# Cognitive Networks: Adaptation and Learning to Achieve End-to-End Performance Objectives

*Ryan W. Thomas, Daniel H. Friend, Luiz A. DaSilva, and Allen B. MacKenzie, Virginia Tech*

## ABSTRACT

In this article we advance the idea of a *cognitive network*, capable of perceiving current network conditions and then planning, learning, and acting according to end-to-end goals. Cognitive networks are motivated by the complexity, heterogeneity, and reliability requirements of tomorrow's networks, which are increasingly expected to self-organize to meet user and application objectives. We compare and contrast cognitive networks with related research on cognitive radios and cross-layer design. By defining cognitive networks, examining their relationship to other technologies, discussing critical design issues, and providing a framework for implementation, we aim to establish a foundation for further research and discussion.

## INTRODUCTION

Current data networking technology limits a network's ability to adapt, often resulting in suboptimal performance. Limited in state, scope and response mechanisms, the network elements (consisting of nodes, protocol layers, policies, and behaviors) are unable to make intelligent adaptations. Communication of network state information is stifled by the layered protocol architecture, making individual elements unaware of the network status experienced by other elements. Any response that an element may make to network stimuli can only be made in the context of its limited scope. The adaptations that are performed are typically reactive, taking place after a problem has occurred. In this article we advance the idea of cognitive networks, which have the promise to remove these limitations by allowing networks to observe, act, and learn in order to optimize their performance.

Cognitive networks are motivated by complexity. Particularly in wireless networks, there has been a trend towards increasingly complex, heterogeneous, and dynamic environments. While wired networks can also take on any of these characteristics, and are not excluded from potential cognitive network applications, because of the internode interactions and the size of the system state space, wireless networks are a natural focus of research in complex networks. Previous wireless research into cognitive radio and crosslayer design have addressed some of these issues, but have shortcomings from the network perspective. Cognitive networks represent a new scope and approach to dealing with this complexity. This article is written to provide the reader with a primer on the cognitive network concept, as envisioned by the authors. As such, it begins by first explaining the need for cognitive networks, how they are defined, and possible applications for the technology. Then the article examines how cognitive networks are related to, but distinct from, previous work in cognitive radios and cross-layer design. A practical discussion of the implementation of a cognitive network and important areas of future work closes the article.

### DEFINITION

We suggest the following definition for a cognitive network, first described by us in [1]: A cognitive network is a network with a cognitive process that can perceive current network conditions, and then plan, decide, and act on those conditions. The network can learn from these adaptations and use them to make future decisions, all while taking into account end-to-end goals.

The cognitive aspect of this definition is similar to that used to describe cognitive radio and broadly encompasses many simple models of cognition and learning. More critical to the definition are the network and end-to-end aspects. Without the network and end-to-end scope, the system is perhaps a cognitive radio or layer, but not a cognitive network. Here, end-to-end denotes all the network elements involved in the transmission of a data flow. For a unicast transmission, this might include the subnets, routers, switches, virtual connections, encryption schemes, mediums, interfaces, or waveforms, to mention just a few. The end-to-end goals are what gives a cognitive network its network-wide scope, separating it from other adaptation approaches, which have only a local, single element scope.


0163-6804/06/$20.00 © 2006 IEEE

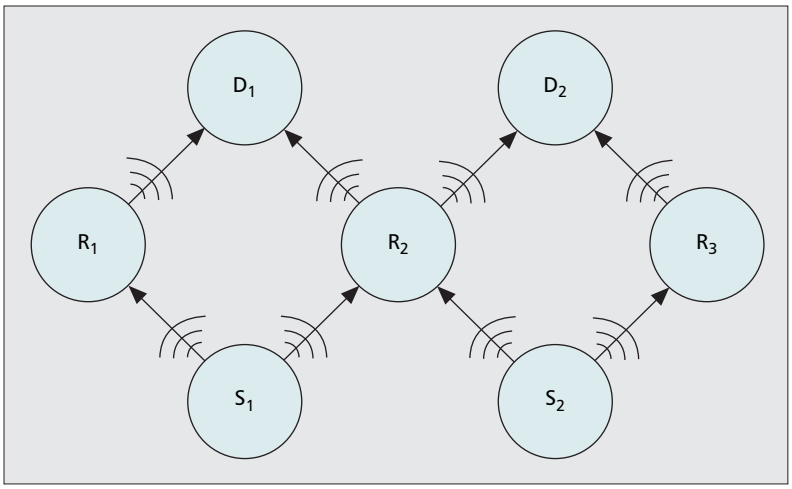


**Figure 1.** *Simple relay network.*

## Motivation and Requirements

The overall goal of any technology is that it meet some need in the best way possible for the least cost. With the first half of this goal in mind, a cognitive network should provide, over an extended period of time, better end-to-end performance than a noncognitive network. Cognition can be used to improve the performance of resource management, quality of service (QoS), security, access control, or many other network goals. Cognitive networks are only limited in application by the adaptability of the underlying network elements and the flexibility of the cognitive process.

In examining the second half of this goal, the cost (in terms of overhead, architecture, and operation) must justify the performance. In almost all cases, implementing a cognitive network requires a system that is more complex than a noncognitive network. Thus for cognitive networks to be justifiable, the performance improvement must outweigh these additional costs. For certain environments, such as static wired networks with predictable behavior, it may not make sense to convert to cognitive behaviors. Other environments, such as heterogeneous wireless networks, may be ideal candidates for cognition.

Cognitive networks should use observations (or proxy observations) of network performance as input to a decision-making process and then provide output in the form of a set of actions that can be implemented in the modifiable elements of the networks. Ideally, a cognitive network should be forward-looking, rather than reactive, and attempt to adjust to problems before they occur. Additionally, the architecture of a cognitive network should be extensible and flexible, supporting future improvements, network elements, and goals.

Cognitive networks require a software adaptable network (SAN) to implement the actual network functionality and allow the cognitive process to adapt the network. Similarly to cognitive radio, which depends on software defined radio (SDR) to modify aspects of radio operation (e.g. time, frequency, bandwidth, code, spatiality, waveform), a SAN depends on a network that has one or more modifiable elements. Practically, this means that a network must be able to modify one or several layers of the network stack in its member nodes. A simple example of a SAN could be a wireless network with directional antennas (antennas with the ability to direct their maximum receive or transmit gain to various points of rotation). A more complex example would incorporate more modifiable aspects at various layers of the protocol stack, such as MAC adaptations or routing control.

## A Simple Example

As an example of the need for end-to-end rather than just link adaptations, consider an ad hoc data session between a source node, $S_1$, and a destination node $D_1$ as shown in Fig. 1. The source node must route traffic through intermediate nodes $R_1$ and $R_2$ acting as regenerative relays. Node $S_1$ performs a link adaptation by choosing the relay node based on the set of minimum hop routes to $D_1$ and the probability of link outage. Based on the simple network in Fig. 1, nodes $R_1$ and $R_2$ are both in the set of minimum hop relays on routes to $D_1$. Therefore, node $S_1$ selects the link on which to transmit by observing the outage probabilities on the links to $R_1$ and $R_2$ and selecting the link with the lower outage probability. From the standpoint of the link layer in node $S_1$, this guarantees that the transmitted packets have the highest probability of arriving correctly at the relay node. However, it does not guarantee anything about the end-to-end performance, that is, the total outage probability from $S_1$ to $D_1$.

In contrast to the link adaptation, the cognitive network uses observations from all nodes to compute the total path outage probabilities from $S_1$ to $D_1$ through $R_1$ and $R_2$. This shows the benefit of a more global view, but there is another advantage to the cognitive network, the learning capability. Suppose that the learning mechanism measures throughput from the source to its destination in order to judge the effectiveness of previous decisions, and suppose that nodes $S_1$ and $S_2$ are both routing their traffic through $R_2$ because this satisfies the minimum outage probability objective.

Now suppose that $R_2$ becomes congested because of a large volume of traffic coming from $S_2$. This becomes apparent to the cognitive process in the throughput reported by $S_1$ and $S_2$. The learning mechanism recognizes that the prior solution is no longer optimal and directs the cognitive process toward another solution. The cognitive network does not explicitly know that there is congestion at node $R_2$ because we have not included this as an observation. Nevertheless, it is able to infer from the reduced throughput that there may be a problem. It is then able to respond to the congestion, perhaps by routing traffic through $R_1$ and/or $R_3$. This example illustrates the potential of cognitive networks in optimizing end-to-end performance as well as reacting to unforeseen circumstances. The cognitive network goes beyond the purely algorithmic approach of the underlying routing protocol and finds efficient operating points even when unexpected events occur.

## FOUNDATIONS AND RELATED WORK

Having defined a cognitive network, it is helpful to review some existing research areas related to the cognitive network concept. We take a look at two areas in particular, cognitive radio and cross-layer design. We also examine other current research into the cognitive network concept.

### COGNITIVE RADIO

***Shared Attributes with Cognitive Networks*** — The 50 percent correlation in nomenclature would of itself imply some degree of commonality, and it can certainly be argued that research in cognitive radio has sparked the formulation of the cognitive network concept. What cognitive radios and cognitive networks do share is the cognitive process that is the heart of the performance optimizations. An essential part of the cognitive process is the capability to learn from past decisions and use this learning to influence future behavior. Both are goal-driven and rely on observations paired with knowledge of node capabilities to reach decisions. Knowledge in cognitive radio is contained within a modeling language such as radio knowledge representation language (RKRL) [2]. A network-level equivalent must exist for the cognitive network to be goal oriented and achieve context awareness, two attributes which it shares with a cognitive radio. A cognitive radio requires tunable parameters which define the optimization space of the cognitive process. These tunable parameters are ideally provided by SDR. The concept of the SAN is the cognitive network analog of SDR. Therefore, both technologies employ a software tunable platform that is controlled by the cognitive process.

***Differences from Cognitive Networks*** — Cognitive networks are clearly delineated from cognitive radios by the scope of the controlling goals. Goals in a cognitive network are based on end-to-end network performance, whereas cognitive radio goals are localized only to the radio's user. These end-to-end goals are derived across the network from operators, users, applications, and resource requirements. This difference in goal scope from local to end-to-end enables the cognitive network to operate more easily across all layers of the protocol stack. Current research in cognitive radio emphasizes interactions with the physical layer, which limits the direct impact of changes made by the cognitive process to the radio itself and other radios to which it is directly linked. Agreement with other radios on parameters which must match for successful link communication is reached through a process of negotiation. Since changes in protocol layers above the physical layer tend to impact more nodes in the network, the cognitive radio negotiation process would have to be expanded to include all nodes impacted by the change. However, because the negotiation process is unable to assign precedence to radios' desires without goals of a broader scope, achieving agreement among multiple nodes may be a slow process. For the same reason, the compromise can be expected to result in suboptimal network performance. In contrast, cognitive networks are more cooperative in nature, since the performance is referenced to the end-to-end goals and nodes within a single cognitive element must cooperate to enact decisions.

Another significant difference between cognitive radios and cognitive networks is the degree of heterogeneity that is supported. Cognitive networks are applicable to both wired and wireless networks, whereas cognitive radios of course apply only to wireless networks. Since the cognitive network may span multiple wired and wireless mediums, it is useful for optimizing performance for these heterogeneous types of networks, which are generally difficult to integrate.

The fact that a cognitive network is composed of multiple nodes also adds a degree of freedom in how the cognitive processing is performed compared to cognitive radio. A cognitive network has the option to implement a centralized cognitive process, a fully distributed cognitive process, or a partially distributed cognitive process.

*The fact that a cognitive network is composed of multiple nodes also adds a degree of freedom in how the cognitive processing is performed compared to cognitive radio. A cognitive network has the option to implement a centralized cognitive process, a fully distributed cognitive process, or a partially distributed cognitive process.*

### CROSS-LAYER DESIGN

***Shared Attributes with Cross-Layer Design*** — Designs that violate the traditional layered approach by direct communication between nonadjacent layers or sharing of internal information between layers are called cross-layer designs [3]. Cognitive networks indirectly share information that is not available externally in the strict layered architecture. Therefore, cognitive networks do perform cross-layer design.

The common theme is that observations are made available externally and some adaptation is performed at a layer other than the layer providing the observation. In a cognitive network, protocol layers provide observations of current conditions to the cognitive process. The cognitive process then determines what is optimal for the network and changes the configurations of network elements' protocol stacks.

***Differences from Cross-Layer Design*** — Despite similarities, cognitive networks reach far beyond the scope of cross-layer designs. The cognitive network must support trade-offs between multiple goals and in order to do so performs multi-objective optimizations (MOO), whereas cross-layer designs typically perform single objective optimizations. Cross-layer designs perform independent optimizations that do not account for the network-wide performance goals. Trying to achieve each goal independently is likely to be suboptimal, and as the number of cross-layer designs within a node grows, conflicts between the independent adaptations may lead to adaptation loops [4]. This pitfall is avoided in a cognitive network by jointly considering all goals in the optimization process.

The ability to learn is another significant difference. The cognitive network learns from prior decisions and applies the learning to future decisions. Cross-layer designs are memoryless adaptations that will respond the same way when presented with the same set of inputs, regardless of how poorly the adaptation may have performed in the past. The ability to learn from past behavior is particularly important in light of

*In a cognitive network, protocol layers provide observations of current conditions to the cognitive process. The cognitive process then determines what is optimal for the network and changes the configurations of network elements' protocol stacks.*

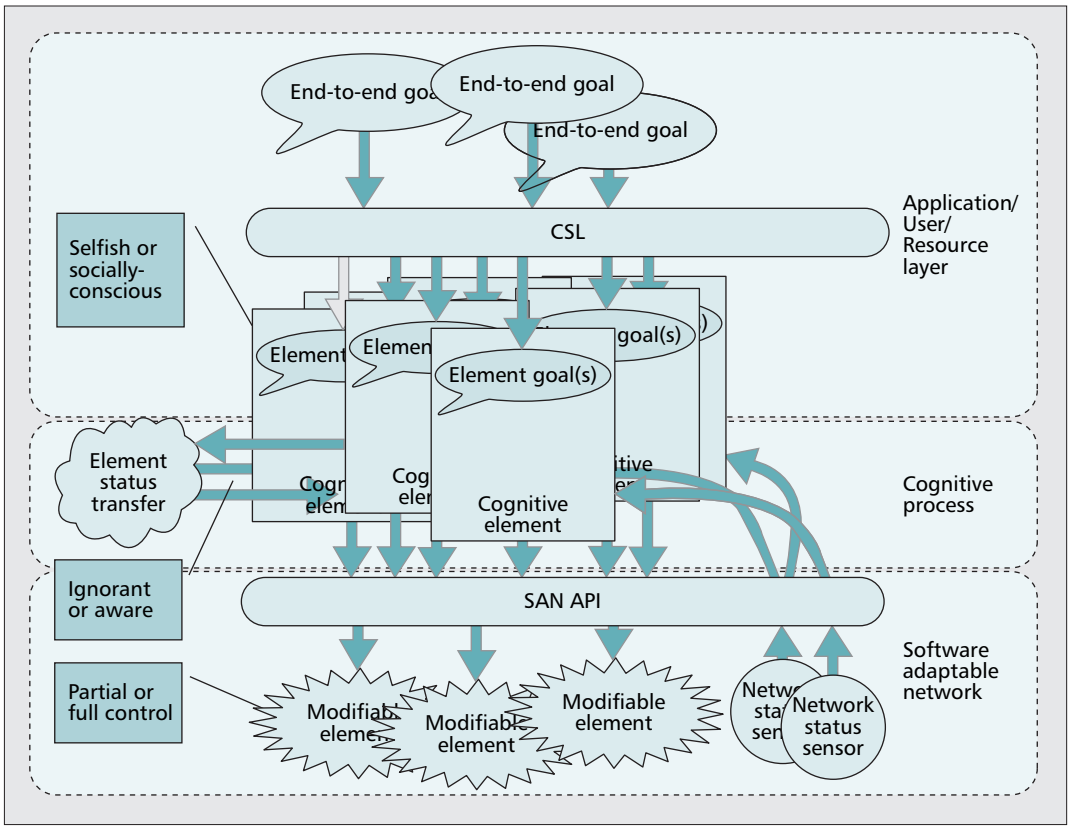


**Figure 2.** *The cognitive framework and critical design issues.*

the fact that our understanding of the interaction between layers is limited.

Again, the scope of the goals and observations sets cognitive networks apart from cross-layer design. The observations used by the cognitive process span multiple nodes and the optimization is performed with the goals of all nodes in mind whereas cross-layer design is node-centric. This global information allows the cognitive process to adapt in ways which simply are not possible when nodes have limited visibility into the state of other nodes in the network, as is the case with cross-layer design.

### Cognitive Networks

This article is the first to provide a comprehensive investigation of the motivations, architecture, and design of cognitive networks. However, other work exists in the literature that is related to the cognitive network concept. Perhaps the first mention of a cognitive network-like concept was in the article by Clark *et al*. on the network knowledge plane, which they define as a "pervasive system within the network that builds and maintains high level models of what the network is supposed to do" [5]. Our work in [1] was the first to define the term "cognitive network" and examine its functionality. More recently, the End-to-End Reconfigurability Project II ($E^2R$ II) [6], m@ANGEL platform [7], CTVR at Trinity College [8], and the Institute for Wireless Networks at RWTH Aachen University [9] have proposed architectures at various degrees of maturity for end-to-end oriented, autonomous networks. Unlike the general, bottom-up approach that this work uses, these architectures are typically more focused on a particular application (such as 4G cellular or wireless), implementation (such as the cognitive mechanism or associated APIs), or problem (such as mobility or management).

## Implementation

In order to synthesize the preceding concepts and components into an actual cognitive network, we investigate how a cognitive network should be implemented. We construct a framework for the cognitive process and identify the critical features of this architecture.

A common model of cognition is the three-level theory [10]. The model is often summarized as consisting of behavioral, functional, and physical layers. The behavioral level determines what observable actions the system produces, the functional layer determines how the system processes the information provided to it, and the physical layer comprises the neuro-physiology of the system.

From this concept, we draw a three-layer framework, with each layer corresponding roughly to the layers in the model described above. At the top layer are the goals of the system and elements in the network that define the behavior of the system. These goals feed into the cognitive process, which computes the actions the system takes. The SAN is the physical control of the system, providing the action space for the cognitive process. This framework is illustrated in Fig. 2.

In our framework we consider the possibility of a cognitive process consisting of one or more cognitive elements, operating in some degree between autonomy and full cooperation. If there

is a single cognitive element, it may still be physically distributed over one or more nodes in the network. If there are multiple elements, they may be distributed over a subset of the nodes in the network, on every node in the network, or several cognitive elements may reside on a single node. In this respect, the cognitive elements operate in a manner similar to a software agent.

### User/Application/Resource Requirements

The top-level component of the cognitive network framework includes the end-to-end goals, cognitive specification language (CSL), and cognitive element goals. The end-to-end goals, which drive the behavior of the entire system, are put forth by the network users, applications or resources. Without end-to-end goals guiding network behavior, undesired consequences may arise. This is a problem with many crosslayer designs and is explored in some depth in [4], which illustrates unintended end-to-end interactions in a MAC/PHY cross-layer design.

Like most engineering problems, there is likely to be a trade-off for every goal optimized on. When a cognitive network has multiple objectives it will not be able to optimize all metrics indefinitely, eventually reaching a point in which one metric cannot be optimized without affecting another. In order to determine the correct Pareto optimal front (the set of actions from which no goal can be improved without worsening another), each cognitive element must have an understanding of all end-to-end goals and their constraints.

To connect the goals of the top-level users of the network to the cognitive process, an interface layer must be developed. In a cognitive network, this role is performed by the CSL, which provides behavioral guidance to the elements by translating the end-end goals to local element goals. Less like the RKRL proposed by Mitola for cognitive radio and more like a QoS specification language [11], the CSL maps end-to-end requirements to underlying mechanisms. Unlike a QoS specification language, the mechanisms are adaptive to the network capabilities as opposed to fixed. Furthermore, a CSL must be able to adapt to new network elements, applications and goals, some of which may not even be imagined yet. Other requirements may include support for distributed or centralized operation, including the sharing of data between multiple cognitive elements.

The scope of the cognitive network is broader than that of the individual network elements; it operates within the scope of a data flow, which may include many network elements. For a distributed cognitive process, the cognitive elements associated with each flow or network element may act selfishly and independently (in the context of the entire network) to achieve local goals, or act in a altruistic manner to achieve network-wide goals. The job of converting the end-to-end goals to these local element goals is often a difficult problem.

An interesting question is how much of a performance difference there is between these two modes (altruistic and selfish) of operation. The performance difference between cognitive elements following selfish, local goals and acting in an altruistic, network-wide mode of operation is called *the price of anarchy* in [12]. Here, the term was defined as the difference in performance between a network run by an all-knowing benevolent dictator (that can specify the "correct solution" for current flows to achieve optimal performance) and one governed purely by selfish anarchy. The concept of the price of anarchy in a larger sense will guide the development of cognitive networks by indicating the scope in which cognitive processes work best. If it turns out that the network performance is significantly poorer when acting selfishly than acting in an altruistic manner, the cognitive network may need to provide more centralized guidance or an appropriate incentive structure to the network elements.

### Cognitive Process

There does not seem to be a common, accepted definition of what cognition means when applied to communication technologies. The term cognitive, as used in this article, follows closely in the footsteps of the definition used by Mitola in [2] and the even broader definition of the FCC. The former incorporates a spectrum of cognitive behaviors, from goal-based decisions to proactive adaptation. Here, we associate cognition with *machine learning*, which is broadly defined in [13] as any algorithm that "improves its performance through experience gained over a period of time without complete information about the environment in which it operates." Underneath this definition, many different kinds of artificial intelligence, decision making, and adaptive algorithms can be placed, giving cognitive networks a wide scope of possible mechanisms to use for learning.

Learning serves to complement the objective optimization part of the cognitive process by retaining the effectiveness of past decisions under a given set of conditions. Determining the effectiveness of past decisions requires a feedback loop to measure the success of the chosen solution in meeting the objectives defined. This is retained in memory, so that when similar circumstances are encountered in the future, the cognitive process will have some idea of where to start or what to avoid.

The effect of a cognitive process's decisions on the network performance depends on the amount of network state information available to it. In order for a cognitive network to make a decision based on end-to-end goals, the cognitive elements must have some knowledge of the network's current state and other cognitive element states. If a cognitive network has knowledge of the entire network's state, decisions at the cognitive element level should be at least as good, if not better (in terms of the cognitive element goals) than those made in ignorance. For a large, complex system such as a computer network, it is unlikely that the cognitive network would know the total system state. There is often a high cost to communicate this information beyond those network elements requiring it, meaning a cognitive network will have to work with less than a full picture of the network status.

Filtering and abstraction may be used to further reduce the amount of information that must be exchanged and to avoid unnecessary triggering of the cognitive process. Filtering means that observations made by the node may be held back from the cognitive process if they are

*An interesting question is how much of a performance difference there is between these two modes (altruistic and selfish) of operation. The performance difference between cognitive elements following selfish, local goals and acting in an altruistic, network-wide mode of operation is called the price of anarchy*

The performance costs (or benefits) of a distributed system knowing less than the whole state of the system could be termed the price of ignorance. Of course, the price of ignorance would only account for costs due to decisions made on insufficient information.

deemed irrelevant. Thus, the nodes themselves make some determination of what is important to the cognitive process. Filtering rules may be identified at design time with additional rules specified in real-time as the cognitive process determines its sensitivity to various types of observations and disseminates filtering rules accordingly. The goal of abstraction is to reduce the number of bits required to represent an observation. Observations or collections of observations made by a node are reported to the cognitive process at a higher level of abstraction than what is available within the node. Abstractions may also be specified at design time with real-time adaptations by the cognitive process. This reductionism resulting from filtering and abstraction carries risk because it may mask information that the cognitive process needs to operate correctly. Therefore, care should be taken in defining the abstractions or filtering.

There are interesting cases, however, where selfish elements acting under the guise of ignorance may act in ways more optimal (in terms of the network goals) than those with complete information. A real-world example of this might be a crowd leaving a theater with two exits during a fire. One exit is short and narrow; the other is long and wide. The management desires to empty the theater as quickly as possible, but each patron will take only the short exit and ignore the long exit until they perceive that crowding has slowed the short exit to the same delay as the long exit. If, because of the darkness in the theater, some patrons misread the crowding at the short exit and determine prematurely to take the longer way out of the theater, they will detrimentally affect their exit time (as compared to the time it would have taken them to go out the short exit). However, this may have a positive effect on the average time for patrons to exit (and by turn, the time to exit the entire theater) since by taking the long way out prematurely, these patrons reduce the crowding at the short exit and allow it to exit patrons even faster.

The performance costs (or benefits) of a distributed system knowing less than the whole state of the system could be termed the *price of ignorance*. Of course, the price of ignorance would only account for costs due to decisions made on insufficient information. Other issues that might arise in systems with limited information, such as the reduction in overhead from reduced data collection and distribution or the increase in critical nodes' knowledge of important information represent different design issues in the engineering trade-space.

### Software Adaptable Network

The SAN consists of the application programming interface (API), modifiable network elements, and network status sensors. The SAN is really a separate research area, just as the design of the SDR is separate from the development of the cognitive radio, but at a minimum the cognitive process needs to be aware of the API and the interface it presents to the modifiable elements. Just like the other aspects of the framework, the API should be flexible and extensible. Continuing the analogy with SDRs, an existing system that is analogous to the API is the software communications architecture (SCA) used in the Joint Tactical Radio System (JTRS).

Another responsibility of the SAN is to notify the cognitive process of the status of the network. To what level and detail is a function of the filtering and abstraction being applied. The status of the network is the source of the feedback used by the cognitive process, and is composed of status sensor observations and communication with other cognitive elements. Possible observations may be local (such as bit error rate, battery life, or data rate), nonlocal (such as end-to-end delay and clique size), or compilations of different local observations.

The modifiable elements can include any object or element used in a network, although it is unlikely that all elements in a SAN would be modifiable. Each element should have public and private interfaces to the API, allowing it to be manipulated by both the SAN and the cognitive process. Modifiable elements are assumed to have a set of states that they can operate in, and a "solution" for a cognitive process consists of a set of these states that, when taken together, meet the end-to-end requirements of the system. At any given instant the set of all possible combinations of states $S$ can be partitioned into two subsets. The first, $S'$, contains all possible combinations of states that meet the end-to-end requirements and the second, $\bar{S}'$, consists of all combinations that do not meet these requirements. Of those in $S'$, some may meet the requirements better than others, making them preferred solutions.

A cognitive network attempts to choose a set of states for the modifiable elements that exists in $S'$. This means that should the network be in state $\bar{S}'$, or some suboptimal state in $S'$, the cognitive process attempts to move the system state to an optimal solution. With cognitive control over every multistate element, the cognitive process can potentially set the system to any state; an ideal cognitive process could set the state to the optimal solution. If the system has only a few points of cognitive control, or chooses not to use all its control, then the cognitive process has to use the functionality and interactions of the noncognitive aspects of the network to set the system state. Like the hole at the bottom of a funnel, certain system states will be basins of attraction, pulling the system towards them from a variety of starting states. If a system has several attractors and some are more optimal than others, then a few points of cognitive control may be enough to draw the system out of one attractor and into another. This is analogous to a watershed, in which moving the source of water a few miles may be enough to change what river the water will finally flow into. The *price of control* is the difference in performance between a system that exercises full control over the network and one that depends on the state transitions of noncognitive elements to control aspects of the network.

## Future Work and Conclusion

The previous sections make a case for the "what and why" of cognitive networks, and address how they are designed and implemented. We now examine major issues that need to be addressed in order to move from concept to

reality. Beyond the three prices of anarchy, ignorance, and control, which exist in any design, there are several open-ended questions.

There is an implicit assumption in this article that the cognitive network implements configuration changes synchronously. The details of actually making this happen with high reliability are likely to be complex. The implications of nodes' switching configuration at different times may be worse than if no adaptation had been performed at all. Also, the varying topology of the network means that not all nodes will receive notification of configuration changes at the same time. A possible approach is to require nodes to be synchronized to some common time reference and to issue configuration changes with respect to the time reference. However, this adds complexity to the nodes and still does not guarantee that messages will not be lost, resulting in stranded nodes. It also forces the network to delay its adaptation to the conditions that spawned the configuration change.

Due to the autonomy of each, there is potential conflict between what a cognitive radio and a cognitive network each control if there is not an integrated architecture. One approach is to turn all control over to the cognitive network, but this is probably unwise. The reason is that the cognitive network has to limit its observations as much as possible just to make cognitive processing for a network feasible. This leaves much detailed local information out of the cognitive network picture. This detailed local information may be used by the cognitive radio to further optimize its performance outside the bounds of what is controlled by the cognitive network. In order to allow this, the cognitive radio must know what it is allowed to change and what is in the hands of the cognitive network. A potential solution is to use the cognitive radios as cognitive elements, allowing the cognitive network to establish regulatory policy for the cognitive radios in a real-time manner, leaving the cognitive radio to perform further optimization under these constraints.

This article has laid the groundwork for the concept of a cognitive network and has proposed a definition for the term. Additionally, the cognitive network concept was compared against both cognitive radio and cross-layer design. Finally, a framework for cognitive networks was presented, and critical themes and issues were identified in the design and implementation of a cognitive network. While a significant amount of work remains to be done to make cognitive networks a reality, the rising complexity of networks and the need to manage this complexity makes the concept timely and attractive.

## References


[1] R. W. Thomas, L. A. DaSilva, and A. B. Mackenzie, "Cognitive Networks," *Proc. IEEE DySPAN 2005*, Nov. 2005, pp. 352–60.

[2] J. Mitola, *Cognitive Radio: An Integrated Agent Architecture for Software Defined Radio*, Ph.D. thesis, Royal Inst. Technology, 2000.

[3] V. Srivastava and M. Motani, "Cross-Layer Design: A Survey and the Road Ahead," *IEEE Commun. Mag.*, vol. 43, no. 12, 2005, pp. 112–19.

[4] V. Kawadia and P. R. Kumar, "A Cautionary Perspective on Cross-layer Design," *IEEE Wireless Commun.*, vol. 12, no. 1, 2005, pp. 3–11.

[5] D. D. Clark *et al.*, "A Knowledge Plane for the Internet," *Proc. SIGCOMM '03*, New York, NY, 2003, pp. 3–10.

[6] D. Bourse *et al.*, "End-to-End Reconfigurability (E2R II): Management and Control of Adaptive Communication Systems," IST Mobile Summit 2006, June 2006.

[7] P. Demestichas *et al.*, "m@ANGEL: Autonomic Management Platform for Seamless Cognitive Connectivity to the Mobile Internet," *IEEE Commun. Mag.*, vol. 44, no. 6, June 2006, pp. 118–27.

[8] P. Sutton, L. E. Doyle, and K. E. Nolan, "A Reconfigurable Platform for Cognitive Networks," *Proc. CROWNCOM 2006*, 2006.

[9] P. Mähönen *et al.*, "Cognitive Wireless Networks: Your Network Just Became a Teenager," *Proc. IEEE INFOCOM 2006*, 2006.

[10] P. N. Johnson-Laird, *The Computer and the Mind*, Cambridge, MA: Harvard Univ. Press, 1988.

[11] J. Jin and K. Nahrstedt, "QoS Specification Languages for Distributed Multimedia Applications: A Survey and Taxonomy," *IEEE Multimedia*, vol. 11, no. 3, 2004, pp. 74–87.

[12] C. H. Papadimitriou, "Algorithms, Games, and the Internet," *Proc. STOC 2001*, 2001.

[13] M. A. L. Thathachar and P. S. Sastry, *Networks of Learning Automata*, Kluwer, 2004.


## Biographies


RYAN W. THOMAS [StM] (rwthomas@vt.edu) is a doctoral candidate in Virginia Tech's Bradley Department of Electrical and Computer Engineering. He received an M.S. in computer engineering from the Air Force Institute of Technology at Wright-Patterson AFB, Ohio, where he was a distinguished graduate, and a B.S. from Harvey Mudd College, Claremont, California. He previously worked at the Air Force Research Laboratory, Sensors Directorate as a digital array engineer. His research focuses on the design, architecture, and evaluation of cognitive networks. He is a member of the Wireless @ Virginia Tech research group.

DANIEL H. FRIEND [StM] (dhfriend@vt.edu) is a doctoral student in Virginia Tech's Bradley Department of Electrical and Computer Engineering, where he is also a Bradley Fellow. He received B.S. and M.S. degrees in electrical engineering from Brigham Young University, Provo, Utah, where he was a National Merit Scholar. Following graduation he worked for Motorola in Phoenix, Arizona for seven years, leaving as a senior staff systems engineer. He spent the summer of 2006 at MIT Lincoln Laboratory performing research on stochastic modeling of heterogeneous wireless ad hoc networks. His primary research interests are in wireless ad hoc networks, cognitive network architecture, and distributed computing and distributed artificial intelligence methods for cognitive networks. He is a member of the Wireless @ Virginia Tech research group.

LUIZ A. DASILVA [SM] (ldasilva@vt.edu) joined Virginia Tech's Bradley Department of Electrical and Computer Engineering in 1998, where he is now an associate professor. He received a Ph.D. in electrical engineering at the University of Kansas and previously worked for IBM for six years. His research focuses on performance and resource management in wireless and mobile ad hoc networks. He is currently researching the application of game theory to model MANETs, topology control, cooperation, and reputation management in heterogeneous ad hoc networks, energy-aware multicast route discovery, and cognitive networks. He has published more than 50 refereed papers in journals and major conferences in the communications and computer areas. He is a member of the Wireless @ Virginia Tech research group, the ASEE, and ACM. In 2006 he was named a College of Engineering Faculty Fellow at Virginia Tech. He frequently teaches distance and distributed learning courses on network architecture and protocols, and mobile and wireless networking.

ALLEN B. MACKENZIE [M] (mackenab@vt.edu) has been an assistant professor in Virginia Tech's Bradley Department of Electrical and Computer Engineering since 2003. He joined Virginia Tech after receiving a Ph.D. from Cornell University and (in 1999) a B.Eng. from Vanderbilt University, both in electrical engineering. His research focuses on adaptation in wireless communications systems and networks. His current research interests include cognitive radio and cognitive network algorithms, architectures, and protocols, and the analysis of cooperation in such systems and networks. He is a member of the ASEE and ACM. In 2006 he received the Dean's Award for Outstanding New Assistant Professor in the College of Engineering at Virginia Tech.



*The implications of nodes' switching configuration at different times may be worse than if no adaptation had been performed at all. Also, the varying topology of the network means that not all nodes will receive notification of configuration changes at the same time.*