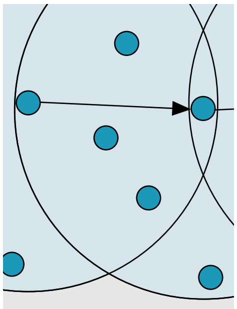
ACCEPTED FROM OPEN CALL

A CAUTIONARY PERSPECTIVE ON CROSS-LAYER DESIGN

VIKAS KAWADIA, BBN TECHNOLOGIES
P. R. KUMAR, UNIVERSITY OF ILLINOIS AT URBANA-CHAMPAIGN



Recently, in an effort to improve the performance of wireless networks, there has been increased interest in protocols that rely on interactions between different layers. However, such cross layer design can run at cross purposes with sound and longer term architectural principles, and can lead to various negative consequences.

ABSTRACT

Recently, in an effort to improve the performance of wireless networks, there has been increased interest in protocols that rely on interactions between different layers. However, such cross-layer design can run at cross purposes with sound and longer-term architectural principles, and lead to various negative consequences. This motivates us to step back and reexamine holistically the issue of cross-layer design and its architectural ramifications. We contend that a good architectural design leads to proliferation and longevity, and illustrate this with some historical examples. Even though the wireless medium is fundamentally different from the wired one, and can offer undreamt of modalities of cooperation, we show that the conventional layered architecture is a reasonable way to operate wireless networks, and is in fact optimal up to an order. However the temptation and perhaps even the need to optimize by incorporating cross-layer adaptation cannot be ignored, so we examine the issues involved. We show that unintended cross-layer interactions can have undesirable consequences on overall system performance. We illustrate them by certain cross-layer schemes loosely based on recent proposals. We attempt to distill a few general principles for cross-layer design. Moreover, unbridled cross-layer design can lead to spaghetti design, which can stifle further innovation and be difficult to upkeep. At a critical time when wireless networks may be on the cusp of massive proliferation, the architectural considerations may be paramount. We argue that it behooves us to exercise caution while engaging in cross-layer design.

INTRODUCTION

Recently there has been increased interest in protocols for wireless networks that rely on significant interactions between various layers of the network stack. Generically termed cross-layer design, many of these proposals are aimed at achieving performance improvements, although, as we argue, often at the cost of good architectural design. This has prompted us to take a step back and examine the whole issue of cross-layer design and its architectural ramifications, and more generally the issue of an appropriate architecture for wireless networks.

We begin by expounding on the importance of a good architectural design. Several historical examples demonstrate the proliferation and longevity a sound architectural design can spark. John von Neumann's architecture for computer systems, the layered architecture for the Internet, Shannon's architecture for communication systems, and the plant controller feedback paradigm in control systems are some successful examples.

A significant measure of credit for the Internet revolution can be attributed to its layered architecture, to the extent that it has become the de facto architecture for wireless systems. Wireless systems, however, are different, and there is no concept of a "link." A transmission is just a spatiotemporal footprint of radio energy. Nodes along a path in a wireless network may operate on this energy in a variety of ways. In addition to the wireline strategy of decode and forward [1], there are various other possibilities like amplify and forward [2], and cooperative interference cancellation [1]. The first issue is therefore to choose a sound design from the infinite options, and it is indeed gratifying that decode and forward is order-optimal for wireless networks as well [1]. By order-optimal we mean that the network capacity achieved by this scheme, as a function of the number of nodes, is as good as any other scheme up to a constant. We illustrate how this naturally translates to the layered architecture being a reasonable first template for designing wireless networks.

However, various optimization opportunities do present themselves through cross-layer design, and the temptation cannot be ignored. Caution needs to be exercised, though. Once the

This material is based on work partially supported by DARPA under Contract Nos. N00014-O1-1-0576 and F33615-O1-C1905, USARO under Contract Nos. DAAD19-00-1-0466 and DAAD19-01010-465, AFOSR under Contract No. F49620-02-10217, DARPA/AFOSR under Contract No. F49620-02-1-0325, and NSF under Contract No. NSF ANT 02-21357. Any opinions, findings, and conclusions or recommendations expressed in this publication are those of the authors and do not necessarily reflect the views of the above agencies.

1536-1284/05/$20.00 © 2005 IEEE

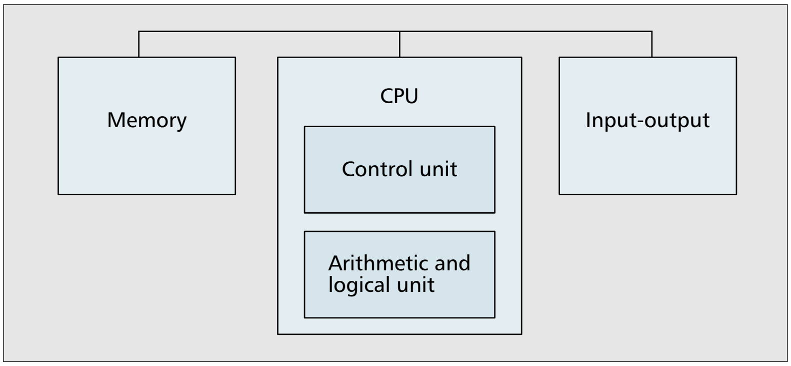


Figure 1. *The von Neumann architecture for computer systems.*

layering is broken, the luxury of designing a protocol in isolation is lost, and the effect of any single design choice on the whole system needs to be considered. Cross-layer design can create loops, and it is well known from control theory that stability then becomes a paramount issue. Compounding this is the fact that some interactions are not easily foreseen. Cross-layer design can thus potentially work at cross purposes; the "law of unintended consequences" can take over if one is not careful, and a negative effect on system performance is possible. We illustrate this by a few examples loosely based on recently proposed schemes. The first involves an adaptive rate MAC protocol, and the second nested adaptation of transmit power.

Design in the presence of interacting dynamics needs care all on its own. Besides stability there is also the issue of robustness. Techniques such as timescale separation may need to be employed to separate interactions, and an accompanying theoretical framework may be needed.

Unbridled cross-layer design can also lead to a spaghetti design. Further design improvements may then become difficult since it will be hard to address satisfactorily how a new modification will interact with the multiplicity of already existing dynamics. Second, it will be hard to upkeep. Rather than modifying just one layer, the entire system may need to be replaced.

All of the foregoing is just a manifestation of the larger and ever present tension between performance and architecture. Performance optimization can lead to short-term gain, while architecture is usually based on longer-term considerations. At a time when wireless networks may be on the cusp of a takeoff, we argue that the longer-term view is paramount, and that greater caution therefore needs to be exercised when indulging in cross-layer design.

The Importance of Architecture

Architecture in system design pertains to breaking down a system into modular components, and systematically specifying the interactions between the components. The importance of architecture is difficult to overemphasize. Modularity provides the abstractions necessary for designers to understand the overall system. It accelerates development of both design and implementation by enabling parallelization of effort. Designers can focus their effort on a particular subsystem with the assurance that the entire system will interoperate. A good architectural design can thus lead to quick proliferation. Moreover, it can lead to massive proliferation. When subsystems are standardized and used across many applications, the per unit cost is reduced, which in turn increases usage. In contrast, a system that does not capitalize on the amortization of effort by exploiting commonality will essentially need to be hand crafted in every application, resulting in great expense. Architecture also leads to longevity of the system. Individual modules can be upgraded without necessitating a complete system redesign, which would stifle further development and longevity of the system.

On the other hand, taking an architectural shortcut can often lead to a performance gain. Thus, there is always a fundamental tension between performance and architecture, and a temptation to violate the architecture. However, architecture can and should also be regarded as performance optimization, although over a longer time horizon. An architecture that allows massive proliferation can lead to very low per-unit cost for a given performance. More properly, therefore, the tension can be ascribed to realizing short-term vs. longer-term gains. In the particular case of wireless networks, which may be on the cusp of massive proliferation, our contention is that the longer-term view of architecture is paramount.

We begin by illustrating the importance of architecture in the proliferation of technology by considering some important examples.

The von Neumann Architecture

The von Neumann [3] architecture for computer systems, consisting of memory, control unit, arithmetic and logical unit, and input-output devices (Fig. 1), is at the heart of most computer systems. It is so much taken for granted now that it takes an effort to appreciate the insight that invented this architecture. It is obvious only in retrospect. Its most important ramification is that it has decoupled software from hardware. Software designers and hardware designers (Microsoft and Intel, say) can work independently, and still be assured that their products will interoperate as long as they each conform to an abstraction of the other side. Software designers can thus develop complex products without worrying about what hardware they will eventually run on. Similarly, hardware designers need not design their hardware for specific application software. Both are in stark contrast to the early ENJAC, for example. As Valiant [4] notes, it is this von Neumann "bridge" that is responsible for the proliferation of serial computation. Valiant [4] also points out that it is the lack of such an architecture for parallel computation that is one of the reasons it has not proliferated as successfully. Like any good architecture, the von Neumann architecture has withstood the test of time.

The OSI Architecture for Networking

The layered open systems interconnection (OSI) architecture (Fig. 2) for networking, on which the current Internet architecture (Fig. 3) is loosely based, is another successful example. In fact, we contend that the success of the Internet is primarily architectural and only secondarily algorithmic, although this may be controversial. It has enabled diverse networks to be interconnected efficiently. The hierarchy of layers provides natural abstractions to deal with the natural hierarchy present in networks, again a fact that may only be obvious with the benefit of hindsight. The physical layer deals with signals, and provides a service to communicate bits. The data link layer provides the abstraction of a link, and the ability to transmit and receive groups of bits over the link. The network layer introduces the concept of a path or route, which is a sequence of links. The transport layer provides an end-to-end pipe or channel, which may be reliable or not, depending on the protocol used. The remaining three layers are less clearly defined, and have actually been merged in the TCP/IP architecture that developed later. The interactions between the layers is controlled, and conducted primarily through protocol headers each layer prepends to packets.

In addition, the architecture also involves the notion of peer-to-peer protocols (e.g., TCP) that mediate between corresponding layers on different hosts. Given this overarching architecture, designers can afford the luxury of concentrating on designing the protocols for their layer efficiently, with assurance that the overall system will function reasonably well.

Source-Channel Separation and Digital Communication System Architecture

In his seminal work on the problem of information transmission, Shannon [5] determined the capacity of discrete memoryless channels as the supremum of the mutual information between channel input and output. More important than this expression for the capacity, we contend, is his constructive proof for an architecture where the “layers” of source compression, and coding for reliability over an error-prone channel can be separated, and that the latter layer is invariant with respect to sources. Dubbed the source channel separation theorem, it allows us to associate source coding (data compression) with the source and channel coding (error protection) with the channel, thus allowing the same channel plus channel code to serve a large variety of sources without loss of (near) optimality (Fig. 4.) That there is nothing to be gained by designing a code that takes into account the source and channel statistics simultaneously is of profound importance and nonobvious in the history of ideas. This result lies at the heart of the digital communication revolution. In today’s communication systems, source coding is done by a program (gzip or bzip2) in the operating system, whereas channel coding is part of the network interface card designed for a particular channel. It is a huge convenience that network cards do not have to be designed for a specific application. This architecture has thus had important consequences on the development and proliferation of digital communication systems.

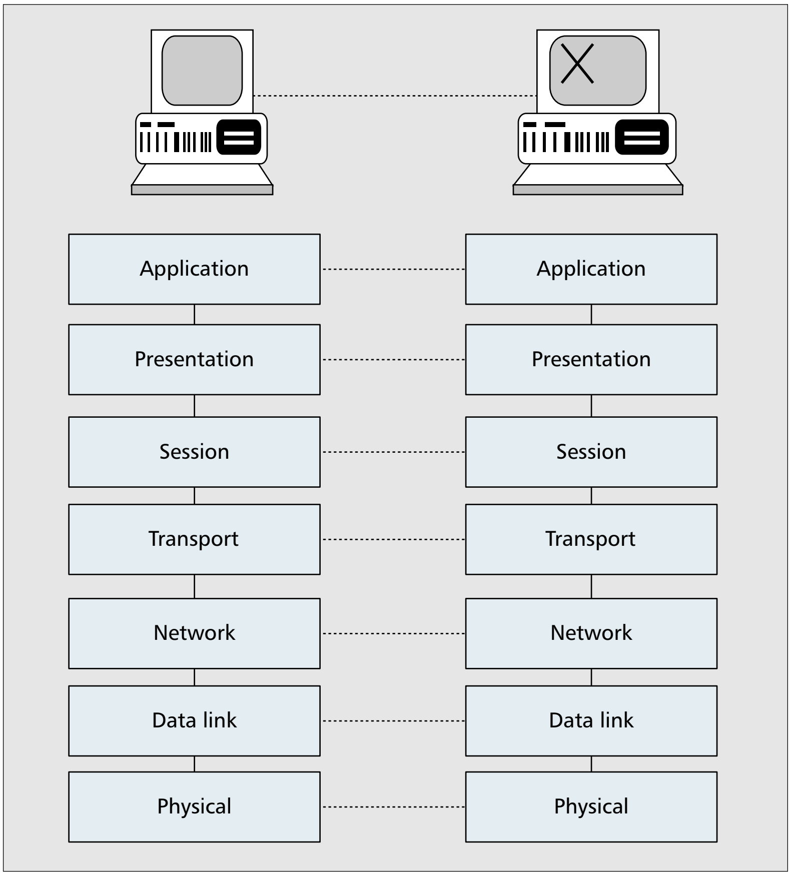


Figure 2. *The layered OSI architecture.*

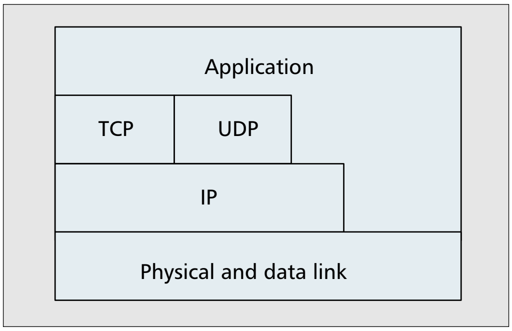
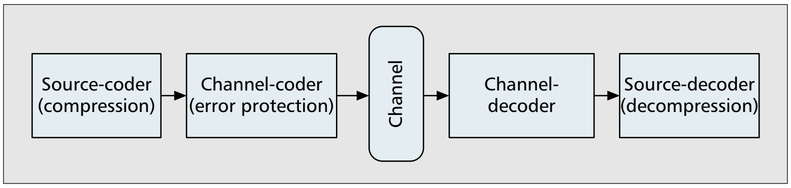


Figure 3. *The current Internet architecture.*

Feedback Control Systems

In control systems, the feedback architecture reigns supreme. While the design of the feedback layer is not independent of the plant, the architecture of the solution is universal. It is one principle that is common to human engineered systems as well as naturally occurring biological systems.

More generally, the importance of architecture is not confined to engineering. The replacement of the barter system by the currency system decoupled trade from the need to have a buyer-cum-seller at each of the two ends of a transaction, and replaced it with a much more flexible system where one end could be just a buyer and the other just a seller. Currency provided a bridge between different transactions by coupling the system with conservation laws. Massive


Source-coder (compression)
Channel-coder (error protection)
Channel
Channel-decoder
Source-decoder (decompression)

■ **Figure 4**. *Source-channel separation and the architecture of digital communication systems.*

proliferation of financial transactions and trade ensued.

Architectural Considerations for Wireless Networks

Successful architectural designs, by their pervasiveness, also influence how designers think, and thus cast a long shadow. The success of the layered architecture for wired networks has had just such a great impact on network design paradigms. It has become the default architecture for designing wireless networks as well. However, it is not at all obvious that this architecture is a priori appropriate for wireless networks. The reason all this needs re-examination is because the wireless medium allows modalities of communication that are simply nonexistent for wired networks. Our first order of business is therefore to re-examine from scratch the whole architectural basis for wireless networks. This will allow us to understand where layering stands.

To do this, we first need to be very clear about the precise description of the current proposal for ad hoc networks being pursued by the Internet Engineering Task Force (IETF) and researchers around the globe. We need to be crystal clear not only about the constructive aspect of the scheme, but also what other possible choices have been foreclosed in the current solution. First, packets are transported over several hops. At each hop, the receiver receives not just the intended signal, but also the superposition of the interfering transmissions of other nodes, plus ambient noise. The interfering transmissions are treated as noise, and the packet is fully decoded. This digital regeneration of a packet lies at the heart of the digital (as opposed to analog) revolution. The regenerated packet is then rebroadcast to the next node, where again it is decoded in the presence of noise with other interfering transmissions simply regarded as noise, and so on.

There are several reasons why this solution is appealing. First, receivers can afford to be simple, since they merely depend on an adequate signal-to-noise ratio. Second, the decode and forward feature along with multihop relaying provide us the abstraction of links. Thus, protocols for wireline networks can be reused for wireless networks.

However, it should be kept in mind that the choice of a multihop architecture with decoding and forwarding at each relay node, while treating all interference as noise, creates certain special problems for wireless networks that need to be solved. For example, since one is treating all interference as noise, one wants to regulate the number of potential interferers in the vicinity of the receiver. This necessitates a medium access control (MAC) protocol whereby one controls the number of interferers. This is rendered difficult by the presence of hidden and exposed terminals, and the need to have a distributed real-time solution.

Another problem we have created is the routing problem. This arises because we wish to employ a multihop scheme involving relays. One then needs to find a sequence of relays from origin to destination: the routing problem. It is useful to note that there would be no routing problem at all if we were to communicate in just one hop. However, then there would be need for other protocols to distinguish between transmissions that are superimposed.

Yet another problem is the power control problem. Because all interference is regarded as noise, it is useful to regulate the powers of transmitters. This necessitates power control. Power control also arises from the need to maximize capacity by spatial reuse of the spectral frequency resource.

Thus, we see that it is our choice of architecture that has created protocol needs — in particular, MAC, routing, and power control — the community is busy solving.

However, there are other choices. For example, interference need not interfere. Consider a powerful transmission interfering with a weak transmission. Since the powerful transmission has a high SNR, it can be successfully decoded. Therefore, it can be subtracted, assuming good channel state information, and the weak signal can be successfully decoded. Thus, the more powerful transmission has not "collided" with the weaker one.

Fundamental Properties of the Wireless Medium

The first point to note is that there is no intrinsic concept of a link between nodes in a wireless medium. Nodes simply radiate energy, and communicate through the superposition of each other's transmissions. There is no notion of a switch that allows a receiver to receive just one transmission while shutting out all others. While this could be regarded as a pure liability — and often is by designers who wish to operate it as a sequence of links — it also offers possibilities unimaginable in wireless networks. One really needs to dispel all existing notions and think afresh about designing wireless networks.

The Infinitude of Possibilities for Operating Wireless Networks

The fact is that the wireless channel potentially offers other modalities of cooperation, as we next examine. For example, node A (or a group of nodes) could cancel the interference created by node B at node C. Thus, A can help C by reducing the denominator in its signal-to-interference plus noise ratio, rather than by boosting the numerator. At the same time, node A could also expend a portion of its energy to relay packets from node D to node E. Moreover, the relaying could be based on amplifying the received signal, rather than first decoding it, regenerating the packet, and then retransmitting it. Indeed, for some parameter values amplify and forward may be preferable to decode and forward [2]. To quote Shakespeare from Hamlet:

"There are more things in heaven and earth, Horatio, than are dreamt of in your philosophy."

How then should nodes cooperate? Indeed, this problem is much richer than simple cross-layer optimization. In fact, there need not even be layers. The search space is not finite dimensional, but infinite dimensional. Some recent results, however, have indicated how wireless networks should be structured, that is, what their architecture should be. Here, the focus is on electronic transfer of information rather than mechanically through mobility.

Theorem 1 (Multihopping Is Order-Optimal [1]) — Suppose there is absorption in the medium, leading to exponential attenuation with distance, or the exponent in polynomial path loss is greater than three. Then multihop decode and forward, treating interference as noise, is order-optimal with respect to the transport capacity when load across nodes can be balanced by multipath routing. Here, transport capacity is defined as the distance weighted sum of rates. Moreover, the transport capacity grows as $\Theta(n)$ (where n is the number of nodes) when the nodes are separated by a minimum positive distance.

In fact, such exponential attenuation is the norm in the scenarios of interest in ad hoc networks [6] rather than the exception. In scenarios with absolutely no exponential attenuation and low path loss exponent, other strategies emerge, as detailed in the theorem below.

Theorem 2 (Architecture Under Low Attenuation) — If there is no absorption and the polynomial path loss exponent is small, other strategies like coherent multistage relaying with interference subtraction can be order-optimal [1] with respect to the scaling law for transport capacity. The transport capacity can grow superlinearly in n in a network of nodes lying along a straight line.

The first result is gratifying. It establishes the order optimality of the multihop decode and forward proposal. It also provides architectural guidance since the multihop decode and forward strategy establishes the need for certain functions in operating a wireless network. Each of these functions can be thought of as a layer in the protocol stack. Decoding at each hop means a data link layer is needed to deal with a one-hop packet transmission, although there are some important differences from the wired case. For example, the medium access sublayer is usually necessary for the wireless case to solve the channel access problem. Also, due to the high error rates in wireless, a somewhat reliable link layer (i.e., hop-by-hop acknowledgment) is necessary. After decoding, the packet needs to be forwarded, which brings in the concept of a path, and hence the network layer. The transport layer is necessary, as before, to provide an end-to-end reliable pipe. Organizing these functions into layers and operating a wireless network based on such a layered stack is a natural way to implement the multihop decode and forward strategy. Thus, a layered architecture can achieve optimal performance, within a constant, with regard to network capacity.

The above result shows that tweaking with layering, and in fact all cross-layer design, can only improve throughput by at most a constant factor (which may still be very important), but cannot result in any unbounded improvements even in networks with large numbers of nodes [1].

Cross-Layer Design Principles

While we have established that a layered architecture is order-optimal and thus a good candidate for a baseline design, there is still the ever present desire, and perhaps need, to optimize. Indeed, several optimization opportunities do present themselves through increased interaction across the layers. The recent past has thus seen a flurry of cross-layer design proposals that explore a much richer interaction between parameters across layers [7].

In evaluating these proposals, the trade-off between performance and architecture needs to be fundamentally considered. As noted above, the performance metrics of the two are different. The former is more short-term, the latter longer-term. Thus, a particular cross-layer suggestion may yield an improvement in throughput or delay performance. To be weighed against this are longer-term considerations.

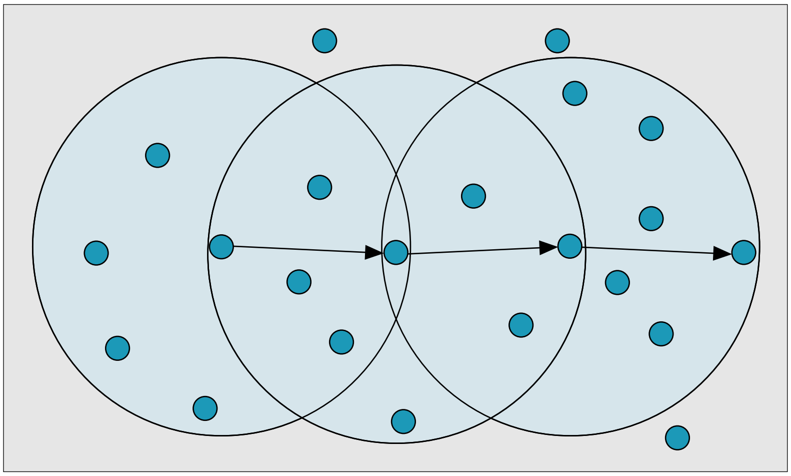
First, while an individual suggestion for cross-layer design in isolation may appear appealing, what is the consequence when other cross-layer interactions are incorporated? Can these interactions work at cross purposes? The point is that every installation of a cross-layer interaction rules out simultaneous use of other potential interactions that might interfere with it. Thus, the case for adoption of a cross-layer interaction must be made more holistically by showing why other mechanisms, potentially conflicting with the suggested one, should not be entertained.

Evaluating the merit of proposals on the overall architecture is thus not an easy task. However, there are some general cautions that can be exercised. We make an attempt to derive some general principles to assist in this process.

Interactions and the Law of Unintended Consequences

The layered architecture and controlled interaction enable designers of protocols at a particular layer to work without worrying about the rest of the stack. Once the layers are broken through cross-layer interactions, this luxury is no longer available to the designer. The interaction can affect not only the layers concerned, but also other parts of the system. In some cases, the implementation itself may introduce dependencies that are not really essential to providing the functionality. It is important to consider the effect of the particular interaction on a remote, seemingly unrelated part of the stack. There could be disastrous unintended consequences on overall performance.

While we have established that a layered architecture is order optimal and thus a good candidate for a baseline design, there is still the ever present desire, and perhaps a need, to optimize. And indeed several optimization opportunities do present themselves.



■ **Figure 5**. *DSDV chooses a small number of long hops, which give a lower data rate when an adaptive rate MAC is used.*

Dependency Graph

Cross-layer design often causes several adaptation loops that are parts of different protocols to interact with each other. To comprehend possible interactions, it is useful to represent the protocols graphically as interactions between parameters. In this *dependency graph* every relevant parameter is a node, and a directed edge indicates the dependency relation between the parameters. By combining the graphs for various protocols, a dependency graph for the entire stack can be obtained.

Timescale Separation and Stability

Certain stability principles can be derived by observing the dependency graph. If a parameter is controlled and used by two different adaptation loops, they can conflict with each other. It is well known from adaptive control theory [8, 9] that such adaptation can benefit from timescale separation. The idea behind it is simple. Consider two entities controlling the same variable but on different timescales. By using the notion of averaging and timescale separation, stability theorems have been proved. The slower system can be regarded as seeing the averaged value of the parameter, and its stability implies the stability of the overall system under some conditions [10].

For every closed loop in the dependency graph consisting of interactions at similar timescales, proofs of stability, as above, will be required. This is often nontrivial and requires significant analytical effort. Designers of cross-layer protocols will need to contend with this.

The Chaos of Unbridled Cross-Layer Design

In addition to these factors, yet another larger issue needs to be considered. What if several cross-layer interactions are implemented? Does one then get unstructured spaghetti-like code that is hard to maintain? Will the resulting system have longevity? Will the need to update the whole system for every modification stifle proliferation? Will this lead to a higher per-unit cost, which eventually is regarded by the end user as lower performance value?

Illustration by Examples

To more concretely illustrate the possibility of unintended interactions, we now present some examples by simulation studies. These examples are loosely based on schemes proposed recently in literature, and are expressly not intended to cast any aspersion on the particular schemes themselves. Indeed, the only reasons for the choice of the particular examples was that we were able to construct scenarios that exhibit the type of negative results we want to demonstrate when they are used in tandem with other existing protocols.

Rate-Adaptive MAC and Minimum-Hop Routing

The idea behind rate-adaptive MAC protocols is to send data at higher rates when the channel quality is good [11]. Such higher rates are achieved by changing the modulation scheme. As we will show, such schemes can have undesirable consequences for the higher layers. In particular we show that when combined with minimum-hop routing — and most routing protocols are indeed minimum hop — they can lead to performance worse than the original system. Briefly, the reason for the adverse behavior is as follows. Minimum-hop routing chooses longer hops, for which the signal strength is lower, and thus the data rate achieved through channel quality adaptation is low.

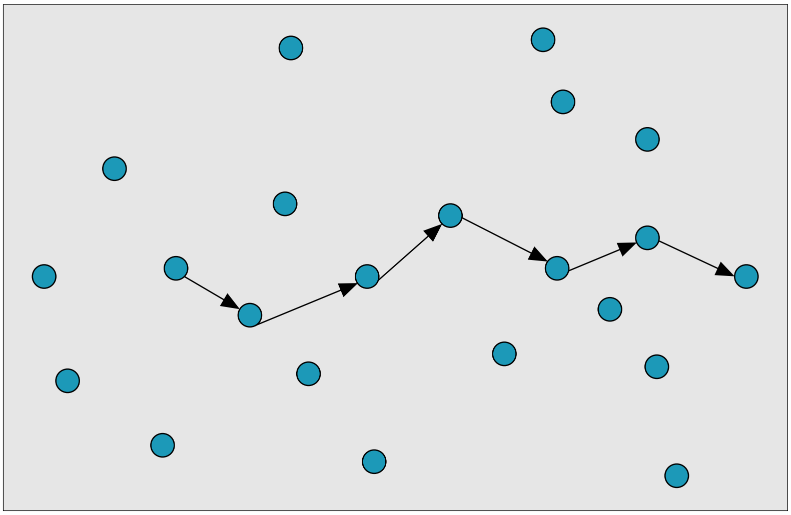
We are not suggesting that adaptive-rate MAC is a bad design. In fact, in this example it may be good, and the problem may lie in the use of a minimum-hop routing protocol. Perhaps a routing protocol using some other metric or a load-adaptive routing protocol may be necessary. The whole point we seek to make is that cross-layer design can lead to unintended adverse interactions, and adequate care is necessary.

The protocol details of the specific adaptive-rate MAC are as follows. It is a modification of the IEEE 802.11 MAC protocol. There are a set of rates available (i.e., modulation schemes), and the transmission rate can be set before transmitting every packet. The request to send/clear to send (RTS/CTS) and broadcast packets are always transmitted at the lowest data rate, called the base rate. The receiver measures the received signal strength of the RTS packet, and figures out the maximum rate at which data can be received given that signal strength. This rate is then communicated to the sender in the CTS packet. The subsequent DATA and acknowledgment (ACK) packets are transmitted at this data rate.

Typically, transmission to nodes close by would occur at higher data rates since the path loss attenuation of the signal is small, while lower data rates would be used for farther nodes for which the received signal is weaker. A modification to this scheme is to opportunistically send more packets when the channel is good [12]. The idea is to "make more hay while the sun shines." Every reservation allocates a fixed time slot, and if the channel is good, there is the opportunity to send more than one packet at higher data rates. These seem like very reasonable cross-layer designs.

However, consider the interaction with higher layers. Suppose we were to use minimum-hop routing, say a protocol like Destination Sequenced Distance Vector (DSDV) [13]. DSDV builds routing tables by sending *hello* packets to neighbors. Hello packets are broadcast packets that contain cumulative routing information (i.e., information that has been gathered from all the neighbors of a node). Since hello packets are broadcast packets, they are sent at the base rate, and thus have a large range. Minimum-hop routing thus chooses the longest possible hops on the path, which causes low received signal strength, which in turn implies a low data rate. This is shown in Fig. 5.

In fact, if we turn off the adaptive-rate MAC and use plain IEEE 802.11 at the highest data rate (i.e., do not send any data when the channel is not good enough to transmit at the highest rate), we can get much better end-to-end throughput. In this case, longer hops simply do not exist, and thus minimum-hop routing is forced to use a larger number of short hops, which provide higher data rates, as depicted in Fig. 6.



■ **Figure 6**. *Plain IEEE 802.11 causes short hops of higher data rate to be used.*

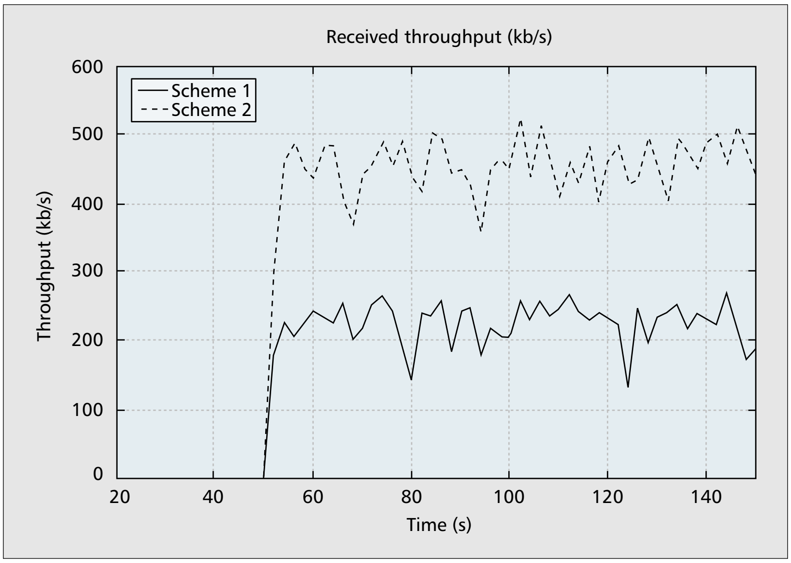
Verification by ns2 Simulations

We now verify the above through ns2 simulations. Scheme 1 uses adaptive-rate MAC with opportunistic scheduling, and our simulation uses, in part, the code provided at [14]. The two-ray-ground propagation model was used for the channel, which results in r^{-4} attenuation between distance and received signal strength in the relatively simple interference model used in ns2. Scheme 1 is set up such that at a fixed transmit power level of 0.28 W, a receiver-transmitter distance of 0–99 m yields a data rate of 11 Mb/s, a distance of 100–198 m yields a data rate of 5.5 Mb/s, while if the distance is between 199-250 m only 2 Mb/s is possible. No communication is possible beyond 250 m. A simple opportunistic policy that gives an equal time share to each data rate is also implemented. Thus, a maximum of five packets are transmitted at 11 Mb/s, three at 5.5 Mb/s, and only one packet if the channel is good enough only for 2 Mb/s.

Scheme 2 is plain vanilla IEEE 802.11 with a data rate of 11 Mb/s. That is, a packet is sent at 11 Mb/s if the receiver-transmitter distance is less than l00 m; otherwise, no communication is possible. Thus, full use of the channel is not made by transmitting at lower data rates when the channel is not good enough. Carrier sensing was turned off for both schemes 1 and 2.

Linear Topology — For the first experiment, 18 nodes are equally spaced in a straight line, in a 1500 m × 200 m area. One TCP connection is run across the end of the chain from node 0 to node 17.

The DSDV routing protocol is used, and a TCP connection is started after the routing tables have stabilized. The received throughput for the two schemes is plotted in Fig. 7. As predicted, scheme 2 outperforms scheme 1. The average end-to-end throughput for scheme 1 is 226 kb/s, whereas it is 455 kb/s for scheme 2, without any adaptation.

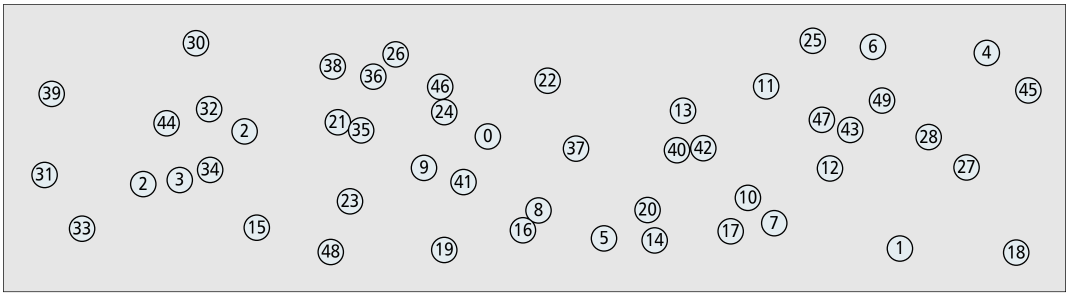


■ **Figure 7**. *Comparing scheme 1 (adaptive-rate MAC) and scheme 2 (plain IEEE 802.11) for the 18-node linear topology.*

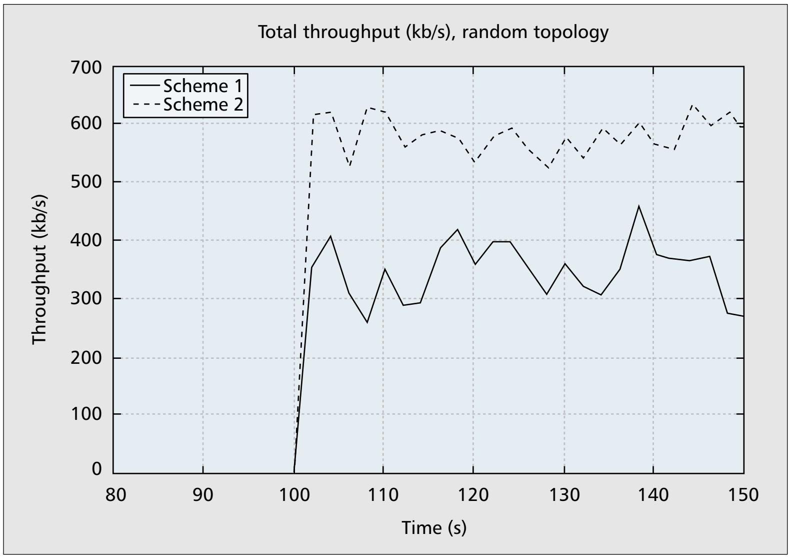
Random Topology — In a second experiment, 50 nodes are randomly located in a 1000 m × 200 m rectangular area (Fig. 8). It was ensured that the topology was connected for scheme 2 as well (i.e., when the transmit range is l00 m). Five TCP connections were then started simultaneously between faraway nodes. The DSDV routing protocol was run as before. The results are shown in Fig. 9. Scheme 2 (i.e., the one without adaptation) again significantly outperforms scheme 1. The total throughput for all the flows is 349 kb/s for scheme 1, while it is 583 kb/s for scheme 2. Thus, the throughput for scheme 2 is better than scheme 1 by a factor of 1.67, whereas the transport capacity is better by a factor of 1.9.

End-to-End Feedback and Topology Control

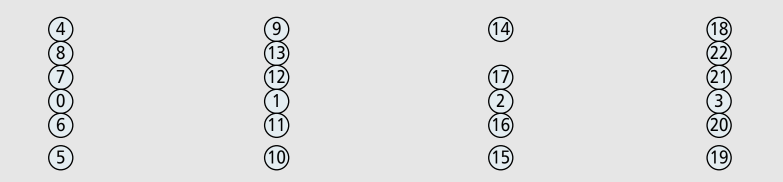
We next consider another example of cross-layer design to illustrate unintended consequences at other layers. The broad idea of the scheme is to adjust the number of neighbors of each node so that TCP performance is enhanced. This is done



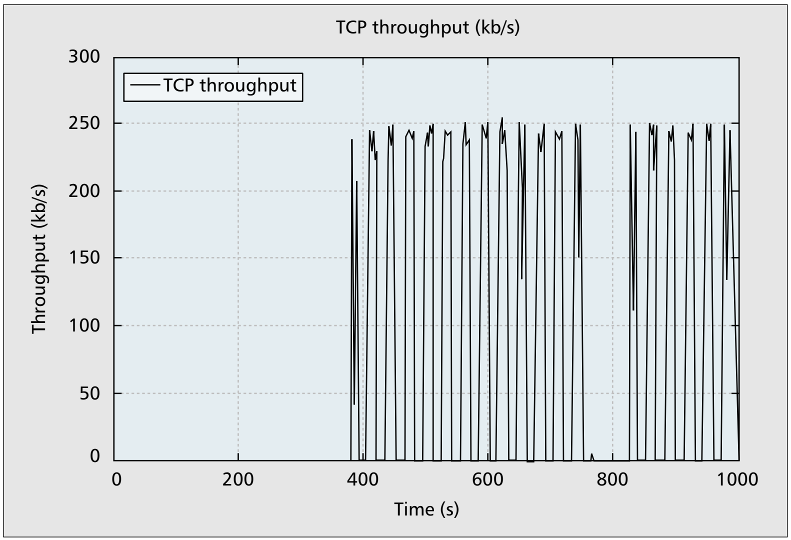
■ **Figure 8**. *50 nodes placed randomly in a 1000 m × 200 m area.*



■ **Figure 9**. *Comparison of Adaptive Rate MAC (Scheme 1) and plain IEEE 802.11 (Scheme 2) for the topology shown in Fig. 8.*



■ **Figure 10**. *Simulation topology for cross-layer design involving power control and end-to-end feedback.*



■ **Figure 11**. *End-to-end throughput for the topology of Fig. 10.*

in two stages. This scheme is a modification of that suggested in [15], and consists of two nested adaptation loops. In the inner loop, each node controls its transmit power so that the number of one-hop neighbors (*out-degree*) is driven to a parameter called *target-degree*. Transmit power is increased by one level if the number of one-hop neighbors is less than target-degree and decreased if it is greater than target-degree. There is an outer loop that sets the value of this parameter, target-degree, based on the average end-to-end network throughput. The action of the previous iterate is repeated (increase by one, decrease by one, or do nothing to the target-degree) if the network throughput increased from the previous iterate; it is reversed if the network throughput decreased from the previous time step. If the network throughput is zero, the network is assumed to be disconnected and the target-degree is increased by one.

The outer loop is operated at a slower timescale than the inner loop to avoid the instability and incoherence that results from two simultaneous adaptation loops interacting with the same phenomenon, as elaborated on earlier. It should be noted that employing just the inner loop alone would be a bad design since it does not even guarantee network connectivity if target-degree is set arbitrarily. The network could thus get stuck in an absorbing state of zero throughput. The outer loop therefore attempts to drive target-degree to a value that maximizes the end-to-end network throughput, and would therefore eventually drive the network out of any possible state of disconnectivity.

The consequences of the two-loop scheme on the performance of higher layers also needs to be examined. As seen below, simulation studies in our deliberately contrived scenarios show that the network may oscillate between connectivity and disconnectivity, which affects TCP performance adversely.

Simulation Studies

The inner adaptation loop adjusts the transmit power level every 15 s to bring the out-degree close to the target-degree. Each node has S transmit power levels, which correspond to transmit ranges of 100 m, 140 m, 180 m, 220 m, and 250 m, respectively, when the two-ray-ground propagation model is used. The outer loop of target-degree adaptation is carried out once every 90 s. The topology consisting of 23 nodes in a 500 m × 500 m area is depicted in Fig. 10. There is one TCP connection from node 0 to node 3.

Initially, the power level of all the nodes is set to the lowest value, and target-degree is set to 2. The network is disconnected in this state and we get zero throughput. The target-degree is gradually increased according to the outer loop until the power level is high enough for the network to be connected. The action is repeated if the throughput keeps on increasing, but after a while it decreases because of increased interference due to the high power levels. Thus, target-degree is decreased and the network goes back into a state of disconnectivity. Consequently, we observe oscillations in the end-to-end throughput, which results in poor average performance over time. These results are shown in Figs. 11 and 12.

Concluding Remarks

There is always a tendency, and in fact a need, to optimize performance in any system. This generally creates tension between performance and architecture. In the case of wireless networks, we currently see this tension manifesting itself in the current interest in cross-layer design.

In venturing into the territory of cross-layer design it may, however, be useful to note some adverse possibilities and exercise appropriate caution. Architecture is important for proliferation of technology, and at a time when wireless networking may be on the cusp of a takeoff, its importance needs to be kept in mind. Unbridled cross-layer design can lead to a spaghetti design, and stifle further innovations since the number of new interactions introduced can be large. Also, such design can stifle proliferation since every update may require complete redesign and replacement. Moreover, cross-layer design creates interactions, some intended, others unintended. Dependency relations may need to be examined, and timescale separation may need to be enforced. The consequences of all such interactions need to be well understood, and theorems establishing stability may be needed. Proposers of cross-layer design must therefore consider the totality of the design, including the interactions with other layers, and also what other potential suggestions might be barred because they would interact with the particular proposal being made. They must also consider the long-term architectural value of the suggestion. Cross-layer design proposals must therefore be holistic rather than fragmenting.

It well behooves us to adopt a cautionary approach to cross-layer design at a critical time in the history of wireless networks when they may well be on the cusp of the massive proliferation that is the objective of us all.

References

[1] L.-L. Xie and P. R. Kumar, "A Network Information Theory for Wireless Communication: Scaling Laws and Optimal Operation," *IEEE Trans. Info. Theory*, vol. 50, no. 5, 2004, pp. 748–67.

[2] B. Schein and R. Gallager, "The Gaussian Parallel Relay Network," *IEEE Int'l. Symp. Info. Theory*, 2000, p. 22.

[3] A. W. Burks, H. H. Goldstine, and J. von Neumann, *John von Neumann Collected Works*, vol. V, A. H. Taub, Ed., Macmillan, 1963.

[4] L. G. Valiant, "A Bridging Model for Parallel Computation," *Commun. ACM*, vol. 33, no. 8, 1990, pp. 103–11.

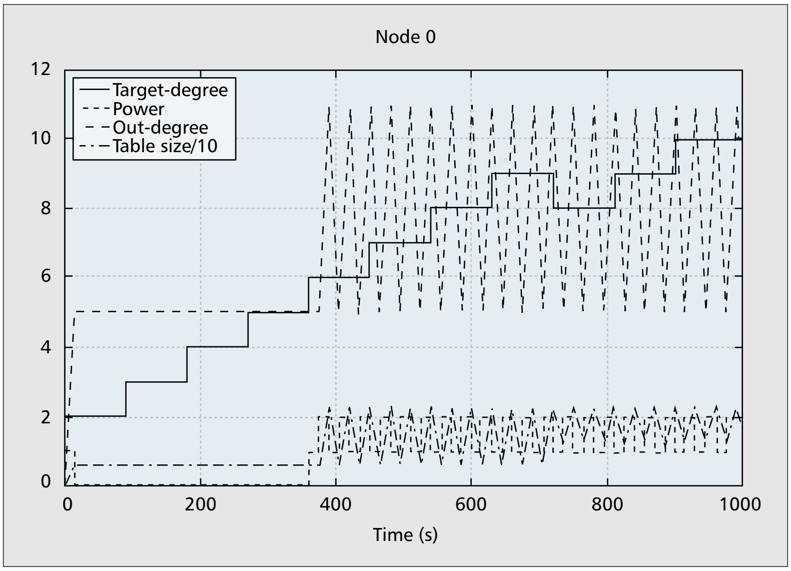


Figure 12. *Parameter traces for the topology of Fig. 10.*

[5] C. Shannon and W. Weaver, *The Mathematical Theory of Communication*, Univ. of IL Press, 1949.

[6] M. Franceschetti, J. Bruck, and L. Schulman, "Microcellular Systems, Random Walks, and Wave Propagation," *Proc. IEEE Int'l. Symp. Antennas and Prop.*, June 2002.

[7] "ICC panel on Defining Cross-layer Design in Wireless Networking," http://www.eas.asu.edurjunshan/-ICC~O3panel.html

[8] K. J. Astrom and B. Wittenmark, *Adaptive Control*, Addison-Wesley, 1995.

[9] P. R. Kumar, "A Survey of Some Results in Stochastic Adaptive Control," *SIAM J. Control and Optimization*, vol. 23, no. 3, 1985, pp. 329–80.

[10] L. Ljung, "Analysis of Recursive Stochastic Algorithms," *IEEE Trans. Auto. Control*, vol. AC-22, 1977, pp. 551–75.

[11] G. Holland, N. Vaidya, and P. Bahl, "A Rate-Adaptive MAC Protocol for Multihop Wireless Networks," *Proc. 7th Annual Int'l. Conf. Mobile Comp. and Net.*, ACM Press, 2001, pp. 236–51.

[12] B. Sadeghi et al., "Opportunistic Media Access for Multirate Ad Hoc Networks," *Proc. 8th Annual Int'l. Conf. Mobile Comp. and Net.*, ACM Press, 2002, pp. 24–35.

[13] C. E. Perkins and P. Bhagwat, "Highly Dynamic Destination-Sequenced Distance-Vector Routing (DSDV) for Mobile Computers," *Proc. ACM SIGCOMM'94*, London, UK, Sept. 1994.

[14] "OAR Implementation in NS." http://www.ece.rice.edu/networks/software/-OAR/OAR."html

[15] T. A. ElBatt *et al.*, "Power Management for Throughput Enhancement in Wireless Ad-hoc Networks," *IEEE ICC*, 2000, pp. 1506–13.

Biographies

VIKAS KAWADIA (vkawadia@bbn.com) obtained his Ph.D. in the Department of Electrical and Computer Engineering at the University of Illinois at Urbana-Champaign. He received his M.S. in 2001 from the same department, and his B.Tech. in engineering physics from the Indian Institute of Technology, Bombay, in 1999. He is a recipient of the E.A. Reid Award from the University of Illinois. His research has focused on wireless ad hoc networks. He currently works as a network scientist for BBN Technologies.

P. R. KUMAR (prkumar@uiuc.edu) studied at the Indian Institute of Technology, Madras (B.Tech., 1973), and Washington University (D.Sci., 1977). From 1977 to1984 he was with the Department of Mathematics, University of Maryland Baltimore County. Since 1985 he has been with the University of Illinois at Urbana-Champaign, where he is currently Franklin Woeltge Professor of Electrical and Computer Engineering, and a research professor in the Coordinated Science Laboratory. He was the recipient of the Donald P. Eckman Award of the American Automatic Control Council in 1985.