Robustness and the Internet: Theoretical Foundations

John C. Doyle, Jean Carlson, Steven H. Low, Fernando Paganini, Glenn Vinnicombe
Walter Willinger, Jason Hickey, Pablo Parrilo, Lieven Vandenberghe

ROUGH DRAFT

March 5, 2002

Abstract

This article uses the Internet as a starting point to illustrate universal aspects of complex systems throughout technology and biology. Complexity in most systems is driven by the need for robustness to uncertainty in their environments and component parts far more than by basic functionality. Protocols organize highly structured and complex modular hierarchies to achieve robustness, but also create fragilities to rare or neglected perturbations. We claim these are also the most important and universal features of complex systems. All of this complexity is largely hidden, deliberately creating the illusion of superficially simple systems, which encourages development of specious theories. We also discuss an emerging theoretical foundation for the Internet. The aim is to provide a framework for understanding both the successes and shortcomings of existing Internet technology, offer alternative protocols for identified problems, guide rational design for future evolution of ubiquitous networking, and suggest what new mathematics and technology will be needed for a useful, general theory of complex systems.

1 Introduction and motivation

The subject of complex systems is in desperate need of both sound theoretical ideas and concrete, canonical examples, and the Internet is in many ways ideal. The Internet shows unpredictable emergent phenomena and the fragility-complexity-robustness spiral characteristic of evolution in many complex systems. The Internet has elaborate (vertical) multiscale protocols interconnecting transistors, wires, and fiber through to computers and routers on which run a sophisticated collection of software to provide applications for the user and control of the network. Each level itself is complex and (horizontally) distributed. Thus evaluating theoretical claims is challenging, but because we know how all the parts work and are interconnected, and we can make detailed measurements, it is always possible to unambiguously diagnose and "reverse engineer" any claims or phenomena after the fact. This lets us separate sound from specious claims and theories.

We will first compare features of the design and evolution of the Internet with other complex systems familiar from everyday life. This is the first of two themes. If we have identified truly universal and necessary features of complex systems, then they should exhibit themselves everywhere, certainly in the various networks of communication, computing, energy, financial, transportation, etc, that surround us, but even in our laptops, toys, clothes, consumer appliances and electronics, automobiles, and homes. This theme requires no prior technical knowledge, but it would be useful to have a familiarity with the complexity and robustness features of Internet technology as described in [53]. It does require us to look carefully at familiar systems to see past the simple illusions that their complexity creates to their true but sometimes hidden nature. Some of the most cherished and appealing of traditional theoretical concepts from science and engineering are entirely misleading when applied to advanced technological systems. The central issue here is that advanced technologies (and biology) use protocols and feedback to create what amount to deliberate illusions regarding their systems-level behavior. Thus theories that do not fundamentally address protocols and feedback, which is essentially all theory in science and most of engineering, simply do not address the complexity aspects of systems, though they may be essential to understanding how the components function. While having a variety of concrete examples is useful in quickly discarding specious theories, they still leave open the question of

necessity versus contingency. That is, what is essential, and what is an historical artifact. For that we need not just counterexamples, but correct theory.

While control and communications theory has played a crucial role throughout in designing aspects of the Internet, a unified and integrated theory of the Internet as a whole has only recently become a practical and achievable research objective. Dramatic progress has been made recently in analytic results that provide for the first time a foundation for a rigorous, coherent, and complete mathematical theory underpinning Internet technology. This new theory addresses directly the performance and robustness of both the "horizontal" decentralized and asynchronous nature of control in TCP/IP as well as the "vertical" separation into the layers of the TCP/IP protocol stack from application down to the link layer. These results generalize notions of source and channel coding from information theory as well as decentralized versions of robust control. Perhaps most importantly for this collection, the new theoretical insights about the Internet combine with our understanding of its origins and evolution (see [53]) to provide a rich source of ideas about complex systems in general. Most surprisingly, our deepening understanding from genomics and molecular biology has revealed that at the network and protocol level, cells and organisms are strikingly similar to technological networks, despite having completely different material substrates, evolution, and development/construction.

Convergent evolution throughout biology and technology results in systems with universal features: elaborate hierarchies of protocols and modularity and layers of feedback regulatory networks, driven by demand for robustness to uncertain environments and often sloppy components. This complexity makes them robust to the uncertainties for which such complexity was selected, but also makes the resulting system potentially vulnerable to rare or unanticipated perturbations. This complexity is also largely cryptic and hidden in idealized laboratory settings and in normal operation, becoming conspicuous acutely when contributing to rare cascading failures or chronically through fragility/complexity evolutionary spirals. These puzzling and paradoxical features are an ongoing source of confusion to experimentalists, clinicians, and theorists alike, and have led to a rash of specious theories both in biology and more recently about the Internet. However, these "robust yet fragile" features are neither accidental nor artificial and derive from a deep and necessary interplay between complexity and robustness, modularity, feedback,and fragility.

The concept of Highly Optimized Tolerance (HOT) was introduced to focus attention on the robust yet fragile character of complexity. This is the most essential and common property of complex systems in biology, ecology, technology, and socio-economic systems. HOT offers a new and promising framework to study not only network problems, but also put networks in a bigger context. This will be important both with the convergence of existing communication and computing networks and their widely proposed role as a central component of vast enterprise and global networks of networks including transportation, energy, logistics, etc. Research within the HOT framework addresses many complementary aspects of the multifaceted area of networked complex systems. Issues such as robustness, scalability, verifiability and computability can now (and must) be investigated and understood within a common framework. We believe that these different requirements are not only compatible, but can be combined together in a very natural fashion. Our technologies have a priori emphasized these as separate issues, and the promise of a unified approach to simultaneously handle these critical aspects is of paramount importance. While seeking decomposition of hard problems into simpler subproblems will continue to be an important design strategy and is at the heart of protocols' role, an integrated and unified theory is required to do this in a rigorous and robust manner.

Our second theme sketches out the basic ideas of a new theory for the Internet, and its relevance to other complex systems. Although what is most important about this theory is its coherence, it is pedagogically useful to decompose the key ideas in a way that reflects the Internet protocol stack itself. And even though the math details differ substantially, it is also pedagogically helpful to build on and make analogies with standard communication and control theories. Thus for this theme it will be helpful though not essential for the reader to have some familiarity with information theory (source and channel separation, data compression, error correction codes, rate distortion), control theory (feedback, stability, robustness), optimization (duality), dynamical systems (bifurcation, attractors), and computational complexity (P-NP-coNP).

The information theory analogy is that the overall network resource allocation problem, from server-client interactions through congestion control and routing through to network provisioning, can be "optimally" decomposed or separated into the TCP/IP protocol stack. Conventional information theory does not address this problem because of the intrinsic role that dynamics and feedback play in TCP/IP. Fortunately, a source/channel "mice/elephant" picture of web and Internet traffic is emerging that does address these issues. New results not only complete the explanation of existing "self-similar" Internet traffic and its connection with application traffic, but are also the foundation for a complete source/channel coding theory analogous to that from Shannon information theory for conventional communication problems, though differing in detail. Here the issue is the "vertical" separation problem: the optimality

of separating the application level source vs. the TCP/IP "channel" level of the protocol stack, and then the optimal "coding" of the source and channel. The most familiar idea is that the protocol separation is optimal in an asymptotic sense, where the asymptotes here involve large network load and capacity, as opposed to Shannon's large block size. The source coding aspect is the most well-developed, with a variety of results, and models and theories of various level of detail.

The control theory analogy comes from a duality theory of decentralized network resource allocation. This theory integrates and unifies existing approaches to TCP, and also proposes a utility and duality framework for designing new TCP/AQM strategies. This duality theory currently addresses most completely the "horizontal" treatment of the decentralized and asynchronous nature of TCP/AQM, but it provides the framework for the mathematics for the above "vertical" theory of the protocol stack as well. This addresses the source/channel separation between applications and TCP/IP, as well as the further decomposition into TCP and IP routing, and router and link layer network provisioning. A complete separation theory for the entire protocol stack with realistic modeling assumptions will likely require years to develop, but the initial results are very encouraging and the status of the research will be described.

A critical control theoretic development concerns the robustness and stability of the dynamics of TCP/AQM. These results clarify the dynamic (including instability) properties of existing protocols, and have led to new designs which are provably, robustly scalable for arbitrary changes in network size, capacity, and delay. The central technical issue here is the "horizontal" one of global robustness and scalability of the feedback control of congestion and routing that is implemented locally in a fully decentralized and asynchronous manner in TCP/IP. Here it is insufficient to take asymptotic limits, and one needs to prove robustness to arbitrary topologies and delays. While this falls outside the usual domain of control theory, it is again possible to find relevant extensions to robust control methods.

2 Universal features of complexity and robustness

What emerges from the study of the design, evolution, and future challenges of the Internet and related technologies is that the evolution of spiraling complexity, feedback regulation, robustness, fragility, and cascading failures are heavily intertwined, as is well known to biologists and engineers alike. Organization and design of advanced technologies suggest that there are universal principles linking protocols, modularity, and feedback control with the robust yet fragile nature of complex systems. Truly universal principles should manifest themselves in at least limited ways in scale model (toy) systems, as well as other aspects of our everyday lives, and this section will focus on familiar everyday examples in addition to the Internet.

The universal principles that we will focus on begin with the idea that a system's complexity is driven far more by the need for robustness to uncertainty in its environment and component parts than by basic functionality. Thus protocols organize highly structured and complex modular hierarchies to achieve robustness, but also create fragilities to rare or ignored perturbations. The evolution of protocols can lead to a robustness/complexity/fragility spiral where complexity added for robustness also adds new fragilities, which in turn leads to new and thus spiraling complexities. The most powerful and also dangerous protocols involve feedback control, which also has the most mathematical and thus least widely understood theoretical foundations. Finally, all of this complexity is largely hidden and deliberately creates the illusion of superficially simple systems, encouraging development of appealing and accessible but completely wrong explanations and theories.

As we attempt to unravel this hidden complexity and robustness, the basic protocols and modularity are the most easily observable characteristics of complex systems generally, and not just the Internet. While meaning varies, modules generally are components, parts, or subsystems of a larger system that contain some or all of the following features: (a) identifiable interfaces (usually involving protocols) to other modules, (b) can be modified and evolved somewhat independently, (c) facilitate simplified or abstract modeling, (d) maintain some identity when isolated or rearranged, yet (e) derive additional identity from the rest of the system. Internet modules are links, routers, hosts, and the various software components that implement features at various levels and locations throughout the Internet. Perhaps the most familiar software modules are for web browsing and serving and exchange of email.

We claim that protocols are far more important to complexity than are modules. Protocols and modularity are intrinsically complementary and intertwined concepts, but are important to distinguish. In everyday usage, protocols are rules designed to manage relationships and processes smoothly and effectively. If modules are ingredients, parts, components, subsystems, and players then protocols describe the corresponding recipes, architectures, rules, interfaces, etiquettes, and codes of conduct. Protocols here are rules that prescribe allowed interface between modules, permitting system functions that could not be achieved by isolated modules. Protocols also facilitate the addition of

new protocols and organization into collections of mutually supportive protocol suites. A good protocol is one that supplies both robustness and evolvability. The TCP/IP protocol suite is the paradigmatic example.

2.1 LEGO protocols and robustness

Consider the ubiquitous LEGO toy system. The signature feature of LEGO is the patented snap connection for easy but stable assembly of components. The snap is the basic LEGO protocol, and plays the role in LEGO analogous to what IP plays in the Internet. LEGO bricks are the basic modules and correspond to Internet links, hosts,and routers. The snap forms the thin waist of a LEGO hourglass protocol stack where at the bottom a huge variety of alternative brick modules (link and physical layers) can be interconnected via the snap protocol to create at the top a virtually unlimited number of toys (applications). "Everything on IP, IP on everything" becomes in LEGO, "All toys are built using the snap, the snap is implemented in all components." Thus we can use LEGO in two complementary ways. One is to broaden the discussion of the Internet by illustrating the universality of the features of Internet protocols. The other is as a gentle introduction to the concepts of protocols themselves, and thus instead as a starting point for studying Internet protocols.

The obvious parallel between LEGO and TCP/IP is that the most critical elements of the protocol stack, the snap and IP, form the thin waist of an hourglass-like protocol stack. This waist mediates between broad layers of lower level physical modules (bricks, physical and link layer) and equally broad higher level functionality (applications and toys). Note also that in normal operation, the snap and IP are largely hidden. A finished LEGO toy is obviously composed of brick modules but the snaps in between brick are completely hidden from view. Only unused and unnecessary snaps remain visible. Similarly, Internet applications are obviously run on computer hardware, which are in turn obviously connected by various link technologies, but the role of TCP/IP is completely hidden in normal operation. One can imagine a variety of specious theories of LEGO that ignore the role of the snap protocol, just as there are popular and specious theories of the Internet that ignore TCP/IP.

Similar to the Internet, LEGO exhibits multilayer robustness, from components and toys to the product line. LEGO bricks and toys are robust to trauma, reusable, and the snap is versatile, permitting endless varieties of toys from an array of components. The modularity of LEGO is analogous to both the modularity of the link/physical layer technologies and the application technologies as well as the basic use of packets. All confer robustness in the form of flexibility to changing demands and needs. This modularity makes both a given LEGO collection and the entire toy system evolvable to changes in what one chooses to build, to the addition of new LEGO-compatible parts, and to novel toy designs. Evolution here is simply robustness to (possibly large) changes on long time scales. Low cost of modules and the popularity of the system confer other forms of robustness and evolvability, as lost parts are easily replaced, and enthusiasts constantly design new modules and toys.

The LEGO protocol also creates fragilities. Superficially minuscule damage to the snap at a key interface can cause an entire toy to fail. This is very unlikely in normal use, but it is a potential target for malicious attack or a source of rare but catastrophic failure (the consequence of which may be minor, this is after all a toy). In contrast, non-interfacing parts of bricks may be heavily damaged with minimal impact, and even a fatally damaged component is easily replaced once it is identifies. The success of LEGO means that any new, even superior snap would not be easily adopted, and the difficulty in evolving to IPV6 is mirrored in the difficult that would accompany a change in the LEGO snap. Selection pressures thus preserve a protocol in two ways: Protocols facilitate evolution and are difficult to change.

As systems become more complex, protocols facilitate layering of additional protocols, particularly involving feedback and signaling. In analogy with the proposed evolution of networking, as well as biological evolution, suppose we want to make a LEGO structure incrementally more useful and versatile by "evolving" it to be (1) mobile then (2) motorized then (3) able to avoid collisions in a maze of obstacles. The first increment is easy to achieve, with LEGO protocol-compatible axles and wheels. Motorizing toys involves a second increment in complexity requiring protocols for motor and battery interconnection as well as a separate protocol for gears. All can be integrated into a motorized protocol suite, to make modular subassemblies of batteries, motors, gears, axles, and wheels. These are available, inexpensive additions. This is roughly analogous in the Internet to the evolution of new link layer technologies, including faster routers and links, which in turn facilitate new applications that utilize the higher available bandwidth.

The third increment to create a more autonomous, collision avoiding robot, increases cost and complexity by orders of magnitude, requiring layers of protocols and modules for sensing, actuation and feedback controls plus subsidiary but essential ones for communications and computing. All are available, but it is here we begin to see the

true complexity of advanced technologies. Unfortunately, we also start to lose the easily described, intuitive story of the basic protocols. Minimal descriptions of advanced LEGO features enabling sensing and feedback control literally fill books, but the protocols also facilitate building elaborate, robust toys, precisely because this complexity is largely hidden from users. This is consistent with the claim that complexity generally is dominated not by minimal function, but by the protocols and particularly regulatory feedback loops that provide robustness and evolvability. Imagine that a LEGO robot was a prototype for a single toy that dispensed entirely with the LEGO modules in favor of custom implementation. This toy could easily have much more robustness to trauma, be faster, and navigate more complex obstacles, but at the expense of limited part reuse. The modules and lower level protocols would be completely different, yet we might claim that the essence of the toy, and what the prototype aimed to capture, remained. That essence involves the protocols that organized the sensors, actuators, and feedback control system that enables the obstacle avoidance, and contributes almost the entire cost and complexity.

The added complexity of feedback control is absolutely necessary for robust collision avoidance, but also unavoidably creates new and potentially extreme fragilities. While removing a collision-avoiding toy's control system might cause reversion to mere mobility, a small change in an otherwise intact control system could cause wild, catastrophic behavior. For example, a small software bug might easily lead to collision *seeking*, a fragility absent in simpler toys. Note also that a simple test of mobility, without the additional challenge of robust collision avoidance, would not distinguish a highly complex robot from a much simpler one without collision avoidance. Furthermore, disabling collision avoidance would not produce any "phenotype" in a test of mere mobility. It would not imply that collision avoidance components were in any sense "redundant," but merely that they supplied a form of robustness not required by a simple mobility test. These issues arise to some extent in TCP in the feedback control of congestion, but will become much more acute if networks are used in embedded applications for control of physical systems, particular those that involve critical infrastructure.

The snap protocol is only concretely instantiated in LEGO modules, but it is also easy to identify the protocol itself as a useful and informative abstraction. Similarly, TCP/IP may be instantiated only in hosts and routers, but we have no difficulty in thinking of the protocol as an object of study. The snap protocol is thus much more fundamental to LEGO than are any individual modules. Similarly, we have no trouble distinguishing the many higher-level protocols that organize LEGO sensing and feedback from the hardware modules themselves. Good protocols allow new functions to be built from existing components, and new components to be added or to evolve from existing ones, powerfully enhancing both engineering and evolutionary "tinkering." Protocols enable modularity and robustness but are in turn sources of fragility. Successful protocols become highly conserved since they both facilitate evolution and are difficult to change.

The snap protocol itself severely constrains the interconnection of bricks, but the set of useful toys is even more severely constrained and highly structured. Consider the set of all the possible interconnections of a given collection of Lego parts. This is a (combinatorial) huge set. The set of interesting toys is also a large set, but an infinitesimally small subset of what is possible. They are very special and highly structured. Similarly, among the potential toy systems that could conceivably be created using the same basic plastic material, LEGO is highly structured and finely tuned. At the component level, the snap coupling is very precisely machined. Robustness of the type exhibited by LEGO toys and the LEGO system is achieved by fine tuning of highly structured components and interfaces at every level. This becomes most acute in the software used to control a LEGO robot. Thus "robust" and "fine-tuned" are in no sense opposites. Indeed, quite the contrary, highly robust systems are necessarily finely tuned to achieve exactly this robustness.

2.2 Protocols and robustness in computer technology

LEGO Mindstorm owners can build sophisticated robots that use computer vision and other sensors to perform complex feedback control tasks such as avoiding collisions with some objects while seeking out and moving other objects. The brains of a LEGO robot is a microprocessor module that has various protocols that interface with sensors, actuators, and an infrared transmitter that allows communication with remote computers. This communication can be used for a variety of functions, including remote or cooperative sensing and actuating, or for downloading programs using software written on a standard PC. It is possible for a network of LEGO robots distributed around the world to communicate via infrared connections to PCs which in turn can communicate via the Internet.

A huge variety of LEGO, PC, and Internet protocols and modules are used in allowing remote communication between robots. For example, in the PC the central separation is into hardware and software and further into an

hourglass protocol stack with the operating system (OS) at the waist. The OS is intended to allow a huge variety of applications software to interact transparently with vast range of hardware components. In principle, new software and hardware need only be compatible with the OS protocols and they immediately interface with all existing modules for free. Hardware and software are further layered with protocols. For example, in PC hardware the control and interconnection of the CPU, memory, and the huge variety input/output (I/O) devices is handle by a chip set controller. The memory is layered into a speed/size/cost hierarchy of caches and storage technologies. I/O is organized by a variety of protocols interfacing to an almost unlimited collection of components. At the lowest level, digital circuitry is built from an analog substrate. This digital abstraction allows the circuit functionality and interconnection to be separate from the physical analog substrate, facilitating parallel and independent evolution of each. Software is similarly modularized using a variety of programming protocols established for this purpose.

Just listing in the briefest terms all the LEGO, PC, and Internet protocols and modules that would be used in allowing remote communication between robots would fill a book, and the details would fill a library. As the sophistication of users and their robots increases, the emphasis shifts from LEGO as a mechanical toy system to LEGO as a programming and control system. While it is possible to manually program some functionality directly into the LEGO microprocessor, there are a variety of tools that allow the user to develop detailed virtual prototypes on their PCs of both the mechanical and software systems to be built from LEGO components. The total time and cost of creating a real LEGO robot (or almost any engineering system) can be greatly reduced using virtual prototyping, and the engineering of most complex systems, from autos to planes to refineries to space vehicles now relies heavily on virtual prototyping. While the total costs go down, the percentage of costs and complexity that is present in the virtual prototype can be substantial and come to dominate the design of a complex LEGO robot, to the point where actual construction and operation of the physical robot can be a vanishingly small part of the whole effort.

The essential challenge in design a complex robot is robustness, and when using virtual prototyping that the physical implementation behaves like its idealized virtual prototype. This is perhaps the most important lesson to draw from these examples, and its implications can be crystallized by a simple thought experiment. Suppose someone has already built a sophisticated and robust robot that performs some interesting task, and it is our job to reverse engineer this robot. Suppose we can observe the robot's behavior and can study the interconnection of LEGO parts, but do not have access to the software implementing the control system or any hints on the control system design. It will be completely straightforward by inspection to make a copy of the entire robot except for the control software.

To reverse engineer the control software it will be very useful to explore alternatives using a virtual prototype on a PC, and here is where the complexity will be revealed. It will often be quite straightforward to produce a convincing virtual simulation of the robot, yet have the resulting downloaded robot software fail miserably. That is because the complexity is dominated by robustness, and not the obvious basic functionality. It is possible to have a working virtual prototype that fails when implemented because of inadequate robustness, or works intermittently or even most of the time, but has occasional catastrophic failures not present with the original. Control theory is such a complicated and mathematically intensive subject primarily because it aims to address this issue in a systematic way. Its goal is to provide mathematical and software tools that insure that real systems are robust, and work like their virtual prototypes. It has played an important role in Internet technology, and we expect this role to expand.

This example also illustrates the dangers in naive reliance on simulation. It is possible to produce simulations of complex systems that are superficially similar but entirely fail to capture the complexity and robustness of the real system. This is because the complexity is largely hidden and is only *for* robustness. For an Internet example, suppose we are browsing a remote website on our laptop. It is easy to download the entire website from some remote location and then disconnect the laptop from the Internet, and browse the website purely using the local copies. This will be largely indistinguishable from the process of browsing the original website remotely, until something changes or we want to link to some other website (this process is the idea behind web caching). In a sense, we have created an extremely faithful simulation of a complex interaction with the Internet, but now without any of the complex protocols and modules that make up the Internet. While it might be tempting to imagine we had captured some simple essence of the Internet, nothing could be further from the truth. We would have merely captured some trivial and superficial features of one application that runs on the Internet.

2.3 Clothing, money, options, and crashes

The universal features of complexity, protocols, and spiraling robustness and fragility can be seen in every aspect of our modern lives. Our various energy, transportation, consumer, financial and social networks provide astonishing

functionality but are also vulnerable to large cascading failures often initiated by small malfunctions or deliberate disruption by a small number of attackers. We read of precision weapons capable of hitting individual rooms or vehicles thousands of miles from the launch site, but which are equally sensitive to errors in targeting protocols and can thus destroy friendly objects as well. Medical marvels are balanced by the horrors of new viruses and antibiotic resistant bacterial pathogens.

More mundane examples surround us. For example, the process of creating clothing is organized into an hourglass-like protocol stack. The thin waist is supplied by the process of sewing, which can integrate various physical layer cloth technologies at the bottom, into a variety of garments at the top. This can create the illusion of a single, "seamless" garment, hiding the protocols turning raw fibres into thread, threads into cloth and seams, and the resulting sewn elements into garments and ensembles. The seam is typically the source of greatest fragility, and can sometimes be unravelled with a minimal perturbation. This protocol stack facilitates the creation of robust garments, but also introduces entirely new fragilities. In particular, it enables fashion, which can cause a garment to become obsolete long before it has ceased to perform its basic function.

Money is the waist in an hourglass protocol stack connecting, say, widely varied interests of consumers at the top with an equally vast choice of commodities at the bottom. Compared to a straight barter economy, money provides numerous benefits, but also creates fragilities, such as the relative ease of counterfeiting money compared with tradeable goods. This in turn drives the development of highly complex currencies that are hard to counterfeit. Money also facilitates the creation of currency markets and other vast financial markets connecting investors with investment opportunities. This evolution of complexity can exhibit dramatic robustness and fragility spirals. Derivatives such as futures and options allow investors and producers to hedge their positions, for example against currency fluctuations, and more effectively manage risks. They also allow aggressive corporate managers to manipulate their apparent financial status, which can mislead investors. In an attempt to control such practices, accounting rules have undergone an exponential explosion that parallels that in information technology. The evidence suggests, however, that the resulting complexities of this system have led to greater and not less obfuscation, and may have contributed to recent dramatic bankruptcies such as Enron and Global Crossing. This complexity is exacerbated by the increasingly intertwined nature of financial institutions and accounting and management consulting.

Our energy use is organized into an hourglass protocol stack with a huge variety of user devices such as appliances, heating and cooling systems, consumer electronics, office equipment, at the top all using the small waist of a common currency of 110 volt, 60 hertz alternating current and a standard plug format. This can in turn be supplied via a transmission and distribution network, which is governed by its own protocols. Finally, the energy can be generated by a variety of energy sources, provided they follow the appropriate physical layer protocols. Gasoline provides a similar common currency for connecting energy producers with consumers. These efficient energy protocols have facilitated massive deployment of vehicles and other devices in homes and factories, but have created myriad new fragilities. Perhaps most ominous is the impact the pollution from energy sources is having on the global ecosystem.

Many of our systems are undergoing an explosion in complexity due to advanced controls and embedded computing and networking. A Boeing 777 has millions of parts, 150,000 distinct subsystems, including roughly a thousand computers. An example more familiar and accessible than an airplane is the automobile. Modern cars have dozens of microprocessors, automating and controlling every aspect of vehicle function. They host sophisticated engine controls to meet emissions and fuel-economy standards (spark timing, fuel/air ratio, transmission, etc), provide advanced diagnostics for maintenance, enhanced safety features (control of airbags, braking, and traction), and improved comfort and convenience (cruise control, internal environmental controls, GPS-based navigation, security and alarms, wireless emergency communications, etc). This is just the beginning of a trend to higher levels of automation and control. Obviously, much simpler vehicles without any of these features are available, but they pollute more, are less safe, require more frequent and expensive maintenance, and will generally deteriorate more quickly. Complex control and computer networking can actually simplify some features such as wiring by using common components and network busses. Manufacturing and design is also simplified by allowing for highly modular design and the use of sloppier and cheaper mechanical components. This is all facilitated throughout by numerous standard protocols for interfacing modules.

Even the user interface in automobiles (steering wheel, pedals, keys, etc) is a protocol that standardizes most human/vehicle interactions, as are the traffic laws, signs, signals, lanes, and other mechanisms that control traffic. Nevertheless, highway traffic relies heavily on the human driver to perform sensing and control of vehicles. As in LEGO, a full automated highway system would require orders of magnitude more complexity than is in place currently. It is likely a fully automated system would greatly reduce the total number of serious injuries and fatalities

in accidents, but because there would also be new fragilities subject to liability litigation, it is unlikely that such a system will be deployed anytime soon.

A more immediate example of spiraling fragility and complexity is the electronic control system that measures vehicle acceleration during a crash and determines whether to deploy an airbag. A seat belt simply provides specific restraining forces as a function of position, but the airbag is more complex. It is designed to deploy only when the vehicle dynamic state enters a certain regime that is potentially dangerous to the occupant, and there is an intrinsic tradeoff about exactly when and where to deploy the bag. Both too often and too rarely are dangerous, but for opposite reasons. An airbag that deploys during a relatively minor accident can cause trauma where none would have occurred without the airbag. Airbags still give, on average, an overall benefit, but actually make certain circumstances for certain occupants more dangerous, and have led to some unnecessary deaths. Automotive engineers are improving the airbag system with greater complexity to sense occupant size and status and more finely control airbag deployment. Most such robustness-producing mechanisms, such as anti-lock brakes, traction control, or automatic collision avoidance systems, are even more complex.

2.4 Biological complexity

Biological organisms exhibit extraordinarily elaborate hierarchical organization of protocols and modularity, and use feedback control even more ubiquitously and aggressively than does any technology. We are aware of organism's external behaviors, and molecular biology has catalogued many of the elementary components, but the layers of protocols and modularity that connect the two have been far harder to discern. Again, these layers are the most critical features of complex systems but also the most cryptic and hidden, whereas the highest level behavior and the lowest level components are the most visible. Nevertheless, thinking in terms of protocols, in addition to genes, organisms, and populations, as foci of both natural selection and biological research, may be a useful abstraction for understanding the nature and evolution of biological as well as technological complexity. While biological cellular processes are far more distributed, stochastic, and heterogeneous than, say, VLSI circuits, in both cases, such circuits form merely the components of complex control, communications, and computing systems. Biology not only integrates these functions but also builds them directly at the molecular level.

Organisms and ecosystems also have extreme robust yet fragile characteristics. Life in all forms is remarkably robust to environmental fluctuations and tolerates substantial component uncertainty. Ecosystems can recover from massive change, yet be virtually destroyed by a single exotic species. Organisms have been found to grow and persist in almost every environment, and have evolved complexity that will take perhaps centuries to unravel. The control systems that enable such organisms are both critical and largely hidden except when they fail. Even then, large multicellular organisms are unaffected by the death of individual cells, but certain malfunctions in one or a few cell's control systems can lead to fatal autoimmune diseases or cancer.

The future of biology also has many parallels with the future of complex engineering systems. Emphasis is now shifting from components and molecules to the study of the vast networks that biological molecules create that regulate and control life. It is our claim, which this article will not attempt to justify, that while biology and technology have different system-level behavior and even more vastly different component parts, the organizational principles that govern the layers of protocols, modules, and feedback that lie in between are far more alike than is commonly realized. LEGO and the Internet are completely unalike, but they share certain essential features involving robustness, protocols, and feedback control, and these essential features are shared with biology. While there are perhaps many ways in principle to interconnect components to make complex systems, very few that are robust and thus likely to persist and be observed. This is both a recent phenomena and a recent discovery. Only in the last few decades have technological systems (and toys) begun to approach biology in their level of complexity, and thus provide any solid basis for comparison. At the same time, molecular biologists have identified enough of the components and their interconnection that the nature of the intermediate level protocols have begun to emerge.

Biological complexity is a fascinating and timely topic and is subject to vigorous and expanding research attention. Unfortunately, even scratching the surface is well beyond the scope of this article, which will return to a deeper look at the Internet and related technology. Since the research needs for systems engineering and systems biology are converging, it is fortunate that the mathematical theory and software infrastructure to address these needs is finally an achievable goal. Central to this theory is a growing understanding of the "design principles" of complex networks, and the role of protocols, modularity, interconnection, and feedback in creating robust, evolvable systems.

2.5 Evolving internetworking challenges

Many popular technological visions emphasize ubiquitous control, communications, and computing, with systems requiring high levels of not only autonomy and adaptation, but also evolvability, scalability, and verifiability. With current technology these are profoundly incompatible objectives, and both biology and nanotechnology create additional novel multiscale challenges. A rigorous, practical, and unified theoretical framework will be essential for this vision, but until recently, has proven stubbornly elusive. Two of the great abstractions of science and technology have been the separation, in both theory and applications, of 1) controls, communications, and computing from each other, and 2) the systems level from its underlying physical substrate. These separations have facilitated massively parallel, wildly successful, explosive growth in both mathematical theory and technology, but left many fundamental problems unresolved and a poor foundation for future systems of systems in which these elements must be integrated. This horizontal and vertical isolation of systems is at the heart of reductionism. Science has focused almost exclusively on understanding the details of physical substrates, whereas technology has increasingly balanced this with an emphasis on systems, and particularly those aspects largely independent of the physical substrate, such as protocols and software.

Our lives are increasingly dominated by our interaction with a wide variety of networks, in transportation, energy, health, utilities, finance, politics, as well as voice, video, and data, which in turn also interact with our local and global environment. These currently disjoint networks will be increasingly integrated, using ubiquitous embedded computing, into a single convergent network of networks. This creates the opportunity for both unprecedented promise and risk. A lightning strike in one state can cause a power outage in another state far away, a hacker on another continent can deny web access, a single firm can trigger a global financial crisis, a software bug can cause a rocket, airplane, or automobile to crash. Terrorists can turn the power of one network against another, causing tragedies of global proportions. Because the associated networks have remained fairly isolated from each other, such events can have large, but still limited impact. This is changing, and will continue to do so.

While novel human-computer interfaces will transform the way we interact with machines and even with each other via networks, even more applications will also involve devices such as sensors and actuators that interact with the physical environment, with requirements much less forgiving than human users. An extra dimension in this context comes from the problem of designing distributed real-time control to be implemented on networks, adding control *over* networks to the existing substantial challenge of robust control of the network flows themselves. Finally, networks of networks, where communications, computing, and control are deeply embedded, creates new challenges in both cooperative operation, and the containing of catastrophic, cascading failure events. Indeed, perhaps one of our greatest national security threats will be the increasing vulnerability of our critical infrastructure to both cascading failure and deliberate attack.

3 Theoretical background

The success of the Internet has largely been a result of adhering more or less faithfully over time to a set of fundamental network design principles adopted by the the early builders of the Internet (e.g., layering, fate-sharing, end-to-end). In this sense, these "principles" constitute a modest "theory" of the Internet. This theory is shallow in addressing only limited and high-level aspects of the full Internet protocol design problem, but it is also brilliant in the choices that were made. Looking closer at some of the Internet's components, a number of different mathematical tools can be seen to contribute to the components' design or specifications. For example, ideas from both traditional control theory and the theory of queueing networks [29] have contributed to the design and evolution of TCP (e.g., see [25]). Similarly, IP routing employs decentralized protocols (e.g., link-state routing, distance-vector routing, path-vector routing) that have been studied in control theory. For another example, elements of Shannon-type information theory are heavily used for the design of error correcting codes at the link-layer and are widely applied for data compression and encryption throughout the application layer. Moreover, there is ongoing research to use aspects of information theory in limited ways to treat various auxiliary or subsidiary technology challenges created by the Internet (like network data compression and network encryption as opposed to point-to-point data compression or encryption).

Internet researchers have drawn on all aspects of engineering systems theory, including information theory (source and channel separation, data compression, error correction codes, rate distortion), control theory (feedback, stability, robustness), optimization (duality), dynamical systems (bifurcation, attractors), and computational complexity (P-NP-coNP). Each offers some perspective on the Internet, but no one view has so far provided a foundation for the protocol

suite as a whole, and in particular the horizontal, distributed, asynchronous and dynamic nature of control at each layer, nor the vertical interaction of the various layers of the protocol stack. In short, what is lacking is a unified theory of the Internet,and the rest of this section describes various partial attempts towards a theoretical foundation for the Internet. Section 4 will then describe a promising new framework.

3.1 The Internet and communication theory

The most obvious theoretical foundation for the Internet comes from information theory, which was developed to enable robust communication over imperfect channels. Information theory also has some of the most mathematically elegant and accessible ideas in all of engineering systems theory, so it makes an attractive starting point. From an information theory perspective, a unified network theory requires that systems models be generalized from primarily two-node systems, represented by a single-transmitter single-receiver pair, to multi-node networks. We also need to address the fact that, in addition to standard notions of information, data can have both value and connections with other data in time and space through geometry, such as in hyperlinked web layouts, and dynamics, such as in sensor measurements. On the channel side, issues of communications delay must be tackled to allow for the treatment of distributed computation and control problems, both of which involve delay sensitive traffic, as do many other network applications, including voice. Ad hoc and wireless networks have channels that are difficult to model, analyze, and control.

Some progress has been made in all of these areas. For example, the fundamental problem of channel coding for an arbitrary distributed protocol over a multi-processor network illustrates the need for the development of entirely new techniques. An elegant treatment of this problem by Schulman and co-workers [45, 46, 44] extends Shannon's result to general distributed computing, characterizing the achievable rate and reliability of channel codes on arbitrary N node networks. Similarly, efficient information transmission through power-, bandwidth- or delay-limited network technologies, requires maximal data compression before data transmission, yet the theory and practice of source coding for networks remain largely unexplored and entirely unexploited in current network technologies. Recent efforts by Effros et al. begin the task of developing a single unifying theory of network source coding [17, 18], with success in coding for packet-based, point-to-multipoint, and multipoint-to-point networks and recent advances in general network scenarios, promising a unifying theory of network source code performance and design.

These network coding theory extensions are absolutely critical to future network protocols and to a complete theory, but other central aspects that might seem related to source and channel coding problems in networks have received almost no theoretical treatment. For example, if the web sites and clients browsing them are viewed collectively as a single aggregate "source," then this source involves both feedback and geometry as users interactively navigate hyperlinked content. While coding theory is relevant to file compression, the geometric and dynamic feedback aspects are less familiar. Furthermore, the "channel" losses experienced by this source are primarily due to congestion caused by traffic generated by the source itself. This traffic has long range correlations and is self-similar on many time scales [31, 43, 54], which in turn can be traced to fat-tailed file distributions in source traffic being streamed out on the net [55, 12]. These discoveries have inspired recent but extensive research in the modeling of network traffic statistics, their relationship to network protocols and the (often huge) impact on network performance. Despite these efforts, the full implications of this research have yet to be understood or exploited, and only recently has there emerged a coherent coding and control theoretic treatment of the geometric and dynamic aspects of web and other application traffic.

The fat-tailed and self-similar source and channel traffic characteristics initially frustrated mainstream theorists, because they violate standard assumptions in information and queueing theory. Our view is radically different. First, we believe that fat-tailed traffic must be embraced, because it is not an artifice of current applications and protocols, but is a *permanent and essential feature* of network traffic, including all advanced network scenarios. Furthermore, we think that not only can new theory be developed to handle fat-tailed traffic, but if properly exploited, fat-tailed traffic is also ideal for efficient and reliable transport over packet-switched networks. In the Section 4 we will sketch our new treatment of this problem, which builds on theories from robust control [40] and duality in optimization [35], all with a generalized coding perspective from the HOT framework [9, 10, 14, 56]. We show that web and Internet traffic can be viewed as a (perhaps very unfamiliar) joint source and channel coding problem which can be treated systematically as a global optimization problem that is implemented in a decentralized manner. Before we do that, however, we will briefly review yet a third radically different alternative viewpoint on fat tails to either HOT or standard information and queueing theory.

3.2 Chaos, criticality and complexity

Emergence and complexity are both terms that have been widely used to describe the Internet and other systems, but are terms that as yet remain poorly defined. In the Internet context, emergence and complexity could both be used to describe the dramatic growth and evolution of the Internet in both its size and heterogeneity in hardware and applications. While this conforms to our notion of complexity, it would be more precise to replace emergence here with robustness, and particularly robustness of the hourglass TCP/IP protocol structure to just the sort of scaling and evolution for which it was designed. That it was so astonishingly robust certainly qualifies for additional superlatives about the design, but calling it emergent seems to add little. Another class of complex, emergent phenomena in the Internet are the various fragilities that have arisen such as congestion collapse and problems with addressing, BGP, denial of service attacks, worms and virus, and so on. Again, fragility seems more precise here than emergence to describe these unpleasant surprises. Finally, the fat tails in various statistics of traffic and topology could be considered emergent. Interestingly, this initially appeared to be a source of potential fragility, but now appears to be a source of robustness. This will be discussed in more detail Section 4.

Given these caveats, it is certainly fair to say that the Internet is teeming with complexity and emergence, including power law distributions, self-similarity, and fractals. It could also be describe as adaptive, self-organizing, far-from-equilibrium, nonlinear, heterogeneous, and so on. These are all popular notions within the community of researchers loosely organized around such rubrics as Complex Adaptive Systems (CAS), New Science of Complexity (NSOC), Chaoplexity, and more specifically Self-Organized Criticality (SOC) and Edge-of-Chaos (EOC). While there are many differences, from the perspective of this paper they are quite minor, and thus we will lump these various approaches all together under the acronym CCC (for Chaos, Criticality, and Complexity). Recently, a rash of papers have offered explanations for emergent Internet phenomena from the CCC perspective. We will very briefly sketch the aspects of this work most relevant to this article. More detailed reviews of the background, the Internet applications, and the broader comparison with the HOT story have appeared elsewhere (e.g. [53, 14, 9, 8]), and are entirely consistent with the brief sketch here.

The foundations of the CCC approach are that 1) emergent complexity occurs between states of order and disorder characterized by phase transitions and bifurcations in otherwise largely generic interconnections of components, and 2) computer simulations of such generic interconnections with even very simple models of components can reveal the essential nature of this emergent complexity. These claims are not just different but exactly opposite from ours. CCC researchers are inspired by phase transitions and critical phenomena, fractals, self-similarity, pattern formation, and self-organization in statistical physics, and bifurcations and deterministic chaos from dynamical systems. Motivating examples vary from equilibrium statistical mechanics of interacting spins on a lattice to the spontaneous formation of spatial patterns in systems far from equilibrium, such as Raleigh-Benard convection cells and certain driven chemical reactions. Favorite model systems include percolation lattices, cellular automata, random boolean networks, and various networks of interacting agents.

What our perspective shares with CCC is that there are universal and important features of complex systems that transcend the details of specific domains. It is in what those universal features are and what mathematical theory and methods are most relevant that we come to not only different, but essentially exactly opposite, conclusions. CCC has its origins in the physics of simple systems that can produce apparently complex phenomena, coupled with the technology of computer simulation, also of simple systems. We are inspired by the Internet and other engineering networks as well as biological networks, and claim that these are radically different in their nature than random networks of simple components. Our theory's origins are in the mathematics of control, communications, and computing systems. CCC focuses on what happens when a few parameters are adjusted in an otherwise random configuration, whereas we focus on systems in which design or evolution has effectively resulted in the fine tuning of *entire protocols*, which in turn confer the extreme robustness that we observe.

The CCC treatment of the Internet is consistently opposite from ours, not surprisingly. One of the beauties of the Internet is that such different claims can be unambiguously resolved in a way unavailable in most other comparably complex systems. One common starting point is that the IP level traffic typically has roughly power-law correlation functions, and application level traffic produces roughly power-law distributions in their TCP sessions. It is now clear that the latter causes the former, and Section 4 pursues this direction to explore why application traffic has power laws. Completely different explanations come from the CCC perspective, which naturally claim that long range correlations and power laws are signatures of critical phenomena. It is easy to create very simple network traffic models that exhibit phase transitions, since a model of individual router as a simple queue alone trivially has a phase transition at its demand approaches its capacity. An interconnected networks of such simple router models

[38, 47] can be made to self-organize to run at criticality, where it exhibits self-similar characteristics and achieves maximum information transfer and efficiency [47]. Moreover, since some of the key features of this network traffic model are shared by highway traffic models [37], it has been claimed that there exist some deep connections between the dynamics displayed by traffic on highways and computer networks close to criticality.

The essential features of this CCC (i.e. SOC) model of the Internet is that self-similar traffic occurs only at maximum capacity flows, and is independent of application traffic demand and the network protocols. Note that this is exactly the opposite of our view, which is that application traffic demand drives the self-similarity of the network traffic, is independent of flow level, and the full TCP/IP protocols, including congestion and flow control, are essential. Because of the massive data sets available on Internet application and network traffic, we can say definitively that the latter predictions are perfectly consistent with measured data, and the former CCC claims are completely inconsistent. Simulation studies and extensive additional theory now confirms these results, and the theory will be pursued further in Section 4. A similar example in this area concerns the claims made in [50], that the Internet is at the edge-of-chaos (EOC), that (under certain parameter settings) TCP behaves chaotically, and that the self-similar scaling property of Internet traffic can be explained (via TCP) with deterministic chaotic mechanisms. This involves a much more complete model of Internet dynamics, but the data again refutes this unambiguously. Interestingly, as discussed in Section 4, the current standard TCP can be unstable in certain regimes, but this does not cause the self-similar network traffic.

Despite the shortcomings of these CCC-based models, it is important not to ignore the CCC approach, for several reasons. First, it may ultimately have no influence on Internet research or other technologies, or even biology, but it remains the dominant perspective on complex systems within the broader scientific community, which is still struggling to break free of its fiercely reductionist past. Thus it must be taken seriously, if only for purely pedagogical reasons. Pioneering paths of research often turn out to be dead ends, and the Internet is a superb test problem in finding them quickly. Thus we should encourage broader applications to the Internet and accept that most ideas, even the most promising, will be wrong. That's why the subject of complex systems is hard. A pedagogical driver is that the CCC-style models are obviously far more accessible to nonexperts than those that we prefer, and creating a bridge between the two would potentially make the correct models also more accessible. Thus our HOT publications have emphasized using models already popularized within CCC, but to make opposite points. More importantly, if we are claiming that the structure and details not only matter, but are *necessary*, then it is important to contrast this view with its converse.

Thus a second value in the CCC ideas are that they in essence provide *null hypotheses* about the fundamental nature of complexity. The CCC approach is a null hypothesis precisely because it denies the importance of the highly structured protocols and feedback, and ignores robust yet fragile characteristics. In that sense, we believe that the study of the Internet does indeed reveal the essence of the CCC approach to other complex systems, and particularly biological and ecological systems, and in particular how it differs in fundamental ways from the HOT perspective. We claim that the essence of complexity is the role that protocols and feedback play in creating highly structured and robust yet fragile systems, and that their superficially simple behavior is a deliberate illusion. Simulation is an essential tool in understanding such systems, but naively used can reinforce the illusion. Recall that a demo of a web browsing session can be perfectly simulated on a laptop disconnected from the Internet, simply by insuring that the needed web files had been downloaded to the laptop in advance. Only by asking the right *robustness* questions, such as specifically checking if a web page is up-to-date, will we discover the ruse.

It is interesting to note various versions of the Internet could, in principle, be implemented that would behave as the CCC theories suggest using a pure broadcast network without TCP/IP. That is, suppose you connected a group of computers together and any packets to be sent to a specific computer would simply be sent to all computers, and only the one to whom it was addressed would read the packet. Call this protocol SOCnet. This is roughly how a primitive version of Ethernet works on a Local Area Network (LAN), but Ethernet adds retransmission if there are packet collisions, and modern switched Ethernet typically sends packets only to the destination computer. Normally, TCP/IP is run on top of Ethernet, but it need not be. Now imagine running the entire Internet using only SOCnet. It is likely that techniques from statistical physics could be used to predict the traffic patterns in this network. It would indeed exhibit a phase transition at essentially zero packet throughput because there would be a collisions whenever two computers in the whole network transmitted simultaneously. Such a network would have such low throughput that any successful transmissions would be rare, and most computers would never be the destination of any successful transmissions. One reason for IP routing is to eliminate such collisions, and the reason for TCP congestion control is to avoid congesting the routers and losing packets. Eliminating these protocols would simplify analysis but effectively

also eliminate the network. One related area where percolation theory actually does play some small role is in the theory of ad hoc wireless networks, where wireless connectivity of randomly located low power devices is required.

HOT offers a completely different alternative theory to CCC for the nature of complexity. The origin of both power laws and "phase transitions" in complex networks are viewed as just two of the more obvious features of their intrinsic robust yet fragile character: intrinsic, natural, and permanent features of not only Web traffic over TCP/IP but the statistics of complex systems in general, including power grids, ecosystems, and financial markets. Thus beyond web traffic/layout, HOT offers a remarkably rich conceptual framework for thinking about complex networks. HOT also shows how statistical physics, when treated properly, can blend with robust control and information theory to give new perspectives on complex networks, but that the standard tools are inadequate to answer the questions that arise. For example, while the web traffic/layout problem can be described in terms familiar to physicists and information theorist, the obvious standard tools that would appear relevant such as Shannon coding and the renormalization group are of no use in solving the resulting design problem or in explaining observed data.

4 New theory

We now describe our recent progress in developing a coherent coding and control theoretic treatment of the geometric and dynamic aspects of web and other application traffic and their interaction with the TCP/IP protocol stack. While our unified Internet theory is far from complete, and much remains heuristic, we can now reasonably envision a complete theory that will provide for the Internet what information theory and control theory provide for traditional communications and controls. The goal of current research is to extend this preliminary work to systematically treat both the "horizontal" (spatially distributed) and the "vertical" (protocol stack) aspect of the Internet.

4.1 HOT web layout and fat-tailed traffic

The current web/Internet traffic illustrates the limitations of both conventional communications and queueing theory. As discussed earlier, both the "source" and "channel" in a packet switched network have a number of features that have made it unattractive to theorists. The strongly fat-tailed, and nearly self-similar, characteristics of both LAN and WAN traffic is quite unlike the traditionally assumed Poisson traffic models. Real network traffic exhibits long-range dependence and high burstiness over a wide range of time scales. While most files ("mice") have few packets, most packets are in large files ("elephants"). It has further been widely argued that the dominant source of this behavior is due to heavy-tailed Web and other application traffic being streamed out onto the network by TCP to create long-range correlations in packet rates. The applications naturally create bandwidth-hogging elephants and delay-sensitive mice, which coexist rather badly in the current Internet. Our new HOT theory of web layout and traffic and duality theory of flow control suggests that this is not only a permanent and ubiquitous feature of network applications (the bad news?), but also a mix of traffic that can coexist quite efficiently (the good news!), with proper protocol design.

To connect and contrast these ideas with the conventional viewpoint, suppose for concreteness that we are interested in a web site that would be used to browse, search and locate photographs or other images of interest from a large database, such as might arise from satellite surveillance images. We'll discuss other types of media later. Typically websites for such an application will create a variety of pages specifically to help the user navigate the website, and might include thumbnails of low resolution grouped by topics or features, if available.

Conventional rate distortion methods can be used to convert a single high resolution image into a sequence of lower resolution images that provide a tradeoff between compressed file size and distortion, and these can be hyperlinked from low to medium to high resolution, so that a user can progressively obtain higher resolution at the expense of larger file downloads. Suppose, then, that the lower quality reproductions are shrunk down to create smaller reproduction images, with image size a function of reproduction fidelity. So the lowest reproduction accuracy images are represented as "thumbnails" with images of increasing fidelity having increasingly larger reproduction sizes up to the highest reproduction fidelity at the original image size. There are two reasons to do this. One is that a collection of small reproduction size thumbnails can be organized together on a single page and rapidly scanned by users to identify for which images they want higher resolution. This primarily navigational process can typically be done much more efficiently with fewer pages each with many small images than with a sequence of many pages with a few or one high resolution images. Secondly, the low resolution images require smaller compressed file sizes, and can be transmitted using less network resources. These two reasons are not unrelated, as they both involve channel bandwidth, one on

the network to the user's screen, and the other from the screen being scanned during navigation to the user's decision as to which thumbnails to click on.

It is worth reviewing exactly what problem rate distortion theory addresses that is relevant to this problem. The general problem of optimally compressing files without regard to the cost of coding or decoding is well-known to be undecidable, and one of the brilliant insights of Shannon theory is to focus instead on a relaxed version of the problem, that is more computationally tractable. Effectively, in traditional compression theory the emphasis is shifted from the specific file to be compressed, to a *stochastic ensemble* of which the given file is just a typical element. The latter problem turns out to be a lot easier than the former, and it can be argued that it is perhaps a better description. This will be a recurring theme for us: the same mechanism of replacing a given hard problem by a closely related one, but tractable, is at the heart of the convex relaxation procedures discussed later. It is important to remark that in many specific cases, the solutions of the relaxed problems can be shown to be provably close to those of the original one.

Given this relaxation, the rate-distortion problem is then the problem of designing an algorithm or "code" for describing the data in a manner that will achieve the best possible tradeoff between the expected value of the rate (or per-symbol compressed description length) used to describe the data and the expected distortion achieved in the data reconstruction. That is, rate-distortion theory aims to find the shortest data description for a given desired reproduction fidelity or, equivalently, to minimize the expected reproduction fidelity subject to a constraint on the allowed file size. All expectations are taken with respect to the assumed underlying source distribution, and the distortion measure is assumed to be fixed and known at design time.

While this traditional rate-distortion relaxation has led to great advances in both theory and practical code design, it fails to address a number of issues critical to our vision of a unifying theory of communications, controls, and computing over networks. First, even within the traditional bounds of rate-distortion theory there remains an enormous tension between the theoretical optimization of the rate-distortion trade-off and the practically required trade-off between rate, distortion, and complexity. While the field of lossless source coding now contains examples of provably good low-complexity codes, the field of lossy source coding is populated by provably good codes and practical codes between which the relationship is still tenuous.

¿From the perspective of our desired unifying theory, the existing rate-distortion theory is limited not only by its own unanswered questions but also by the questions that it fails to address. For example, traditional source code design has required optimization for a single rate and distortion, with the requirement that a separate and independent code be designed for each rate and reproduction fidelity (or "resolution") of interest. Recent advances by Effros et al. have begun the process of bringing together the theory and practice of multiresolution coding [16]. Multiresolution source codes yield a single embedded description that can be read at a variety of rates and therefore can be used to reproduce the data at a variety of reproduction fidelities. While multiresolution codes allow some of the flexibility required of network environments, where large numbers of users with varying bandwidth/computational capabilities and interest may access the same data file, they fail to address issues such as the geometry of web layout and the topology of related web entries needed for a unifying theory. Finally, rate-distortion theory allows us to minimize distortion relative to a given distortion measure, but says nothing about what particular choice of measure should be made. This selection has clear practical implications in different applications such as distributed computing and control scenarios, or website design.

For the website design case, suppose we assume that the website *topology* is determined by the logical relationship between its component parts. For example, the various descriptions of a single image at different levels of resolution have a topological relation in the obvious way. Images may also have some a priori grouping, perhaps by topics or overall features or origin. What is not given by the content alone is the desired *geometry* of the web layout, that is, essentially the specific locations of cuts in and hyperlinks between the images.

Just as in standard source coding, almost any direct formulation of geometry layout will be intractable, so we will similarly seek an ensemble approach that captures the essential issues. To that end we will assume that what is given is a collection of objects with their sizes and topological connection, and that any rate distortion coding has already been done. (Obviously, a research question of immediate interest is to do joint geometry and rate distortion coding.) We further assume that each object has some probability of access across an ensemble of users. This would naturally arise as users would tend to view, for example, a much larger number of thumbnails than high resolution images, and there might be non-uniformity in the probability of accessing different images at the same resolution.

As a first cut, the assumed performance measure to be minimized through the website design is the average size of a downloaded file. This is motivated by the limitation on the bandwidth of both the network and the user, exactly as in standard source coding. In particular, it is highly desirable for the frequently accessed files that are primarily

navigational to be small and download quickly (mice), as the users next action awaits this information, while the large files (elephants) that are the endpoint of the search process typically need large average bandwidth for timely delivery, but per packet latency will typically be less of an issue. The design degrees of freedom then are the grouping of the objects into files, or conversely the cutting of progressively coded images into files, and thus the sizes of files and the locations of hyperlinks. Finally, this minimization is subject to a constraint not only on the topology, but also on either the total number of files or on the total number, or average depth, or maximum depth of the hyperlinks. For most topologies, these latter constraints will be either exactly or roughly equivalent, and are motivated by the need for the website to be easily navigable by the user, and maintainable by the website's creator. This constraint is quite different from that in data compression, where the constraint on the code is that it be uniquely decodable, leading to Kraft's inequality.

The web layout problem so described has features similar to conventional source coding in aiming to minimize ensemble average bandwidth demand, but substantial differences in the constraints and design degrees of freedom. While these differences mean that the existing theories do not apply, we have already made substantial progress on this problem [14, 56], with two particularly striking results. First, we have been able to find a particular abstraction of the problem that includes both standard data compression and this new web layout problem as special cases. Secondly, the web layout problem produces heavy tailed, and typically power law, distributions of sizes both in files on a website and their access probabilities, consistent with empirical observations. This result is very robust to assumptions, and this framework helps make clear why web file lengths are heavy tailed while codeword lengths are exponentially distributed.

While we have for illustrative purposes described web layout of images, this framework should apply to other media and mixtures of media as well. Users searching for large text documents will typically browse a far larger number of reduced descriptions, such as titles and abstracts, than they will full documents. Video clips will have excerpts and still images, plus text descriptions for use in navigating to the ultimately desired large file downloads, and so on. Indeed, this process of multimedia design has little to do with the web per se, but should arise in almost any organization of information. Thus, for example, it has been widely observed that libraries and file systems also have heavy tailed distributions. Of course, existing website were not designed with this theory in mind, and individual websites are not likely to have optimal layouts. Since the traffic statistics are for aggregate flows, all that is required to explain the striking correspondence between theory and data is that the variations across websites between optimal and actual be uncorrelated. Furthermore, websites that deviate substantially from this prescription would likely be so obviously cumbersome and awkward to navigate that they either would be redesigned, or avoided, reducing their presence in the aggregate statistics.

That the heavy tailed distributions characteristics of web traffic are likely to be an invariant of much of the future network traffic, regardless of the application, is one important insight to be gained from this research direction, even in its currently nascent state. We expect that the current split of most traffic into elephants and mice will persist. Most files will be mice; these files make little aggregate bandwidth demand (due to their small size), but need low latency. Most of the packets come from elephants; these files demand high average bandwidth, but tolerate varying per packet latency. Most human-oriented communication process that involve both active navigation and ultimately the transfer of large objects can naturally be "coded" this way. Even real-time, immersive virtual reality command and control systems and simulators are such that much of their traffic naturally codes into a combination of elephants containing configuration information and models together with mice that update the models in real time. Similarly, sensor and real-time control applications also naturally code into time-critical mice with measurement updates and actuator commands, against a background of elephants which update models of the dynamical environment and network characteristics. Of course, a coherent theory to make rigorous these informal observations is far from available, and the HOT web layout results are merely suggestive and encouraging. Nevertheless, we believe we have identified an important "invariant" of network traffic that must be treated.

While the empirical evidence for this current mix in web and other Internet traffic has received substantial attention recently, not only has no other theoretical work been done to explain it, but in fact the implications for congestion control have been largely ignored, except for the repeated assertions that these distributions break all the standard theories. (A minor exception is the CCC work described above claiming that fat-tails in network traffic are due to critical phase transitions. As we noted, this claim is demonstrably false.) Fortunately, as we will show, this type of traffic creates an excellent blend when the network is properly controlled, and we have already made dramatic progress in exploring the profound implications of fat-tailed traffic for network quality of service (QoS) issues. Thus two critical properties of networks converge in a most serendipitous manner: heavy tails are both a ubiquitous and

permanent feature of network traffic, and an ideal mix of traffic, if properly controlled.

4.2 Control of networks

Successful exploitation of heavy-tailed traffic relies largely on proper congestion control and routing. In this subsection, we describe a duality model of congestion control that horizontally integrates TCP and AQM, illuminates the equilibrium and dynamics of the current protocols, exposes their limitations and motivate new designs. In the next subsection, we explore how this model might be extended to vertically integrate the various layers of the HTTP/TCP/IP protocol stack into a coherent theory.

Even though end-to-end congestion control is targeted towards elephants, it affects strongly the QoS experienced by mice. The goal is to effectively control the elephants to maximally utilize network bandwidth, in a way that leaves the network queues mostly empty. Then the mice that are delay sensitive suffer little queueing delay while elephants that value bandwidth share the network capacity in a way that can be optimally traded off. Provided that mice traffic is relatively small, which is a feature of heavy tailed traffic, they act like noise to elephant traffic. Hence a properly controlled TCP/IP network can provide QoS for free, when QoS means small delay for mice and high average bandwidth for elephants, together with an incentive-compatible pricing policy based on marking. Because this strategy keeps intact the soft state and end-to-end principles of TCP/IP, it is both simple and robust.

For this strategy to work, it is imperative that a rigorous theory be developed both to understand how the current protocols allocate bandwidth and routes, their stability and robustness, and to guide the development of new protocols that are optimized for the mice-elephant setting. As we will illustrate below, without such a theory, we can adopt protocols that have surprising equilibrium and dynamic properties. For example, the current protocol TCP Reno, described in the companion paper [53], becomes unstable as network capacity *increases*, when it operates on a network of RED [19] routers with significant queues, and with DropTail[1] routers, it maximizes backlog in equilibrium, subjecting mice to large delay and loss!

We now briefly review our results in developing such a theory. A network is modeled as a set of L links with finite capacities $c = (c_l, l \in L)$. These links are shared by a set of N sources (elephants). Each source (website) i is routed on (uses) a subset of links from source to destination (user), described by an $L \times N$ routing matrix R, with $R_{li} = 1$ if $l \in L_i$, and 0 otherwise. We will fix the routing matrix R and focus on congestion control of these elephants. Each source i attains a utility $U_i(x_i)$ when it transmits at rate x_i packets per second. We assume U_i are strictly concave and increasing, representing the fact that elephants want as much bandwidth as possible with a diminishing return. We will see that, if bandwidth is allocated in a way that maximizes aggregate source utility, then there is a natural decomposition that allows decentralized allocation algorithms in the form of TCP/AQM protocols.

Specifically, consider the problem:

$$\max_{x_i \geq 0} \quad \sum_i U_i(x_i) \quad \text{subject to} \quad Rx \leq c \tag{1}$$

The constraint says that link flows do not exceed capacities. To obtain a decentralized solution, consider the equivalent dual problem:

$$\min_{p \geq 0} \quad D(p) \;=\; \sum_i \max_{x_i \geq 0} \left(U_i(x_i) \;-\; x_i \sum_l R_{li} p_l \right) \;-\; p^T c \tag{2}$$

The decentralized structure is striking: the problem decomposes into subproblems that can be locally optimized. Indeed, duality theory implies that, given a Lagrange multiplier p^* that solves the dual problem, the optimal rates x_i^* can be computed separately by *individual* sources based on their own utility function and p_l^* on its path:

$$x_i^* \;=\; \arg\max_{x_i \geq 0} \left(U_i(x_i) - x_i \sum_{li} R_{li} p_l^* \right) \;=\; U_i^{'-1} \left(\sum_{li} R_{li} p_l^* \right) \tag{3}$$

Hence the dual variable p is a precise measure of network congestion, and we will call it (bandwidth) 'price'. The remaining issue is how to approach the optimal (p^*, x^*) in a distributed and iterative fashion. We will first describe the

[1] An AQM that simply drops packets that arrive at a full buffer.

decentralized structure any algorithm must satisfy. We will then interpret the current TCP/AQM protocols as different algorithms to solve the utility maximization problem, and review some proposed improved algorithms.

The source rates $x(t)$ determine the aggregate flow $y_l(t)$ at each link, $y_l(t) = \sum_i R_{li} x_i(t - \tau_{li}^f)$, where τ_{li}^f denote the forward transmission delays from sources to links. Each link l maintains a price $p_l(t)$. A source i has access only to the *aggregate* price $q_i(t)$ in its route, $q_i(t) = \sum_l R_{li} p_l(t - \tau_{li}^b)$, where τ_{li}^b denote the backward delays in the feedback path. Decentralization requires that source rates $x_i(t)$ be adjusted based only on aggregate prices $q_i(t)$ and possibly internal state variables z_i:

$$\begin{aligned} \dot{z}_i &= F_i(z_i, q_i) \\ x_i &= G_i(z_i, q_i) \end{aligned} \tag{4}$$

Models in the literature mostly fall into two special cases: the static case, where there is no z_i and we have $x_i(t) = G_i(q_i(t))$, and the first-order case with $z_i = x_i$. Similarly, prices $p_l(t)$ must be adjusted based only on aggregate rates $y_l(t)$ and possibly internal variables v_l:

$$\begin{aligned} \dot{v}_l &= H_l(y_l, v_l) \\ p_l &= K_l(y_l, v_l) \end{aligned} \tag{5}$$

Different TCP protocols can be modeled as different (F_i, G_i) functions and different AQM's as different (G_l, H_l). Prices may represent different metrics in different protocols; e.g., for TCP Reno p_l represents packet loss probability, and for TCP Vegas [7], it represents queueing delay [36]. The key to understanding the equilibrium of a TCP/AQM protocol pair is to regard it as a distributed primal-dual algorithm carried out by sources and links over the Internet in the form of congestion control to solve (1–2). Different protocols all solve the same prototypical problem, but they use different utility functions and implement different iterative rules to optimize them.

Interestingly, even though the equilibrium of (4–5) generally depends on both TCP (F_i, G_i) and AQM algorithms (H_l, G_l), the utility function and the underlying optimization problem, and hence the equilibrium rate allocation and fairness, depends *solely* on the TCP algorithm. As long as the AQM algorithm matches input rate to link capacity at every bottleneck link, the equilibrium of the *pair* will maximize the aggregate utility that is identified with the TCP algorithm. This is because stabilized queues drive the gradient of the dual problem to zero and hence, since the dual problem is convex, they produce prices that are Lagrange multipliers that solve the dual problem.

The utility functions of TCP Reno and Vegas take the form [34, 36]:

$$\begin{aligned} \text{Reno:} \qquad U_i(x_i) &= \frac{\sqrt{2}}{\tau_i} \tan^{-1}\left(\frac{x_i \tau_i}{\sqrt{2}}\right) \\ \text{Vegas:} \qquad U_i(x_i) &= \alpha_i \log x_i \end{aligned}$$

They immediately imply that TCP Reno equalizes source windows in equilibrium, and hence sources with larger delay receive smaller bandwidth, an "unfairness" widely observed empirically. In contrast, TCP Vegas achieves proportional fairness. Since loss probability is the Lagrange multiplier for Reno, it is determined *solely* by the network topology and the number of sources and their utility, *independent of AQM*. More importantly, increasing the buffer size does not change the equilibrium loss probability, and hence, under DropTail, a *larger backlog* must be maintained in order to generate the same loss probability. In other words, increasing buffer size does not reduce loss probability, but only increases average delay, an intriguing phenomenon that becomes apparent in the duality model. This delay and loss behavior is exactly opposite to the mice-elephant control strategy we seek. Under RED [19], the prices and queue lengths steadily increases as the number of sources grow.

As explained above, we can interpret the role of TCP as deciding the equilibrium rate allocation and fairness, in the sense of defining the primal problem (1). Then the role of AQM is to (i) stabilize this equilibrium, (ii) in a way that matches rates to capacity and clears queues, in order to achieve high utilization and low loss and delay. The current AQM protocol RED has failed on both counts (see below for its failing of (i)). This failure has motivated several new AQM algorithms that decouple prices from performance measures such as loss, queue length, or queueing delay, in order to achieve (ii) [2, 40, 30, 24]. More generally, while TCP algorithm can be identified with a utility maximization problem, AQM algorithm can be identified with an optimal control problem where the goal is to generate congestion measures to stabilize a *given* TCP algorithm with minimum cost. We have shown in [28], using a linearized model of TCP Reno, that the solution of the optimal control problem takes the form of AQM, and conversely, any AQM of

the right structure solves the optimal control problem with appropriate parameters. This approach allows a systematic design of AQM that is tailored to a given TCP.

We now remark on the dynamics of the current protocols. It is well known that Reno/RED oscillates wildly and it is extremely hard to reduce the oscillation by tuning RED parameters. The AIMD strategy employed by TCP Reno and mice traffic that is not effectively controlled by TCP no doubt contribute to this oscillation. It is shown in [23, 33] however that, contrary to common belief, these effects pale in comparison with protocol stability. It is whether the network is operating in a stable or an unstable regime that largely determines its dynamic property. Using a linearized model, a stability condition for a single link is obtained that has a surprising implication: TCP/RED becomes unstable when delay increases, or more strikingly, when link capacity increases, when RED maintains a significant queue. This is because doubling link capacity roughly quadruples the control gain, as sources reduce their rates by twice the amount, twice as frequently. This confirms the folklore that TCP performs poorly at large window size.

This result suggests that the current protocols are ill-suited for future networks where both delay and capacity can be large. Indeed, the high control gain introduced by TCP Reno, which induces instability at high delay or capacity, makes compensation by AQM at links extremely difficult. Recently, two new TCP/AQM pairs have been developed, in [41] and [52] (which builds on the results of [27], [26], [51]), using multivariable robust control theory. Each of these schemes maintains linear stability for *arbitrary* delay, capacity, topology and load. Moreover they are both capable of achieving high utilization with low loss and delay in equilibrium. In each case, the key idea is to compensate for delay at sources by scaling down the gain on rates by their individual round trip times, and to compensate for loop gain introduced by capacity and routing by scaling down the control gain at links by their arrival rates/capacities and scaling it up at sources by their current sending rates. Roughly speaking, a source reacts more slowly if its round trip delay is large or if its rate is small; a link updates its price more slowly if it has a larger capacity. Note that network delay is the *only* open-loop dynamics not under our control. It therefore *should*, and will, set the time-scale of the system response and hence scaling down with delay is the best we can do. The individualized scaling here has the appealing feature that sources with low round-trip times can respond quickly, and take advantage of available bandwidth, and it is only those sources whose fast response compromises stability (those with long delays) that must slow down. This strategy leads to the following scalable TCP/AQM schemes. Both schemes have first order dynamics; the difference between them stems from where those dynamics are placed.

The scheme of [52] places the dynamics at the sources and allows arbitrary utility functions, setting

$$p_l \;=\; k_l \left(y_l\right)^B \qquad \text{and} \qquad \dot{x}_i(t) \;=\; \frac{\beta_i}{BT_i} x_r(t-T_i)\left(1-\frac{q_i(t)}{U_i'(x_i(t))}\right)$$

where $\beta_i \in (0, \pi/2)$, τ_i is the round trip time of source i, and B is a globally agreed constant. Utilization can be controlled by the choice of k_l (possibly in an adaptive manner, [30]) and B.

The scheme of [41] places the dynamics at the links and sets

$$\dot{p}_l \;=\; \frac{1}{c_l}(y_l(t)-c_l) \qquad \text{and} \qquad x_i(t) \;=\; x_{\max,i}\, e^{-\frac{\alpha_i q_i(t)}{M_i \tau_i}}$$

where $\alpha_i \in (0,1)$ and M_i is an upper bound on the number of bottleneck links in i's path. This source algorithm implies a utility function

$$U_i(x) = \frac{M_i\tau_i}{\alpha_i} x\left[1-\log\left(\frac{x}{x_{\max,i}}\right)\right], \quad \text{for } x \le x_{\max,i}.$$

An alternative source algorithm that has the same scalable stability but a better fairness property is [39]

$$x_i(t) \;\;=\;\; \left(\frac{\varphi_i}{M_i\tau_i + \beta_i q_i(t)}\right)^{\gamma_i}$$

where $\varphi_i > 0$, $\beta_i > 0$, $\gamma_i > 0$ are parameters such that $\beta_i\gamma_i < \pi/2$. It has a utility function

$$U_i(x) \;\;=\;\; \frac{x}{\beta_i}\left[\frac{\varphi_i\gamma_i}{\gamma_i - 1} x^{-\frac{1}{\gamma_i}} - M_i\tau_i\right]$$

Each of these schemes has its particular merits and demerits of course, and their performance needs to be evaluated further on real, finite networks at the packet level and with binary signalling. The important point here though is that they are all, at least in principle, capable of achieving high utilization with small queues and thereby, with applications generating heavy tailed traffic, also capable of sharing the available bandwidth between elephants whilst allowing the mice to fly through with little delay.

4.3 Integration and cross-layer design

In this subsection, we present tentative ideas on how the HOT theory of web layout/traffic and the duality model of TCP/AQM can be integrated and extended to a coherent theory that treats both the "horizontal" (spatially distributed) and the "vertical" (protocol stack) aspect of the Internet.

Design and implementation of any complex system is inevitably broken down into simpler subsystems that are separately optimized and implemented and then interconnected, often in an ad hoc manner. One of the most visible manifestations is the standardization of protocol stacks for communication networks. IP networks have five protocol layers, application, transport (TCP), network (IP), data link and physical layer. Each layer hides the complexity of the layer below and provides a well-defined service to the layer above (e.g., congestion and error control in TCP, routing in IP). Together, they implement a reliable communication service over a large array of components with different functionality, capability and reliability. Layering manages the tremendous complexity in controlling this set of networked resources, possibly distributed over a wide area, and in coordinating the action of a large pool of competing users. Our goal is to integrate the various protocol layers into a coherent theory, by regarding them as carrying out an asynchronous distributed computation over the Internet to solve an optimization problem. Different layers iterate on different subsets of the decision variables using local information to achieve individual optimality. Taken together, these local algorithms attempt to achieve a global objective. Such a theory will expose the interconnection between protocol layers and can be used to study rigorously the performance tradeoff in protocol layering, as different ways to distribute a centralized computation. Even though the design of a complex system will always be broken down into simpler modules, this theory will allow us to systematically carry out this layering process and explicitly trade off design objectives such as performance, simplicity, and robustness.

To illustrate the potential of this HOT-based framework for vertical as well as horizontal integration, we focus on the TCP- and IP-layers and consider an example that interprets TCP congestion control and IP routing as a (decentralized) algorithm to jointly maximize aggregate utility subject to resource capacity constraints. Using the same notation as above, routing is described by a $L \times N$ matrix R, whose ith column $R_i = (R_{1i}, \ldots, R_{Li})^T$ describes the path of source i. Let $\mathcal{R}$ denote the set of admissible routings R that connect sources (websites) to destinations (users), and $\mathcal{R}_i$ be the set of admissible paths for source i (R_i for $R \in \mathcal{R}$). Using the same objective as in (1), but maximizing over both source rates x_i and admissible routing R, we get:

$$\max_{x_i \geq 0} \max_{R_i} \quad \sum_i U_i(x_i) \quad \text{subject to} \quad Rx \leq c, \quad R \in \mathcal{R}$$

By taking its dual, we get

$$\max_{p \geq 0} \quad D(p) \;=\; \sum_i \max_{x_i \geq 0} \left(U_i(x_i) \;-\; x_i \min_{R_i \in \mathcal{R}_i} \sum_l p_l R_{li} \right) \;-\; p^T c \tag{6}$$

The dual problem has a striking decentralized structure where bandwidth allocation and routing are seperately optimized, over x_i and R_{li} respectively, and coordinated by pricing p. We have described in the last subsection how the dual problem leads to a natural decomposition of the bandwidth allocation problem into TCP/AQM. Here, notice from (6) that routing can be computed by individual sources based only on prices in its paths. Moreover, if dual-optimal prices are used as link distances, then the shortest path algorithm employed by IP is optimal, i.e., solves the dual problem. Thus we can interpret TCP/AQM and IP as asynchronously solving the dual problem along separate coordinates (x, p) and R.[2]

This observation can be used as a starting point for a theory of TCP/AQM/IP, analogous to what has been developed for TCP/AQM (see Section 4.2). It will be interesting to see what such a theory has to say about the sense (if any) in which the current organization of the TCP/IP stack is optimal or whether the evolution of the TCP/IP stack is in danger of getting trapped in some inherently sub-optimal trajectories, and what it has to say about the sense (if any) in which the current approach to routing with its separate inter- and intra-AS routing protocols is optimal or whether the theory will suggest radically different routing protocol designs. Note, for example, that today's inter- and intra-domain routing systems use in effect distributed computation to tackle respectively the inter- and intra-domain domain routing problems and can thus be expected to be in the feasible solution set of an appropriately formulated

[2]We caution however that since neither the feasible set nor the constraint functions are convex when maximization is also taken over R, there may be a duality gap.

HOT-type TCP/AQM/IP problem. This example illustrates the unique potential that the HOT-based framework offers for developing a full-fledged theory of protocol design, based on asynchronous distributed computation, in which integration and cross-layer design issues can be studied. Moreover, HOT offers not only a new and promising framework to study network problems, but it also is capable of putting networks in a bigger context.

5 Additional challenges: Robustness, verification, and computation.

The ultimate challenge in this overall research program is creating a theoretical infrastructure for formal and algorithmic verification of the correctness and robustness of scalable network protocols and embedded software for control of distributed systems. We must ultimately be able to *prove* that systems are robust, and at multiple levels of abstraction, including that of embedded code. This challenge frames all of our research, and the research described in the previous section is one step in addressing this challenge. This section describes our broader and longer-term program in robustness and verification, to provide context for the last section's work, and to point out that the techniques of robust control have recently been extended in ways that offer extremely promising new approaches to our ultimate challenge.

Recent studies indicate that current software practice is nowhere near ready to meet this important challenge. The National Research Council study on information system trustworthiness concluded that the current science and technology base is not adequate for building systems to control critical software infrastructure [15]. The President's commission on critical infrastructure protection and the PITAC report reached the same conclusions [1]: 1) demand for software exceeds our ability to produce it, 2) the nation depends on fragile software, 3) technologies to build reliable software are inadequate.

The two approaches traditionally most concerned with these issues are robust control and formal software verification, which until recently have been both limited in their own domains, and largely independent. New results both radically expand and extend these methods, but also offer hope for their ultimate integration. In both cases, the main conceptual objective is to guarantee that a clearly defined set of "bad behaviors" is empty. For example, in the case of robustness analysis of linear systems, that set can correspond to a particular combination of uncertain parameters producing a large performance index. In protocol verification the bad behavior can be associated, for instance, to a deadlock condition.

Under minimal assumptions, most problems such as those above can be shown to belong to the computational complexity class known as co-NP. The reason is that, provided we can "guess" a solution (a non-deterministic procedure), it can be *verified* in polynomial time that bad behaviors do indeed occur. In other words, if the exact sequence of failures (or bad values of the parameters) is provided, it is straightforward to show that the performance specifications are actually violated. Note, however, that a guarantee that the behavior of the system is the expected one, is equivalent to a *proof* that the set of bad behaviors is empty. The asymmetry between the classes NP and co-NP are as important and profound as those between P and NP (unless they are the same, which is unlikely). Problems in NP always have short proofs, while those in co-NP have short refutations. In contrast, there is no reason in principle to expect co-NP proofs to be short, i.e., polynomial time verifiable. It is a remarkable fact that in many cases, concise arguments about robustness (or correctness) can be provided. The consequences and implications of this are not fully understood, but we have made remarkable recent progress, and our efforts will continue in this direction.

5.1 Software and reliability

The software reliability problem can be made precise starting with the notion of a program containing a set of functions f_i that each compute a result y_i given an argument x_i. A *validation* assigns a specification $X_i \rightarrow Y_i$ to each function f_i, where X_i is the set of possible inputs to f_i and Y_i defines the possible outputs. Properly designed, the program is robust: each of the functions f_i is *guaranteed* to produce a correct result in Y_i given an argument $x_i \in X_i$. It is also fragile: if any of the inputs are invalid $x_i \notin X_i$, the result is *undefined* (and arbitrary). When applied to a control system, the results might be catastrophic, e.g. if a controller $u = f(x)$ produced arbitrary control u given a state x just outside the flight envelope of an aircraft.

These problems have two sources: (1) the functions f_i that define a program are neither continuous nor monotone (an argument x_i just outside the input space X_i will produce arbitrary results), and (2) as a system/network evolves and new features are added, adverse interaction between new and old features may lead to system failure. More precisely, general-purpose programming languages are not *compositional*.

Hickey [21] pointed out that the problem is that programs are defined as *closed* systems. That is, when we develop a program, we *assume* the the program and the environment are known and fixed (the "closed-world assumption"). Programs are defined by *adding* code to implement new features, possibly violating existing assumptions. Hickey developed an *open* model of programming, where it is assumed that the environment is not known, and programs are defined by *subtracting* undesired behavior. Subtractive reasoning is purely compositional: if we have functions f_i that each rule out behaviors other than $F_i = X_i \rightarrow Y_i$, their composition has *at least* the properties of all the parts $\forall_i F_i$. Constable and Hickey [11], used this method to define a *class* theory, based on type theory, that defines an open logical framework for compositional programming using methods from object-oriented programming.

This approach is extremely promising. In earlier work, Hickey and co-workers applied it to the development of multicast group communication protocols [6, 32]. Protocols are constructed by specifying a set of desired properties in the form of a logical algebra [5, 22], then synthesizing and optimizing the resulting protocol. Multicast protocols require much greater functionality than point-to-point protocols—while TCP/IP requires roughly four protocol layers (independent of application), group communication protocols are typically application-specific, and require roughly 30 protocol layers that depend on application requirements. Compositional methods are the only feasible means for protocol synthesis and validation. Furthermore, using online logical methods, these methods have been able to perform optimizations that achieve multicast performance comparable to TCP/IP (roughly a 20x performance improvement over protocols built by hand). This kind of automated protocol construction is a key building-block for networked control systems.

One issue with these methods is that the programs are high-level logical implementations of program properties. While *simple* (automated) compositional reasoning can be used to derive the conjunction of program properties, in general the *actual* program safety properties are stronger, and are defined by a fixed-point construction. In particular, while subtractive composition preserves program *safety*, it does not preserve program *performance*.

These techniques are related to approaches for robustness analysis for control system, where composition corresponds to interconnection of dynamical components. These interconnections restrict the class of possible behaviors by equating inputs and outputs of interconnected blocks. Stability can be characterized in terms of the lack of non-zero solutions to the closed loop equations and robustness is analyzed by allowing structured uncertainty blocks that allow for additional solutions of the system. There are also links with formal methods for program correctness and Lyapunov-based techniques for stability of nonlinear systems. Here, the use of inductive arguments and "invariant assertions" in proving correctness of distributed algorithms can be compared with Lyapunov stability analysis, where one uses a Lyapunov certificate to verify stability of the dynamical system. The results of Parrilo and Doyle described below demonstrate how construction of such Lyapunov functions can be solved using convex relaxations.

These formal techniques for interconnection/composition are an important element in construction of large scale, interconnected systems. They will allow the abstraction of program properties and interconnection protocols to insure that program correctness and performance is maintained. Without such techniques, the concerns of the PITAC cannot be properly addressed.

5.2 Relaxations, robustness and computation

Perhaps the most promising theoretical framework for the unification of robustness and verification methods is the systematic theory of convex relaxations for co-NP problems recently developed in Parrilo's PhD thesis at Caltech. The theory deals with a very general class of *semialgebraic* problems, i.e., those that can be defined with a finite number of polynomial equalities and inequalities. This powerful formulation includes in a unified way instances with both discrete and/or continuous variables, a critical step in connecting robust control with software verification. In this framework, a natural and extremely general question to ask is: given a semialgebraic set S, is it empty? It can be shown that (modulo some technicalities), the question is in co-NP: it is easy to provide (polynomial time) certificates that can prove that S is not empty, since for this it is enough to provide just one point in S. However, if S is actually empty, this may be very hard to prove, as there might not exist an polynomially sized proof of this fact. As a simple example, consider a constrained optimization problem: Let $f(x)$ be a polynomial function, to be minimized over a semialgebraic set S. Then, γ is a lower bound of $\inf_{x \in S} f(x)$ if and only if the semialgebraic set $\{x \in S, f(x) < \gamma\}$ is empty.

The breakthrough in Parrilo's work is that the search for *short proof certificates* can be carried out in an *algorithmic way*. This is achieved by coupling efficient optimization methods and powerful theorems in semialgebraic geometry. A starting point is Stengle's Positivstellensatz, a central theorem in *real* algebraic geometry. It is a common general-

ization of linear programming duality (for linear inequalities) and Hilbert's Nullstellensatz (for an algebraically closed field). It states that, given a system of polynomial equations and inequalities, either there exists a solution in R^n, or there exists a certain polynomial identity which bears *witness* to the fact that no solution exists. This identity takes the form:

$$f(x) + 1 + h(x) = 0, \qquad f \in \mathcal{P}, \quad h \in \mathcal{I}$$

where $\mathcal{P}, \mathcal{I}$ are the *cone* and the *ideal* generated by the given inequalities and equations, respectively. For concreteness the results are stated here for R, instead of the general case of real closed fields. The Positivstellensatz guarantees the existence of a certain kind of proof, but gives no guarantees that a given problem has a *short proof*, i.e., one that can be verified in polynomial time. In fact, not all problems will have short proofs, since otherwise NP=co-NP. However, often we can find short proofs that provide useful information: for instance, in the case of minimization problems, this procedure can provide lower bounds on the value of the optimal solution.

The central piece of the puzzle is the key role played by *sums of squares decompositions*. Sums of squares are an intrinsic element of real algebra, since even the definition of a (formally) real field depends on them. The fact that these decomposition can be computed using *convex optimization* techniques is the critical element needed for the constructive application of results from semialgebraic geometry, and enables an automated search of the sought-for short proofs. For instance, a concise (polynomially-sized) certificate of nonnegativity of a polynomial function (a coNP property) would be given by a explicit sum of squares decomposition. The principal numerical tool used in the search for certificates is *semidefinite programming* [49], a broad generalization of linear and convex quadratic optimization. Semidefinite programs, also known as Linear Matrix Inequalities (LMI) methods, are convex optimization problems, and correspond to the particular case of the convex set being the intersection of an affine family of matrices and the positive semidefinite cone. It is well known that semidefinite programs can be efficiently solved both theoretically and practically.

The main advantage of this new approach is that it transparently provides a *nested hierarchy* of polynomial-time computable relaxations that exhausts co-NP. As mentioned, the existence of certificate polynomials $(f(x), h(x))$ is guaranteed by the Positivstellensatz, so by imposing a degree constraint on them a very appealing hierarchy of proofs is obtained. For each fixed degree bound, the search for proof certificates reduces to a standard semidefinite program.

Many standard results from robustness analysis and combinatorial optimization can be recovered as special cases of this framework. Furthermore, the use of algebraic tools provides a strong connection with established techniques in other domains, such as coding theory. As a concrete example of the power of these results, consider the ubiquitous problem of quadratic inequalities. Given m symmetric matrices $A_1, \ldots, A_m$, define the set $\mathcal{A} := \left\{x \in R^n |\ \mathbf{x}^T A_i \mathbf{x} \geq 0, \quad \|\mathbf{x}\| = 1\right\}$. A well-known sufficient condition for the set $\mathcal{A}$ to be empty is given by the existence of scalars λ_i that satisfy the condition:

$$\sum_{i=1}^{m} \lambda_i A_i \preceq -I, \qquad \lambda_i \geq 0. \tag{7}$$

The condition (7), also known as S-procedure, is the basis of many results in semidefinite relaxations for quadratic programming problems, such as the one underlying the Goemans-Williamson MAXCUT algorithm, and many others. It is well-known that it can be conservative, generally speaking. With our tools, a hierarchy of polynomial-time computable stronger conditions can be obtained. For the first one, assume there exists solutions $Q_i \in R^{n \times n}, r_{ij} \in R$ to:

$$\sum_{i=1}^{n_a} Q_i(x) A_i(x) + \sum_{1 \leq i < j \leq n_a} r_{ij} A_i(x) A_j(x) < 0, \qquad \forall x \in R^n / \{0\}. \tag{8}$$

where $Q_i(x) := x^T Q_i x, Q_i \succeq 0$ and $r_{ij} \geq 0$. Then, the set $\mathcal{A}$ is empty.

Notice that the left-hand size of (8) is a homogeneous form of degree four. Checking the full condition as written would be again a hard problem, so we check instead a sufficient condition: that the left-hand side of (8) can be written (except for the sign change) as a sum of squares. The search for the Q_i, r_{ij} and the verification of the sum of squares condition can be done efficiently and in a simultaneous fashion, using semidefinite programming methods. The new relaxation is *always* at least as powerful as the standard one, and often strictly stronger.

We have verified that the new techniques provide efficient algorithms and improved bounds on the optimal solution of these difficult and well-studied problems. The new bounds are provably never worse than those of standard methods, and in many cases they are strictly better. There are already many extremely well-studied applications where

the developed machinery has demonstrated extraordinary potential, beating the gold standard algorithms, such as optimization of polynomial functions [42], robustness (μ) analysis of uncertain linear systems, Lyapunov analysis for nonlinear systems, and combinatorial optimization (matrix copositivity and 0-1 problems such as MAX CUT). The latter is the well-known NP-hard problem of partitioning a graph or network in two components, in a way that maximizes the sum of the edges cut. All these problems, in their standard formulations, are obviously semialgebraic, and therefore, the application of our methods is in principle completely straightforward. Even more important, having a common mathematical language for all these previously fragmented questions allows a seamless extension of the techniques to mixed problem formulations.

By replacing the computationally intractable positivity condition $V(x) \geq 0$ with relaxed conditions (such as the weaker requirement that $V(x)$ be a sum of squares in $R[x]$), extremely good compromises between efficiency and accuracy are achieved. For instance, the results in [42] show that the *global* minimum of difficult nonconvex optimization problems can be obtained using the new techniques in problem instances with sizes completely outside the range of all other techniques.

For the case of real and complex μ analysis, the first relaxation is the standard upper bound, but an infinite family of higher order relaxations give improved bounds at the expense of increasing, but still polynomial time computation. These bounds are guaranteed to converge to the exact value of μ. Although there is no uniform bound on the degree of the relaxation need for a given accuracy (since otherwise P=NP), numerical experiments have exhibited very encouraging convergence behavior. Finding Lyapunov functions for nonlinear stability proofs involve both the complexity of the Lyapunov function and the complexity of the sum of squares decomposition. Again, under mild conditions, the nested family of relaxations is guaranteed to produce a proof if the system is stable, but with no problem independent uniform bound.

That a single conceptual framework addresses this diverse array of problems is astonishing, but equally exciting is that it does so in a way that should help us treat mixed problems involving, for example, robustness analysis of nonlinear and hybrid systems, and we hope, ultimately including embedded code. At a deeper level, our approach emphasizes the advantages of viewing the formal process of system analysis and verification as one of formal theorem proving. From a practical standpoint, only *short proofs* are truly useful, even though they cannot be taken for granted, and their power should be fully explored and exploited. This has direct parallels with the software reliability ideas in the previous section. The fact that we have now a unified machinery under which to pose our problems, coupled to its extreme versatility and demonstrated efficiency in new and old problem classes, makes us extremely confident that we are in the right direction towards the achievement of our long-term research goals.

In the formulation of relaxations, such as the ones described here, for complex heterogeneous networks a natural difficulty that will need to be addressed is the numerical solution of large-scale convex optimization problems. Previous experience shows in this regard the enormous advantages of using special-purpose code that takes into account the problem structure. In this sense, the numerical expertise and previous related work of Vandenberghe [48, 20] will provide a solid foundation to build on.

Standard relaxation techniques have also recently been applied to the analysis of stochastic networks via convex optimization [3, 13]. The goal here is to describe the region of achievable performance, based on the fact that the performance measures are moments of random variables with an unknown (or incompletely known) distribution. This corresponds to a convex relaxation of the region of achievable performance. Bounds on the performance measures can then be calculated efficiently via linear or semidefinite programming [4].

As mentioned earlier, there is a need for a better understanding, both at the theoretical and algorithmic levels, of the procedures by which we formally guarantee satisfactory performance of a given system. One of our main motivations in this regard is the possible interrelationships between the control-theoretic notion of *robustness* and the theoretical computer science concept of *short proofs*. Effectively, the usual tools of robust control (Lyapunov functions, D-scales, etc.) can be exactly interpreted as *witnesses* or *certificates* that provide a concise proof that some property of the system (for instance, instability) holds. As suggested earlier, there are several hints that point towards a relationship between the minimum length of such a proof, and the robustness of the underlying property. as opposed to relying purely on ad hoc schemes.

5.3 Distributed optimization and control

The design of sensor networks or other types of large-scale wireless networks for communication and control, will require reliable, adaptive, and distributed optimization algorithms (for resource allocation, topology selection, data

processing, ...), and systematic algorithms for analyzing and verifying their performance. The convex optimization techniques (convex duality and relaxations) discussed before in the context of verification, are also essential to the design and analysis of distributed optimization and control algorithms.

Distributed optimization via decomposition

One of the central ideas in large-scale and distributed optimization is decomposition based on duality combined with methods for nondifferentiable optimization, such as cutting-plane algorithms. Decomposition techniques apply to optimization problems in which each node controls one or more decision variables (for example, transmit power in an RF power control problem, FIR filter weights in a beamforming application, ...), and the objective is to minimize a global objective function (e.g., total operating cost, maximum signal to noise plus interference ratio, sidelobe level attenuation, ...), subject to various constraints that couple the decision variables (e.g., limits on shared resources, total power, ...). In a distributed algorithm, the coupling between processors is first translated into constraints on the decision variables, with levels that are iteratively adjusted. Each node optimizes its contribution to the common objective function, subject to the local constraints. From duality, it also calculates the sensitivity of its optimal value with respect to the constraint levels. In other words, for each of the constraints, it calculates by how much the optimal value will improve if the constraint is relaxed, or by how much the optimal value will increase if the constrained is tightened. In the next iteration, this sensitivity (or 'price') information is used to adjust the constraints levels that couple the nodes. A first possibility is that a central 'master' node supervises the algorithm, i.e., collects the results from the nodes (optimal values and sensitivities) and sets constraint levels (e.g., allocates total resources over nodes) for the next iteration. This is the classical framework studied in large-scale optimization. At the other extreme, we find completely decentralized solutions in which processors trade resources with each other. More generally, we can envision solutions where processors organize themselves hierarchically in clusters, with a designated master node supervising each cluster.

Distributed algorithms for the complex networks studied in this article are related to these classical techniques, with a number of fundamental differences. One of the main goals in programming wireless sensor networks is to minimize communication between nodes, as opposed to classical decomposition methods, which were designed to minimize the amount of computation required to solve the subproblems. This need for minimizing communication requires more complicated topologies. More generally, the nodes face a trade-off between energy consumption for local processing and for communication. The amount of communication can be reduced by increasing the complexity and size of the local optimization problem, at the cost of an increase in local processing. As a second important difference, self-organization, essential to the complex networks that we consider, is a new and challenging research problem in distributed optimization.

One of the objectives of this project is to fully explore the range of applications of these ideas. An important example which makes us optimistic about this area is the recent work on TCP flow control, which concerns a duality-based decomposition of a large-scale optimization problem. Other possibilities arise from the sensor networks arena, which is one of the experimental focuses of this project. In particular one can think of power allocation problems, where the network would minimize the required power to accomplish a certain collective task. This is different from classical work in power control in RF systems (which, interestingly, also have connections to optimization). Other related problems would be problems of topology selection, bandwidth allocation, and collaborative signal processing tasks such as localization or tracking of a target source. For a mobile robot platform, optimization objectives could be used as a way to execute a certain maneuvers. Another important objective will be to develop formal methods for evaluating the performance (for example, convergence, robustness) of distributed algorithms.

Distributed optimization and decentralized feedback control

Control and optimization have been strongly related since the days of optimal control, indeed optimization of cost functions over time or frequency provides provides to this day the most powerful methods for multivariable feedback synthesis. Optimal control has, however, been mostly powerless in dealing with decentralized information structures inherent in the distributed systems we are considering.

It is a different blend of optimization and control that can contribute to this challenging area, as exemplified by work in TCP congestion control (see Section 3). Here the dynamics are viewed as iterative algorithms to solve a *static* optimization problem, in a distributed manner over the network. While some features of the problem are reminiscent of optimal control (e.g., the cost function can often be used as a Lyapunov function to prove stability, similarly to the

role of the value function), we are dealing with an entirely different setup here, since there is no optimization over time. Most importantly, ideas from optimization (barrier functions, Lagrange duality, or the methods described above) provide means to introduce feedback with the necessary decentralization. The static cost functions are merely a tool to explore the space of decentralized control laws.

The feedback control aspects of this strategy go, however, beyond the domain of optimization theory. In this regard, typical results on asymptotic convergence are often not too useful since they must assume small enough step-sizes and fixed network parameters over a long enough time. In practical problems, the network is undergoing changes (e.g., new connections in TCP, channel variations in wireless), and one would want to adapt to these as fast as possible consistent with a stable operation. Such performance/robustness tradeoffs in feedback are the domain of control theory, and a closer look reveals the dominant role of delay, a key component of the dynamics in these communication-based systems. We have been successful in resolving these issues for the TCP case, developing algorithms that scale themselves to the existing delay. A main research objective is to expand this viewpoint to other decentralized optimization problems, such as the ones described in the previous section.

5.4 Robustness, verifiability, and scalability

There exists a great deal of evidence, although much of it circumstantial, that suggests that by using short proofs and relaxed problems, the resulting solutions are actually better (less brittle, more scaleable, evolvable, etc) than the original formulation. For instance, there are some results in the mathematical literature, describing an inverse relationship between the "hardness" of a problem (or its proof length) and the distance to the set of ill-posed instances. An example more familiar to the control and robustness analysis field, is the μ upper bound: although originally introduced as a purely computational device for an NP-hard problem, it has been shown later to have very sensible system-theoretic properties. Rather that protecting against *time-invariant* uncertainty (which is what the simplest version of μ does), the upper bound provides a robustness measure for *time-varying* uncertainties of various kinds, depending on the type of bounds used. From an engineering standpoint, this question is as (if not more) relevant than the original one.

The IP layer was designed primarily for robustness and performance was sacrificed. Scalability was a less explicit consideration, but one that turned out to be a very strong property of the design. Our work discussed above on robustness analysis of TCP/AQM/IP suggests that the protocol may be *provably* robust as well, again not a consideration originally. Indeed, the TCP needed to make this proof work is not the one originally designed nor is it one currently deployed. Thus designing for extreme robustness has produced a protocol that was remarkably scalable, and may evolve into one that is verifiable as well.

In general, when we formulate an optimization problem (perhaps intractable), it may be fragile to assumptions in ways we don't realize, lack scalability and evolvability (in ways we don't realize), and so on. In pursuing short proofs and relaxations, we are pursuing both tractability and hopefully providing some degree of protection against fragility to assumptions. One analog to begin understanding the complex relationship among short proofs and sensitivity properties is the simple case of Lagrange multipliers in constrained optimization. In the convex case, the Lagrange multipliers *are* the optimality proof, since they provide a concise optimal bound on the solution. In practice we don't like Lagrange multipliers that are numerically large, since they suggest extreme variability of the optimal cost to the assumptions. Similarly, in a more general nonconvex case we don't want high degree polynomial proofs. In a sense, if linear functionals (Lagrange multipliers) with large coefficients are somehow ill-conditioned, then clearly high degree polynomials can be much worse. Though somewhat rudimentary, this example hopefully illustrates the main points we are after. At this point, we have only anecdotal evidence and a few fragments in our theory, but the picture is definitely encouraging. Both information theory and control theory can be heuristically described this way, and much can be formalized.

We strongly believe that our viewpoint can be fully extended to most verifiability questions that have a natural place in the coNP class. We hope to back up these admittedly speculative observations with stronger theoretical support. On the other hand, *synthesis* (as opposed to *analysis*) seems to be intrinsically much harder, and situated in higher levels of the polynomial hierarchy. This is fairly relevant, since questions such as controller design or code generation have natural synthetic formulations. Therefore, one of our focuses will be on the synthesis implications: how to design systems and protocols that naturally include a (short) proof of correctness, perhaps while sacrificing optimal performance?

Towards this direction, our recently derived TCP modifications provide a perfect example of the practical advan-

tages of including a strong theoretical machinery as an integral part of the design rationale. They also suggest the style of proofs that we think are possible, and are consistent with the existing methods in communications, controls, and computing. We expect to be able to systematically verify the robustness and correctness of protocols, systems, and software with realistic assumptions, and to further prove approximate optimality in extreme, asymptotic scenarios, which may be severe abstraction from reality. Note we are essentially abandoning the hope of optimally robust systems as intrinsically unobtainable for problems of practical interest. Even so, we expect the expansion of the classes of problems for which we can rigorously and algorithmically verify robustness to be slow and difficult, particularly when physical layers impose resource limitations. Nevertheless, we strongly believe our approach is the only viable one for creating a robust and reliable network technologies.

One interesting issue that was not covered in this article is that while the HOT framework was developed primarily for problems involving complex networks, it is also resolving many problems at the foundations of theoretical physics where interconnected, multiscale phenomena is involved. The original publications on HOT involved explaining the ubiquity of power laws in natural and man-made systems. It has recently led to a new view of turbulence in the highly sheared flows that results from design for drag minimization. Other related current research efforts involve the origin of macroscopic dissipation and thermodynamic irreversibility in microscopically reversible dynamics, and aspects of the quantum/classical transition and quantum measurement. The importance of these developments is that a critical issue in future motivating applications is the importance of the physical substrate for the network. Unfortunately, at this point the details are too speculative and the space too limited to expand on the implications of this work. We expect that it may be important to have a theory that can not only unify the horizontal elements of controls, communications, and computing, and the vertical elements of the protocol stack, but also include the physical layers.

References

[1] Information technology research: investing in our future. President's information technology advisory committee report to the president, February 1999.

[2] Sanjeewa Athuraliya, Victor H. Li, Steven H. Low, and Qinghe Yin. REM: active queue management. *IEEE Network*, May/June 2001. Extended version in *Proceedings of ITC17*, Salvador, Brazil, September 2001. http://netlab.caltech.edu.

[3] D. Bertsimas. The achievable region method in the optimal control of queuing systems; formulations, bounds and policies. *Queueing Systems*, 21:337–389, 1995.

[4] D. Bertsimas and J. Sethuraman. Moment problems and semidefinite optimization. In H. Wolkowicz, R. Saigal, and L. Vandenberghe, editors, *Handbook of Semidefinite Programming*. Kluwer Academic Publishers, 2000.

[5] Mark Bickford and Jason Hickey. Predicate transformers for infinite-state automata in nuprl type theory. In Andrew Butterfield and Klemens Haegele, editors, *3rd Irish Workshop on Formal Methods, Electronic Workshops in Computing*. July 1999.

[6] K. Birman, R. Constable, M. Hayden, J. Hickey, C. Kreitz, R. van Renesse, and W. Vo gels O. Rodeh. The horus and ensemble projects: Accomplishments and limitations. In *Proceedings of DISCEX'00*, 2000.

[7] Lawrence S. Brakmo and Larry L. Peterson. TCP Vegas: end to end congestion avoidance on a global Internet. *IEEE Journal on Selected Areas in Communications*, 13(8):1465–80, October 1995. http://cs.princeton.edu/nsg/papers/jsac-vegas.ps.

[8] J. M. Carlson and J.Doyle. Complexity and robustness. In *Proc. Natl. Acad. Sci. USA, 99, Suppl. 1*, pages 2538–2545, 2002.

[9] Jean M. Carlson and John. C. Doyle. Highly Optimized Tolerance: a mechanism for power laws in designed systems. *Physics Review E*, 60:1412–1428, 1999.

[10] Jean M. Carlson and John. C. Doyle. Highly Optimized Tolerance: Robustness and design in complex systems. *Physics Review Letters*, 84(11):2529–2532, 2000.

[11] Robert L. Constable and Jason Hickey. Nuprl's Class Theory and its Applications. In Friedrich L. Bauer and Ralf Steinbrueggen, editors, *Foundations of Secure Computation, NATO ASI Series, Series F: Computer & System Sciences*, pages 91–116. IOS Press, 2000.

[12] M.E. Crovella and A. Bestavros. Self-similarity in World Wide Web traffic: Evidence and possible causes. *IEEE/ACM Transactions on Networking*, 5(6):835–846, 1997.

[13] M. Dacre, K. Glazebrook, and J. Niño-Mora. The achievable region approach to the optimal control of stochastic systems. *Journal of the Royal Statistical Society B*, 61:747–791, 1999.

[14] John C. Doyle and Jean M. Carlson. Power laws, Highly Optimized Tolerance and generalized source coding. *Physics Review Letters*, 84(24):5656–5659, 2000.

[15] Fred Schneider (Ed.). *Trust in Cyberspace*. National Research Council, 1999.

[16] M. Effros. Practical multi-resolution source coding: TSVQ revisited. In *Proceedings of the Data Compression Conference*, pages 53–62, March 1998.

[17] M. Effros. Universal multi-resolution source coding. To appear in *IEEE Transactions on Information Theory*, March 1999.

[18] M. Fleming and M. Effros. The rate-distortion region for the multiple description problem. In *Proceedings of the IEEE International Symposium on Information Theory*, June 2000.

[19] S. Floyd and V. Jacobson. Random early detection gateways for congestion avoidance. *IEEE/ACM Trans. on Networking*, 1(4):397–413, August 1993. ftp://ftp.ee.lbl.gov/papers/early.ps.gz.

[20] A. Hansson and L. Vandenberghe. Efficient solution of linear matrix inequalities for integral quadratic constraints. In *Proceedings IEEE CDC*, 2000.

[21] Jason Hickey. *The MetaPRL logical programming environment*. PhD thesis, Cornell University, Department of Computer Science, 2001. PhD thesis.

[22] Jason Hickey, Nancy Lynch, and Robbert van Renesse. Specifications and proofs for Ensemble layers. In *TACAS '99*, March 1999.

[23] Chris Hollot, Vishal Misra, Don Towsley, and Wei-Bo Gong. A control theoretic analysis of RED. In *Proceedings of IEEE Infocom*, April 2001. http://www-net.cs.umass.edu/papers/papers.html.

[24] Chris Hollot, Vishal Misra, Don Towsley, and Wei-Bo Gong. On designing improved controllers for AQM routers supporting TCP flows. In *Proceedings of IEEE Infocom*, April 2001. http://www-net.cs.umass.edu/papers/papers.html.

[25] V. Jacobson. Congestion avoidance and control. *Proceedings of SIGCOMM'88, ACM,* August 1988. An updated version is available via ftp://ftp.ee.lbl.gov/papers/congavoid.ps.Z.

[26] R. Johari and D Tan. End-to-end congestion control for the internet: Delays and stability. *IEEE/ACM Transactions on Networking*, 9(6):818–832, December 2001.

[27] Frank P. Kelly, Aman Maulloo, and David Tan. Rate control for communication networks: Shadow prices, proportional fairness and stability. *Journal of Operations Research Society*, 49(3):237–252, March 1998.

[28] Ki Baek Kim and Steven H. Low. Analysis and design of AQM based on receding horizon control in stabilizing TCP. submitted for publication, 2002.

[29] L. Kleinrock. *Queueing Systems, Vol. 1 and 2*. Wiley-Interscience, New York, 1976.

[30] Srisankar Kunniyur and R. Srikant. A time–scale decomposition approach to adaptive ECN marking. In *Proceedings of IEEE Infocom*, April 2001. http://comm.csl.uiuc.edu:80/ srikant/pub.html.

[31] W.E. Leland, M.S. Taqqu, W. Willinger, and D.V. Wilson. On the self-similar nature of Ethernet traffic. *IEEE/ACM Transactions on Networking*, (1):1–15, 1994.

[32] Xiaoming Liu, Christoph Kreitz, Robbert van Renesse, Jason Hickey, Mark Hayden, Kenneth Birman, and Robert Constable. Building reliable, high-performance communication systems from components. In *17^{th} ACM Symposium on Operating Systems Principles (SOSP'99)*, pages 80–92, 1999.

[33] S. H. Low, F. Paganini, J. Wang, S. A. Adlakha, and J. C. Doyle. Dynamics of TCP/AQM and a scalable control. In *Proc. of IEEE Infocom*, June 2002. http://netlab.caltech.edu.

[34] Steven H. Low. A duality model of TCP flow controls. In *Proceedings of ITC Specialist Seminar on IP Traffic Measurement, Modeling and Management*, September 18-20 2000. http://netlab.caltech.edu.

[35] Steven H. Low and David E. Lapsley. Optimization flow control, I: basic algorithm and convergence. *IEEE/ACM Transactions on Networking*, 7(6):861–874, December 1999. http://netlab.caltech.edu.

[36] Steven H. Low, Larry Peterson, and Limin Wang. Understanding Vegas: a duality model. *J. of ACM, to appear*, 2002. http://netlab.caltech.edu.

[37] K. Nagel and M. Schreckenberg. A cellular automaton model for freeway traffic. *J. Physique I*, 2:2221–2229, 1992.

[38] T. Ohira and R. Sawatari. Phase transition in computer network traffic model. *Physical Review E*, 58:193–195, 1998.

[39] F. Paganini, S. H. Low, Z. Wang, S. Athuraliya, and J. C. Doyle. A new TCP congestion control with empty queues and scalable stability. submitted for publication, 2002.

[40] Fernando Paganini. On the stability of optimization-based flow control. In *Proceedings of American Control Conference*, 2001. http://www.ee.ucla.edu/ paganini/PS/remproof.ps.

[41] Fernando Paganini, John C. Doyle, and Steven H. Low. Scalable laws for stable network congestion control. In *Proceedings of Conference on Decision and Control*, December 2001. http://www.ee.ucla.edu/ paganini.

[42] P. A. Parrilo and B. Sturmfels. Minimizing polynomials functions. Submitted to the DIMACS volume of the Workshop on Algorithmic and Quantitative Aspects of Real Algebraic Geometry in Mathematics and Computer Science. http://xyz.lanl.gov/abs/math.OC/0103170, 2001.

[43] V. Paxson and S. Floyd. Wide-area traffic: the failure of poisson modeling. *IEEE/ACM Transactions on Networking*, 3(3):226–244, 1995.

[44] S. Rajagopalan and L. J. Schulman. A coding theorem for distributed computation. In *Proceedings of the 26th Annual Symposium on Theory of Computing*, pages 790–799, 1994.

[45] L. J. Schulman. Deterministic coding for interactive communication. In *Proceedings of the 25th Annual Symposium on Theory of Computing*, pages 747–756, 1993.

[46] L. J. Schulman. Coding for interactive communication. *Special Issue on Codes and Complexity of the IEEE Transactions on Information Theory*, 42(6)I, 1996.

[47] R. V. Sole and S. Valverde. Information transfer and phase transition in a model of internet traffic. *Physica A*, 289:595–605, 2001.

[48] L. Vandenberghe and V. Balakrishnan. Algorithms and software for LMI problems in control. *IEEE Control Systems Magazine*, pages 89–95, 1997.

[49] L. Vandenberghe and S. Boyd. Semidefinite programming. *SIAM Review*, 38(1):49–95, 1996.

[50] A. Veres and M. Boda. The chaotic nature of tcp congestion control. In *Proc. IEEE INFOCOM'00*, March 2000.

[51] Glenn Vinnicombe. On the stability of end-to-end congestion control for the Internet. Technical report, Cambridge Univiversity, CUED/F-INFENG/TR.398, December 2000.

[52] Glenn Vinnicombe. On the stability of networks operating TCP-like congestion control. In *Proc. of IFAC*, 2002.

[53] W. Willinger and J. C. Doyle. Robustness and the Internet, I: design and evolution. Preprint, 2002.

[54] W. Willinger and V. Paxson. When mathematics meets the Internet. *Notices of the AMS*, 45(8):961–970, 1998.

[55] W. Willinger, M.S. Taqqu, R. Sherman, and D.V. Wilson. Self-similarity through high variability: statistical analysis of Ethernet LAN traffic at the source level. *IEEE/ACM Transactions on Networking*, 5(1):71–86, 1997.

[56] Xiaoyun Zhu, Jie Yu, and John C. Doyle. Heavy tails, generalized coding, and optimal web layout. In *Proceedings of IEEE Infocom*, April 2001.