opportunistic spectrum access in ad hoc networks: a POMDP framework. IEEE J Sel Areas Commun 25(3):589–600

Zhao N, Li S, Wu Z (2012) Cognitive radio engine design based on ant colony optimization. Wirel Pers Commun 65(1):15–24

Zheng C, Sicker D (2013) A survey on biologically inspired algorithms for computer networking. IEEE Commun Surv Tutor 15(3):1160–1191

Zheng K, Li H, Qiu RC, Gong S (2012) Multi-objective reinforcement learning based routing in cognitive radio networks: walking in a random maze. In: Computing, networking and communications (ICNC), 2012 international conference on. IEEE, pp 359–363

Zhu G-M, Akyildiz IF, Kuo G-S (2008) Stod-rp: A spectrum-tree based on-demand routing protocol for multi-hop cognitive radio networks. In: Global telecommunications conference, 2008. IEEE GLOBECOM 2008. IEEE, pp 1–5

Zhu Q, Yuan Z, Song JB, Han Z, Basar T (2010) Dynamic interference minimization routing game for on-demand cognitive pilot channel. In: Global telecommunications conference (GLOBECOM 2010), 2010 IEEE. IEEE, pp 1–6

Zorzi M, Rao RR (2003) Geographic random forwarding (GeRaF) for ad hoc and sensor networks: multihop performance. IEEE Trans Mobile Comput 2(4):337–348