INVITED PAPER

Layering as Optimization Decomposition: A Mathematical Theory of Network Architectures

There are various ways that network functionalities can be allocated to different layers and to different network elements, some being more desirable than others. The intellectual goal of the research surveyed by this article is to provide a theoretical foundation for these architectural decisions in networking.

By Mung Chiang, *Member IEEE*, Steven H. Low, *Senior Member IEEE*, A. Robert Calderbank, *Fellow IEEE*, and John C. Doyle

ABSTRACT | Network protocols in layered architectures have historically been obtained on an *ad hoc* basis, and many of the recent cross-layer designs are also conducted through piecemeal approaches. Network protocol stacks may instead be holistically analyzed and systematically designed as distributed solutions to some global optimization problems. This paper presents a survey of the recent efforts towards a systematic understanding of "layering" as "optimization decomposition," where the overall communication network is modeled by a generalized network utility maximization problem, each layer corresponds to a decomposed subproblem, and the interfaces among layers are quantified as functions of the optimization variables coordinating the subproblems. There can be many alternative decompositions, leading to a choice of different layering architectures. This paper surveys the current status of horizontal decomposition into distributed computation, and vertical decomposition into functional modules such as congestion control, routing, scheduling, random access, power control, and channel coding. Key messages and methods arising from many recent works are summarized, and open issues discussed. Through case studies, it is illustrated how "Layering as Optimization Decomposition" provides a common language to think about modularization in the face of complex, networked interactions, a unifying, top-down approach to design protocol stacks, and a mathematical theory of network architectures.

KEYWORDS | *Ad hoc* network; channel coding; computer network; congestion control; cross-layer design; distributed algorithm; feedback control; game theory; Internet; Lagrange duality; medium access control (MAC); network utility maximization (NUM); optimization; power control; reverse-engineering; routing; scheduling; stochastic networks; transmission control protocol (TCP)/Internet protocol (IP); wireless communications

I. INTRODUCTION

A. Overview

1) Structures of the Layered Protocol Stack: Network *architecture* determines *functionality* allocation: "who does what" and "how to connect them," rather than just *resource*

Manuscript received July 24, 2005; revised September 6, 2006. The works at Princeton University and Caltech that are summarized in this paper were supported by the National Science Foundation (NSF) Grants ANI-0230967, EIA-0303620, CNS-0417607, CNS-0435520, CCF-0440443, CCF-0448012, CNS-0417607, CNS-0427677, CNS-0430487, CCF-0635034, and CNS-0519880, by the Air Force Office of Scientific Research (AFOSR) Grants F49620-03-1-0119 and FA9550-06-1-0297, by the ARO Grant DAAD19-02-1-0283, by the Defense Advanced Research Projects Agency (DARPA) Grant HR0011-06-1-0008 and CBMANET program, and by the Cisco Grant GH072605.

M. Chiang is with the Electrical Engineering Department, Princeton University, Princeton, NJ 08544 USA (e-mail: chiangm@princeton.edu).
S. H. Low is with the Computer Science and Electrical Engineering Departments, California Institute of Technology, Pasadena, CA 91125 USA (e-mail: slow@caltech.edu).
A. R. Calderbank is with the Electrical Engineering and Mathematics Departments, Princeton University, Princeton, NJ 08544 USA (e-mail: calderbk@math.princeton.edu).
J. C. Doyle is with the Control and Dynamic Systems, California Institute of Technology, Pasadena, CA 91125 USA (e-mail: doyle@cds.caltech.edu).

Digital Object Identifier: 10.1109/JPROC.2006.887322

allocation. It is often more influential, harder to change, and less understood than any specific resource allocation scheme. Functionality allocations can happen, for example, between the network management system and network elements, between end-users and intermediate routers, and between source control and in-network control such as routing and physical resource sharing. The study of network architectures involves the exploration and comparison of alternatives in functionality allocation. This paper presents a set of conceptual frameworks and mathematical languages for a foundation of network architectures.

Architectures have been quantified in fields such as information theory, control theory, and computation theory. For example, the source-channel separation principle is a fundamental result on architecture in information theory. The choices of architectural decisions are even more complicated in networking. For example, the functionality of rate allocation among competing users may be implemented through various combinations of the following controls: end-to-end congestion control, local scheduling, per-hop adaptive resource allocation, and routing based on end-to-end or per-hop actions. However, we do not yet have a mature theoretical foundation of network architectures.

Layered architectures form one of the most fundamental structures of network design. They adopt a modularized and often distributed approach to network coordination. Each module, called layer, controls a subset of the decision variables, and observes a subset of constant parameters and the variables from other layers. Each layer in the protocol stack hides the complexity of the layer below and provides a service to the layer above. Intuitively, layered architectures enable a scalable, evolvable, and implementable network design, while introducing limitations to efficiency and fairness and potential risks to manageability of the network. There is clearly more than one way to "divide and conquer" the network design problem. From a data-plane performance point of view, some layering schemes may be more efficient or fairer than others. Examining these choices of modularized design of networks, we would like to tackle the question of "how to" and "how not to" layer.

While the general principle of layering is widely recognized as one of the key reasons for the enormous success of data networks, there is little quantitative understanding to guide a systematic, rather than an *ad hoc*, process of designing layered protocol stack for wired and wireless networks. One possible perspective to understand layering is to integrate the various protocol layers into a single theory, by regarding them as carrying out an asynchronous distributed computation over the network to implicitly solve a global optimization problem modeling the network. 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 design process can be quantitatively understood through the mathematical language of *decomposition theory* for constrained optimization [104]. This framework of "Layering as Optimization Decomposition" exposes the interconnections between protocol layers as different ways to modularize and 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 tradeoff design objectives.

The core ideas in "Layering as Optimization Decomposition" are as follows. Different *vertical decompositions* of an optimization problem, in the form of a generalized network utility maximization (NUM), are mapped to different *layering schemes* in a communication network. Each decomposed subproblem in a given decomposition corresponds to a layer, and certain functions of primal or Lagrange dual *variables* (coordinating the subproblems) correspond to the *interfaces* among the layers. *Horizontal decompositions* can be further carried out within one functionality module into *distributed computation and control* over geographically disparate network elements. Since different decompositions lead to alternative layering architectures, we can also tackle the question of "how and how not to layer" by investigating the pros and cons of decomposition methods. Furthermore, by comparing the objective function values under various forms of optimal decompositions and suboptimal decompositions, we can seek "separation theorems" among layers: conditions under which layering incurs no loss of optimality. Robustness of these separation theorems can be further characterized by sensitivity analysis in optimization theory: how much will the differences in the objective value (between different layering schemes) fluctuate as constant parameters in the generalized NUM formulation are perturbed.

There are two intellectually fresh cornerstones behind "Layering as Optimization Decomposition." The first is "network as an optimizer." The idea of viewing protocols as a distributed solution (to some global optimization problem in the form of the basic NUM) has been successfully tested in the trials for transmission control protocol (TCP) [56]. The key innovation from this line of work (e.g., [64], [72], [73], [87], [89], [90], [96], [116], [125], and [161]) is to view the TCP/IP network as an optimization solver, and each variant of congestion control protocol as a distributed algorithm solving a specified basic NUM with a particular utility function. The exact shape of the utility function can be reverse-engineered from the given protocol. In the basic NUM, the objective is to maximize the sum of source utilities as functions of rates, the constraints are linear flow constraints, and optimization variables are source rates. Other recent results also show how to reverse-engineer border gateway protocols (BGPs) as a solution to the stable path problem [44], and contention-based medium access control (MAC) protocols as a game-theoretic selfish utility maximization [76], [78]. Starting from a given protocol originally designed based on engineering heuristics, reverse-engineering discovers the underlying mathematical problems being solved by the protocols. Forward-engineering based on the insights obtained from reverse-engineering then systematically improves the protocols.

The second key concept is "layering as decomposition." As will be discussed in Sections I-A2, generalized NUM problems can be formulated to represent a network design problem involving more degrees of freedom than just the source rates. These generalized NUM problems put the end-user utilities in the "driver's seat" for network design. For example, benefits of innovations in the physical layer, such as better modulation and coding schemes, are now characterized by the enhancement to applications rather than just the drop in bit-error rates (BERs), which the users do not directly observe. Note that an optimal solution to a generalized NUM formulation automatically establishes the benchmark for all layering schemes. The problem itself does not have any predetermined layering architecture. Indeed, layering is a human engineering effort.

The overarching question then becomes how to attain an optimal solution to a generalized NUM in a modularized and distributed way. Vertical decompositions across functional modules and horizontal decompositions across geographically disparate network elements can be conducted systematically through the theory of decomposition for nonlinear optimization. Implicit message passing (where the messages have physical meanings and may need to be measured anyway) or explicit message passing quantifies the information sharing and decision coupling required for a particular decomposition.

There are many ways to decompose a given problem, each of which corresponds to a different layering architecture. Even a different representation of the same NUM problem may lead to different decomposability structures even though the optimal solution remains the same. These decompositions have different characteristics in efficiency, robustness, asymmetry of information and control, and tradeoff between computation and communication. Some are "better" than others depending on the criteria set by network users and operators. A systematic exploration in the space of alternative decompositions is possible, where each particular decomposition leads to a systematically designed protocol stack.

Given the layers, crossing layers is tempting. As evidenced by the large and ever growing number of papers on cross-layer design over the last few years, we expect that there will be no shortage of cross-layer ideas based on piecemeal approaches. The growth of the "knowledge tree" on cross-layer design has been exponential. However, any piecemeal design jointly over multiple layers does not bring a more structured thinking process than the *ad hoc* design of just one layer. What seems to be lacking is a level ground for fair comparison among the variety of cross-layer designs, a unified view on how and how not to layer, and fundamental limits on the impacts of layer-crossing on network performance and robustness metrics.

"Layering as Optimization Decomposition" provides a candidate for such a unified framework. It advocates a first-principled way to design protocol stacks. It attempts at *shrinking* the "knowledge tree" on cross-layer design rather than growing it. It is important to note that "Layering as Optimization Decomposition" is *not* the same as the generic phrase of "cross-layer optimization." What is unique about this framework is that it views the network as the optimizer itself, puts the end-user application needs as the optimization objective, establishes the globally optimal performance benchmark, and offers a common set of methodologies to design modularized and distributed solutions that may attain the benchmark.

There have been many recent research activities along the above lines by research groups around the world. Many of these activities were inspired by the seminal work by Kelly *et al.* in 1998 [64], which initiated a fresh approach of optimization-based modeling and decomposition-based solutions to simplify our understanding of the complex interactions of network congestion control. Since then, this approach has been substantially extended in many ways, and now forms a promising direction towards a mathematical theory of network architectures. This paper[1] provides a summary of the key results, messages, and methodologies in this area over the last 8 years. Most of the surveyed works focus on resource allocation functionalities and performance metrics. The limitations of such focus will also be discussed in Section V.

2) NUM: Before presenting an overview of NUM in this section, we emphasize the primary use of NUM in the framework of "Layering as Optimization Decomposition" as a *modeling tool*, to capture end-user objectives (the objective function), various types of constraints (the constraint set), design freedom (the set of optimization variables), and stochastic dynamics (reflected in the objective function and constraint set). Understanding architectures (through decomposition theory), rather than computing an optimum of a NUM problem, is the main goal of our study.

The *Basic NUM* problem is the following formulation [64], known as Monotropic Programming and studied since the 1960s [117]. TCP variants have recently been reverse-engineered to show that they are implicitly solving this problem, where the source rate vector $\mathbf{x} \geq 0$ is the

[1]Various abridged versions of this survey have been presented in 2006 at the Conference of Information Science and Systems, IEEE Information Theory Workshop, and IEEE MILCOM. Two other shorter, related tutorials can be found in [85] and [105].

only set of optimization variables, and the routing matrix $\mathbf{R}$ and link capacity vector $\mathbf{c}$ are both constants

$$\begin{aligned} &\text{maximize} && \sum_s U_s(x_s) \\ &\text{subject to} && \mathbf{Rx} \leq \mathbf{c}. \end{aligned} \tag{1}$$

Utility functions U_s are often assumed to be smooth, increasing, concave, and dependent on local rate only, although recent investigations have removed some of these assumptions for applications where they are invalid.

Many of the papers on "Layering as Optimization Decomposition" are special cases of the following generic problem [18], one of the possible formulations of a *Generalized NUM* for the entire protocol stack:

$$\begin{aligned} &\text{maximize} && \sum_s U_s(x_s, P_{e,s}) + \sum_j V_j(w_j) \\ &\text{subject to} && \mathbf{Rx} \leq \mathbf{c}(\mathbf{w}, \mathbf{P}_e), \\ &&& \mathbf{x} \in \mathcal{C}_1(\mathbf{P}_e), \quad \mathbf{x} \in \mathcal{C}_2(\mathbf{F}) \;\; \text{or} \;\; \in \mathbf{\Pi}(\mathbf{w}), \\ &&& \mathbf{R} \in \mathcal{R}, \quad \mathbf{F} \in \mathcal{F}, \quad \mathbf{w} \in \mathcal{W}. \end{aligned} \tag{2}$$

Here, x_s denotes the rate for source s and w_j denotes the physical layer resource at network element j. The utility functions U_s and V_j may be any nonlinear, monotonic functions. $\mathbf{R}$ is the routing matrix, and $\mathbf{c}$ are the logical link capacities as functions of both physical layer resources $\mathbf{w}$ and the desired decoding error probabilities $\mathbf{P}_e$. For example, the issue of signal interference and power control can be captured in this functional dependency. The rates may also be constrained by the interplay between channel decoding reliability and other hop-by-hop error control mechanisms like Automatic Repeat Request (ARQ). This constraint set is denoted as $\mathcal{C}_1(\mathbf{P}_e)$. The issue of rate-reliability tradeoff and coding is captured in this constraint. The rates are further constrained by the medium access success probability, represented by the constraint set $\mathcal{C}_2(\mathbf{F})$, where $\mathbf{F}$ is the contention matrix, or, more generally, the schedulability constraint set $\mathbf{\Pi}$. The issue of MAC (either random access or scheduling) is captured in this constraint. The sets of possible physical layer resource allocation schemes, of possible scheduling or contention-based medium access schemes, and of single-path or multipath routing schemes are represented by $\mathcal{W}$, $\mathcal{F}$, and $\mathcal{R}$, respectively. The optimization variables are $\mathbf{x}$, $\mathbf{w}$, $\mathbf{P}_e$, $\mathbf{R}$, and $\mathbf{F}$. Holding some of the variables as constants and specifying some of these functional dependencies and constraint sets will then lead to a special class of this generalized NUM formulation. Utility functions and constraint sets can be even more general than those in problem (2), possibly at the expense of losing specific problem structures that may help with finding distributed solutions.

A deterministic fluid model is used in the above formulations. Stochastic network dynamics change the NUM formulation in terms of both the objective function and the constraint set. As will be discussed in Section V-D, *stochastic NUM* is an active research area.

Whether modeled through a basic, general, or stochastic NUM, there are three separate steps in the design process of "Layering as Optimization Decomposition:" First formulate a specific NUM problem, then devise a modularized and distributed solution following a particular decomposition, and finally explore the space of alternative decompositions that provide a choice of layered protocol stacks.

The following questions naturally arise: How to pick utility functions, and how to guarantee quality-of-service (QoS) to users?

First of all, in reverse-engineering, utility functions are implicitly determined by the given protocols already, and are to be discovered rather than designed. In forward-engineering, utility functions can be picked based on any combination of the following four considerations:

- First, as in the first paper [122] that advocated the use of utility as a metric in networking, elasticity of application traffic can be represented through utility functions.
- Second, utility can be defined by human psychological and behavioral models such as mean opinion score in voice applications.
- Third, utility functions provide a metric to define optimality of resource allocation efficiency.
- Fourth, different shapes of utility functions lead to optimal resource allocations that satisfy well established definitions of fairness (e.g., a maximizer of α-fair utilities parameterized by $\alpha \geq 0$: $U(x) = (1-\alpha)^{-1} x^{1-\alpha}$ [96] can be proved to be an α-fair resource allocation).

In general, depending on who is interested in the outcome of network design, there are two types of objective functions: sum of utility functions by end users, which can be functions of rate, reliability, delay, jitter, power level, etc., and a network-wide cost function by network operators, which can be functions of congestion level, energy efficiency, network lifetime, collective estimation error, etc. Utility functions can be coupled across the users, and may not have an additive structure (e.g., network lifetime).

Maximizing a weighted sum of all utility functions is only one of the possible formulations. An alternative is multiobjective optimization to characterize the Pareto-optimal tradeoff between the user objective and the operator objective. Another set of formulations, which is not covered in this survey, is game-theoretic between users and operators, or among users or operators themselves.

While utility models lead to objective functions, the constraint set of a NUM formulation incorporates the following two types of constraints. First is the collection of

Table 1 Summary of 10 Key Messages

Key Message	*Section*
Existing protocols in layers 2,3,4 have been reverse engineered	Sections II-A.2, II-B.1
Reverse engineering leads to better design	Sections II-A.5, II-B.2
There is one unifying approach to cross-layer design	Section III
Loose coupling through layering price is possible	Section III
Queue length often is a right layering price, but not always	Section III
Convexity is the key to proving global optimality	Section III
Decomposability is the key to designing distributed solution	Section IV
Many alternatives in decompositions and layering architectures	Section IV
There are still many open problems and under-explored directions	Section V
Architecture, rather than optimality, is the emphasis	The whole paper

physical, technological, and economic restrictions in the communication infrastructure. Second is the set of per-user, hard, inelastic QoS constraints that cannot be violated at the equilibrium. This is in contrast to the utility objective functions, which may represent *elastic* QoS demands of the users.

Given a generalized NUM formulation, we do not wish to solve it through centralized computation. Instead, we would like to modularize the solution method through decomposition theory. Each decomposed subproblem controls only a subset of variables (possibly a scalar variable), and observes only a subset of constant parameters and values of other subproblems' variables. These correspond, respectively, to the limited control and observation that each layer has.

The basic idea of decomposition is to divide the original large optimization problem into smaller subproblems, which are then coordinated by a master problem by means of signaling. Most of the existing decomposition techniques can be classified into *primal decomposition* and *dual decomposition* methods. The former is based on decomposing the original primal problem, whereas the latter is based on decomposing the Lagrange dual of the problem. Primal decomposition methods have the interpretation that the master problem directly gives each subproblem an amount of resources that it can use; the role of the master problem is then to properly allocate the existing resources. In dual decomposition methods, the master problem sets the price for the resources to each subproblem which has to decide the amount of resources to be used depending on the price; the role of the master problem is then to obtain the best pricing strategy.

Most papers in the vast, recent literature on NUM use a standard dual-decomposition-based distributed algorithm. Contrary to the apparent impression that such a decomposition is the only possibility, there are in fact many alternatives to solve a given NUM problem in different but all distributed manners [104], including multilevel and partial decompositions. Each of the alternatives provides a possibly different network architecture with different engineering implications.

Coupling for generalized NUM can happen not only in constraints, but also in the objective function, where the utility of source s, $U_s(x_s, \{x_i\}_{i\in I(s)})$, depends on both its local rate x_s and the rates of a set of other sources with indices in set $I(s)$. If U_s is an increasing function of $\{x_i\}_{i\in I(s)}$, this coupling models cooperation, for example, in a clustered system, otherwise it models competition, such as power control in wireless network or spectrum management in digital subscriber loop (DSL). Such coupling in the objective function can be decoupled through "consistency prices" [130].

3) Key Messages and Methodologies: The summary list of key messages in Table 1 illustrates the conceptual simplicity in this rigorous and unifying framework, which is more important than any specific cross-layer design derived from this framework.

In Table 2, the summary list of main methods developed in many recent publications aims at popularizing these analytical techniques so that future research can invoke them readily. Each method will be summarized in a stand-alone paragraph at the *end* of the associated development or explanation.

Sections II and III cover the reverse- and forward-engineering aspects for both horizontal and vertical decompositions, as outlined in Table 3.

After presenting the main points of horizontal and vertical decompositions, we turn to a more general discussion on decomposition methods in Section IV.

At this point, curious readers may start to raise questions, for example, on the issues involving stochastic network dynamics, the difficulties associated with nonconvex optimization formulations, the coverage of accurate models, the comparison metrics for decomposition alternatives, the engineering implications of asymptotic convergence, and the justification of performance optimization in the first place. Some of these questions have recently been answered, while others remain under-explored. Indeed, there are many challenging open problems and interesting new directions in this emerging research area, and they will be outlined in Section V.

In concluding this opening section, we highlight that, more than just an ensemble of specific cross-layer designs for existing protocol stacks, "Layering as Optimization Decomposition" is a *mentality* that views networks as

Table 2 Summary of 20 Main Methods Surveyed

Method	*Section*
Reverse engineering cooperative protocol as an optimization algorithm	Section II-A.2
Lyapunov function construction to show stability	Section II-A.3
Proving convergence of dual-decomposition-based descent algorithm	Section II-A.3
Proving stability by singular perturbation theory	Section II-A.3
Proving stability by passivity argument	Section II-A.3
Proving equilibrium properties through vector field representation	Section II-A.4
Reverse engineering non-cooperative protocol as a game	Section II-B.1
Verifying contraction mapping by bounding the Jacobian's norm	Section II-B.1
Log change of variables for decoupling (and computing minimum curvature needed)	Section II-B.2
Analyzing cross layer interaction through generalized NUM	Section III-A
Dual decomposition for jointly optimal cross layer design	Section III-B
Computing conditions under which a general constraint set is convex	Section III-B
Introducing an extra "layer" to decouple the problem	Section III-B
End user generated pricing	Section III-B
Different timescales of protocol stack interactions through different decompositions	Section III-C
Maximum differential congestion pricing for node-based back-pressure scheduling	Section III-D
Absorbing routing functionality into congestion control and scheduling	Section III-D
Primal and dual decomposition for coupling constraints	Section IV-A
Consistency pricing for decoupling coupled objective	Section IV-B
Partial and hierarchical decomposition for architectural alternatives	Section IV-C

optimizers, a *common language* that allows researchers to quantitatively compare alternative network architectures, and a suite of *methodologies* that facilitates a systematic design approach for modularized and distributed network architectures.

Notation: Unless otherwise specified, vectors are denoted in boldface small letters, e.g., $\mathbf{x}$ with x_i as its ith component; matrices are denoted by boldface capital letters, e.g., $\mathbf{H}$, $\mathbf{W}$, $\mathbf{R}$; and sets of vectors or matrices are denoted by script letters, e.g., $\mathcal{W}_n$, $\mathcal{W}_m$, $\mathcal{R}_n$, $\mathcal{R}_m$. Inequalities between two vectors denote component-wise inequalities. We will use the terms "user," "source," "session," and "connection" interchangeably.

Due to the wide coverage of materials in this survey paper, notational conflicts occasionally arise. Consistency is maintained within any section, and main notation is summarized in the tables of notation for each section: Tables 4–9.

B. From Theory to Practice

1) Optimization: Linear programming has found important applications in communication networks for several decades. In particular, network flow problems, i.e., minimizing linear cost subject to linear flow conservation and capacity constraints, include important special cases such as the shortest path routing and maximum flow problems. Recently, there have been many research activities that utilize the power of recent developments in nonlinear convex optimization to tackle a much wider scope of problems in the analysis and design of communication systems. These research activities are driven by both new demands in the study of communications and networking, and new tools emerging from optimization theory. In particular, a major breakthrough in optimization over the last two decades has been the development of powerful theoretical tools, as well as highly efficient computational algorithms like the interior-point method, for nonlinear convex optimization, i.e., minimizing a convex function (or maximizing a concave function as is often seen in this paper) subject to upper bound inequality constraints on other convex functions and affine equality constraints

$$\begin{aligned} \text{minimize} \quad & f_0(\mathbf{x}) \\ \text{subject to} \quad & f_i(\mathbf{x}) \le 0, \qquad i = 1, 2, \ldots, m \\ & \mathbf{A}\mathbf{x} = \mathbf{a} \end{aligned} \tag{3}$$

where the variables are $\mathbf{x} \in \mathbf{R}^n$. The constant parameters are $\mathbf{A} \in \mathbf{R}^{l \times n}$ and $\mathbf{a} \in \mathbf{R}^l$. The objective function f_0 to be minimized and the m constraint functions f_i are convex functions.

Since the early 1990s, it has been recognized that the watershed between efficiently solvable optimization

Table 3 Organization of Sections II and III

	Reverse Engineering	*Forward Engineering*
Horizontal Decomposition	Sections II-A.2 and II-B.1	Sections II-A.5 and II-B.2
Vertical Decomposition	Section III-A.1	Sections III-A.2, III-B, III-C, III-D

problems and intractable ones is *convexity*. It is well known that for a convex optimization problem, a local minimum is also a global minimum. The Lagrange duality theory is also well-developed for convex optimization. For example, the duality gap is zero under constraint qualification conditions, such as Slater's condition [9] that requires the existence of a strictly feasible solution to nonlinear inequality constraints. When put in an appropriate form with the right data structure, a convex optimization problem can also be efficiently solved numerically, such as the primal-dual interior-point method, which has worst-case polynomial-time complexity for a large class of functions and scales gracefully with problem size in practice.

Special cases of convex optimization include convex quadratic programming, second-order cone programming, and semidefinite programming [9], as well as seemingly nonconvex optimization problems that can be readily transformed into convex problems, such as geometric programming [19]. The last decade has witnessed the appreciation-application cycle for convex optimization, where more applications are developed as more people start to appreciate the capabilities of convex optimization in modeling, analyzing, and designing communication systems. When tackling the much more difficult nonconvex optimization problems, there are some classical approaches, which have been enhanced by new ones in recent years.

The phrase "optimization of communication systems" in fact carries three different meanings. In the most straight-forward way, an analysis or design problem in a communication system may be formulated as minimizing a cost, or maximizing a utility function, or determining feasibility over a set of variables confined within a constraint set. Decomposition, robustness, and fairness, in addition to optimality of the solutions, can then be studied on top of the optimization model. In a more subtle and recent approach, emphasized in Section II, a given network protocol may be interpreted as a distributed algorithm solving an implicitly defined global optimization problem. In yet another approach, the underlying theory of a network control method or a communication strategy may be generalized using nonlinear optimization techniques, thus extending the scope of applicability of the theory.

In addition to optimization theory and distributed algorithm theory, the results surveyed here also naturally borrow tools from feedback control theory, stochastic network theory, game theory, and general market equilibrium theory. They are also connected with other branches of mathematics, such as algebraic geometry and differential topology.

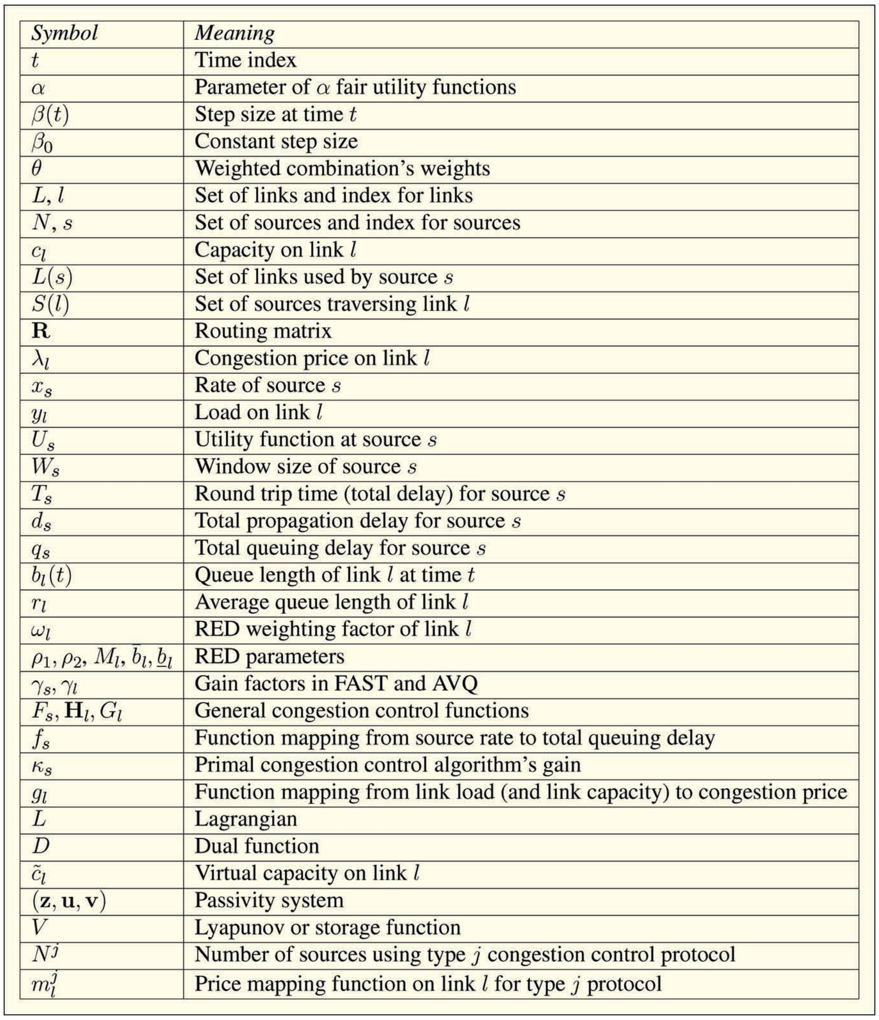
2) Practice: Industry adoption of "Layering as Optimization Decomposition" has already started. For example, insights from reverse-engineering TCP have led to an improved version of TCP in the FAST Project (Fast AQM Scalable TCP) [56], [57], [146], [147]. Putting end-user application utilities as the objective function has led to a new way to leverage innovations in the physical and link layers beyond the standard metrics such as BER, e.g., in the "FAST Copper" Project (here FAST stands for frequency, amplitude, space, time) for an order-of-magnitude boost to rates in fiber/DSL broadband access systems [38].

FAST TCP [37] is a joint project between computer science, control and dynamic systems, electrical engineering, and physics departments at Caltech and UCLA, and involves partners at various national laboratories around the world. It integrates theory, algorithms, implementation, and experiment so that they inform and influence each other intimately. Its goal is to understand the current TCP congestion control, design new algorithms, implement and test them in real high-speed global networks. Through reverse-engineering, as will be discussed in Section II-A2, the NUM and duality model allows us to understand the limitations of the current TCP and design new algorithms. Until about six years ago, the state of the art in TCP research had been simulation-based using simplistic scenarios, with often a single bottleneck link and a single class of algorithms. We now have a theory that can predict the equilibrium behavior of a TCP-like algorithm in any arbitrary network topology. Moreover, we can prove, and design, their stability properties in the presence of feedback delay for large scale networks. As explained in detail in Section II-A5, the insights from this series of theoretical work have been implemented in a software prototype FAST TCP and it has been used to break world records in data transfer in the last few years.

FAST Copper [38] is a joint project at Princeton University, Stanford University, and Fraser Research Institute, aiming at providing at least an order-of-magnitude increase in DSL broadband access speed, through a joint optimization of frequency, amplitude, time, and space dimensions to overcome the attenuation and crosstalk bottlenecks in today's DSL systems. One of the key ideas is to treat the DSL network as a multiple-input–multiple-output (MIMO) system rather than a point-to-point channel, thus leveraging the opportunities of multiuser cooperation and mitigating the current bottleneck due to multiuser competition. Another key idea is to leverage burstiness of traffic over broadband access networks under QoS constraints. The overarching research challenge is to understand how to engineer the functionality allocation across modules and network elements. "Layering as Optimization Decomposition" provides a framework for these design issues in the interference environment of fiber/DSL broadband access.

Clean-slate design of the entire protocol stack is another venue of application of "Layering as Optimization Decomposition." For example, Internet 0 [54] is a project initiated at the Center for Bits and Atoms at MIT and jointly pursued by an industrial consortium. Its goal is to develop theory, algorithms, protocols, and implementations to connect a large number of small devices. Another

Table 4 Summary of Main Notation for Section II-A



Symbol	*Meaning*
t	Time index
α	Parameter of α fair utility functions
$\beta(t)$	Step size at time t
β_0	Constant step size
θ	Weighted combination's weights
L, l	Set of links and index for links
N, s	Set of sources and index for sources
c_l	Capacity on link l
$L(s)$	Set of links used by source s
$S(l)$	Set of sources traversing link l
$\mathbf{R}$	Routing matrix
λ_l	Congestion price on link l
x_s	Rate of source s
y_l	Load on link l
U_s	Utility function at source s
W_s	Window size of source s
T_s	Round trip time (total delay) for source s
d_s	Total propagation delay for source s
q_s	Total queuing delay for source s
$b_l(t)$	Queue length of link l at time t
r_l	Average queue length of link l
ω_l	RED weighting factor of link l
$\rho_1, \rho_2, M_l, \bar{b}_l, \underline{b}_l$	RED parameters
γ_s, γ_l	Gain factors in FAST and AVQ
$F_s, \mathbf{H}_l, G_l$	General congestion control functions
f_s	Function mapping from source rate to total queuing delay
κ_s	Primal congestion control algorithm's gain
g_l	Function mapping from link load (and link capacity) to congestion price
L	Lagrangian
D	Dual function
$\tilde{c}_l$	Virtual capacity on link l
$(\mathbf{z}, \mathbf{u}, \mathbf{v})$	Passivity system
V	Lyapunov or storage function
N^j	Number of sources using type j congestion control protocol
m_l^j	Price mapping function on link l for type j protocol

opportunity of clean-slate protocol stack design for wireless *ad hoc* networks is the control-based MANET program by DARPA. Eventually, "Layering as Optimization Decomposition" may even be used to develop computer-aided design tools for protocol design and implementation.

There are also other potential points of interaction between the theoretical foundation surveyed in this paper and industry practice, e.g., distributed joint rate and power control through decomposition theory by cellular network infrastructure vendors [47], and "visibility" across layers enabled by service providers [70].

II. HORIZONTAL DECOMPOSITION

It is well-known that physical layer algorithms try to solve the data transmission problem formulated by Shannon: maximizing data rate subject to the constraint of asymptotically vanishing error probability. Widely used network protocols, such as TCP, BGP, and IEEE 802.11 DCF, were instead designed based primarily on engineering intuitions and *ad hoc* heuristics. Recent progress has put many protocols in layers 2–4 (of the standard seven-layer reference model) on a mathematical foundation as well.

- The congestion control functionality of TCP has been reverse-engineered to be implicitly solving the basic NUM problem [87], [88], [125]. While *heterogeneous* congestion control protocols do not solve an underlying NUM problem, their equilibrium and dynamic properties can still be analyzed through a vector field representation and the Poincare–Hopf index theorem [134], which together show that bounded heterogeneity implies global uniqueness and local stability of network equilibrium.
- Interior gateway protocol of IP routing is known to solve variants of the shortest path problem, and the policy-based routing protocol in BGP has recently been modeled as the solution to the stable path problem [44].
- Scheduling-based MAC protocols are known to solve variants of maximum weight matching problems [12], [14], [80], [121], [149], [150], [165] or

graph-coloring problems [114] (and the references therein), and random access (contention-based MAC) protocols have recently been reverse-engineered as a noncooperative utility maximization game [76], [78].

In Sections II-A and II-B, the reverse- and forward-engineering results for TCP congestion control and random access MAC are summarized, respectively.

A. TCP Congestion Control

1) Congestion Control Protocols: Congestion control is a distributed mechanism to share link capacities among competing users. In this section, a network is modeled as a set L of links (scarce resources) with finite capacities $\mathbf{c} = (c_l, l \in L)$. They are shared by a set N of sources indexed by s. Each source s uses a set $L(s) \subseteq L$ of links. Let $S(l) = \{s \in N | l \in L(s)\}$ be the set of sources using link l. The sets $\{L(s)\}$ define an $L \times N$ routing matrix[2]

$$R_{ls} = \begin{cases} 1, & \text{if } l \in L(s), \text{ i.e., source } s \text{ uses link } l \\ 0, & \text{otherwise.} \end{cases}$$

Associated with each source s is its transmission rate $x_s(t)$ at time t, in packets/second. Associated with each link l is a scalar congestion measure $\lambda_l(t) \geq 0$ at time t. We will call $\lambda_l(t)$ the link *(congestion) price*.

A congestion control algorithm consists of two components: a source algorithm that dynamically adjusts its rate $x_s(t)$ in response to prices $\lambda_l(t)$ in its path, and a link algorithm that updates, implicitly or explicitly, its price $\lambda_l(t)$ and sends it back, implicitly or explicitly, to sources that use link l. On the current Internet, the source algorithm is carried out by TCP, and the link algorithm is carried out by (active) queue management (AQM) schemes such as DropTail or RED [43]. Different protocols use different metrics to measure congestion, e.g., TCP Reno [55], [127] and its variants use loss probability as the congestion measure, and TCP Vegas [10] and FAST [56], [147] use queueing delay as the congestion measure [89]. Both are implicitly updated at the links and implicitly fed back to sources through end-to-end loss or delay, respectively. Mathematical models for congestion control started [25] immediately after the development of the protocol, and many of the results since 1999 follow the approach advocated in [64] and focus on average models of the congestion avoidance phase in TCP.

In this section, we show that a large class of congestion control algorithms can be interpreted as distributed algorithms to solve a global optimization problem. Specifically, we associate with each source s a utility function $U_s(x_s)$ as a function of its rate x_s. Consider the basic NUM proposed in [64]

$$\begin{aligned} \text{maximize} \quad & \sum_s U_s(x_s) \\ \text{subject to} \quad & \mathbf{Rx} \leq \mathbf{c} \end{aligned} \tag{4}$$

and its Lagrangian dual problem [90]

$$\text{minimize}_{\boldsymbol{\lambda} \geq 0} \quad D(\boldsymbol{\lambda}) := \sum_s \max_{x_s \geq 0} \left(U_s(x_s) - x_s \sum_l R_{ls} \lambda_l \right) + \sum_l c_l \lambda_l. \tag{5}$$

We now present a general model of congestion control algorithms and show that they can be interpreted as distributed algorithms to solve NUM (4) and its dual (5).

Let $y_l(t) = \sum_s R_{ls} x_s(t)$ be the aggregate source rate at link l and let $q_s(t) = \sum_l R_{ls} \lambda_l(t)$ be the end-to-end price for source s. In vector notation, we have

$$\mathbf{y}(t) = \mathbf{Rx}(t)$$

and

$$\mathbf{q}(t) = \mathbf{R}^T \boldsymbol{\lambda}(t).$$

Here, $\mathbf{x}(t) = (x_s(t), s \in N)$ and $\mathbf{q}(t) = (q_s(t), s \in N)$ are in $\mathbf{R}_+^N$, and $\mathbf{y}(t) = (y_l(t), l \in L)$ and $\boldsymbol{\lambda}(t) = (\lambda_l(t), l \in L)$ are in $\mathbf{R}_+^L$.

In each period, the source rates $x_s(t)$ and link prices $\lambda_l(t)$ are updated based on local information. Source s can observe its own rate $x_s(t)$ and the end-to-end price $q_s(t)$ of its path, but not the vector $\boldsymbol{\lambda}(t)$, nor other components of $\mathbf{x}(t)$ or $\mathbf{q}(t)$. Similarly, link l can observe just local price $\lambda_l(t)$ and flow rate $y_l(t)$. The source rates $x_s(t)$ are updated according to

$$x_s(t+1) = F_s(x_s(t), q_s(t)) \tag{6}$$

for some nonnegative functions F_s. The link congestion measure $\lambda_l(t)$ is adjusted in each period based only on $\lambda_l(t)$ and $y_l(t)$, and possibly some internal (vector) variable $\mathbf{v}_l(t)$, such as the queue length at link l. This can be modeled by some functions $(G_l, \mathbf{H}_l)$: for all l

$$\lambda_l(t+1) = G_l(y_l(t), \lambda_l(t), \mathbf{v}_l(t)) \tag{7}$$

$$\mathbf{v}_l(t+1) = \mathbf{H}_l(y_l(t), \lambda_l(t), \mathbf{v}_l(t)) \tag{8}$$

[2]We abuse notation to use L and N to denote both sets and their cardinalities.

where G_l are nonnegative so that $\lambda_l(t) \geq 0$. Here, F_s model TCP algorithms (e.g., Reno or Vegas) and $(G_l, \mathbf{H}_l)$ model AQMs (e.g., RED, REM). We will often refer to AQMs by G_l, without explicit reference to the internal variable $\mathbf{v}_l(t)$ or its adaptation $\mathbf{H}_l$. We now present some examples.

TCP Reno/RED: The congestion control algorithm in the large majority of current TCP implementations is (an enhanced version of) TCP Reno, first proposed in [55]. A source maintains a parameter called *window size* that determines the number of packets it can transmit in a round-trip time (RTT), the time from sending a packet to receiving its acknowledgment from the destination. This implies that the source rate is approximately equal to the ratio of window size to RTT, in packets per second. The basic idea of (the congestion avoidance phase of) TCP Reno is for a source to increase its window by one packet in each RTT and halve its window when there is a packet loss. This can be modeled by (see, e.g., [72], [87]) the source algorithm $F_s(t+1) := F_s(x_s(t), q_s(t))$

$$F_s(t+1) = \left[x_s(t) + \frac{1}{T_s^2} - \frac{2}{3} q_s(t) x_s^2(t)\right]^+ \tag{9}$$

where T_s is the RTT of source s, i.e., the time it takes for s to send a packet and receive its acknowledgement from the destination. Here we assume T_s is a constant even though in reality its value depends on the congestion level and is generally time-varying. The quadratic term reflects the property that, if rate doubles, the multiplicative decrease occurs at twice the frequency with twice the amplitude.

The AQM mechanism of RED [43] maintains two internal variables, the instantaneous queue length $b_l(t)$ and average queue length $r_l(t)$. They are updated according to

$$b_l(t+1) = [b_l(t) + y_l(t) - c_l]^+ \tag{10}$$
$$r_l(t+1) = (1-\omega_l) r_l(t) + \omega_l b_l(t) \tag{11}$$

where $\omega_l \in (0,1)$. Then, (the "gentle" version of) RED marks a packet with a probability $\lambda_l(t)$ that is a piecewise linear, increasing the function of $r_l(t)$ with constants ρ_1, ρ_2, M_l, $\overline{b}_l$, and $\underline{b}_l$

$$\lambda_l(t) = \begin{cases} 0, & r_l(t) \leq \underline{b}_l \\ \rho_1\left(r_l(t) - \underline{b}_l\right), & \underline{b}_l \leq r_l(t) \leq \overline{b}_l \\ \rho_2\left(r_l(t) - \overline{b}_l\right) + M_l, & \overline{b}_l \leq r_l(t) \leq 2\overline{b}_l \\ 1, & r_l(t) \geq 2\overline{b}_l. \end{cases} \tag{12}$$

Equations (10)–(12) define the model $(\mathbf{G}, \mathbf{H})$ for RED.

TCP Vegas/DropTail: A duality model of Vegas has been developed and validated in [89]; see also [96]. We consider the situation where the buffer size is large enough to accommodate the equilibrium queue length so that Vegas sources can converge to the unique equilibrium. In this case, there is no packet loss in equilibrium.

Unlike TCP Reno, Vegas uses queueing delay as congestion measure $\lambda_l(t) = b_l(t)/c_l$, where $b_l(t)$ is the queue length at time t. The update rule $G_l(y_l(t), \lambda_l(t))$ is given by (dividing both sides of (10) by c_l)

$$\lambda_l(t+1) = \left[\lambda_l(t) + \frac{y_l(t)}{c_l} - 1\right]^+. \tag{13}$$

Hence, AQM for Vegas does not involve any internal variable. The update rule $F_s(x_s(t), q_s(t))$ for source rate is given by

$$x_s(t+1) = x_s(t) + \frac{1}{T_s^2(t)} \mathbf{1}(\alpha_s d_s - x_s(t) q_s(t)) \tag{14}$$

where α_s is a parameter of Vegas, d_s is the round-trip propagation delay of source s, and $\mathbf{1}(z) = 1$ if $z > 0$, -1 if $z < 0$, and 0 if $z = 0$. Here $T_s(t) = d_s + q_s(t)$ is the RTT at time t.

FAST/DropTail: The FAST algorithm is developed in [56], [57], and [147]. Let d_s denote the round-trip propagation delay of source s. Let $\lambda_l(t)$ denote the queueing delay at link l at time t. Let $q_s(t) = \sum_l R_{ls} \lambda_l(t)$ be the round-trip queueing delay, or in vector notation, $\mathbf{q}(t) = \mathbf{R}^T \boldsymbol{\lambda}(t)$. Each source s adapts its window $W_s(t)$ periodically according to

$$W_s(t+1) = \gamma\left(\frac{d_s W_s(t)}{d_s + q_s(t)} + \alpha_s\right) + (1-\gamma) W_s(t) \tag{15}$$

where $\gamma \in (0,1]$ and $\alpha_s > 0$ is a protocol parameter. A key departure from the model described above and those in the literature is that, here, we assume that a source's *send rate* cannot exceed the *throughput* it receives. This is justified because of self-clocking: within one RTT after a congestion window is increased, packet transmission will be clocked at the same rate as the throughput the flow receives. A consequence of this assumption is that the link queueing delay vector $\boldsymbol{\lambda}(t)$ is determined implicitly by the instantaneous window size in a *static* manner: given $W_s(t) = W_s$ for all s, the link queueing delays $\lambda_l(t) = \lambda_l \geq 0$ for all l are given by

$$\sum_s R_{ls} \frac{W_s}{d_s + q_s(t)} \quad \begin{cases} = c_l, & \text{if } \lambda_l(t) > 0 \\ \leq c_l, & \text{if } \lambda_l(t) = 0 \end{cases} \tag{16}$$

where again $q_s(t) = \sum_l R_{ls} \lambda_l(t)$.

Hence, FAST is defined by the discrete-time model (15), (16) of window evolution. The sending rate is then defined as $x_s(t) := W_s(t)/(d_s(t) + q_s(t))$.

2) Reverse-Engineering: Congestion Control as Distributed Solution of Basic NUM: Under mild assumptions on $(\mathbf{F}, \mathbf{G}, \mathbf{H})$, it can be shown using Kakutani's fixed point theorem that equilibrium $(\mathbf{x}, \boldsymbol{\lambda})$ of (6)–(8) exists and is unique [96], [134]. The fixed point of (6) defines an implicit relation between equilibrium rate x_s and end-to-end congestion measure q_s

$$x_s = F_s(x_s, q_s).$$

Assume F_s is continuously differentiable and $\partial F_s/\partial q_s \neq 0$ in the open set $A := \{(x_s, q_s)|x_s > 0, q_s > 0\}$. Then, by the implicit function theorem, there exists a unique continuously differentiable function f_s from $\{x_s > 0\}$ to $\{q_s > 0\}$ such that

$$q_s = f_s(x_s) > 0. \tag{17}$$

To extend the mapping between x_s and q_s to the closure of A, define

$$f_s(0) = \inf\{q_s \geq 0 | F_s(0, q_s) = 0\}. \tag{18}$$

If $(x_s, 0)$ is an equilibrium point $F_s(x_s, 0) = x_s$, then define

$$f_s(x_s) = 0. \tag{19}$$

Define the utility function of each source s as

$$U_s(x_s) = \int f_s(x_s) dx_s, \qquad x_s \geq 0 \tag{20}$$

which is unique up to a constant.

Being an integral, U_s is a continuous function. Since $f_s(x_s) = q_s \geq 0$ for all x_s, U_s is nondecreasing. We assume that f_s is a nonincreasing function—the more severe the congestion, the smaller the rate. This implies that U_s is concave. If f_s is strictly decreasing, then U_s is strictly concave since $U_s''(x_s) < 0$. An increasing utility function models a greedy source (a larger rate yields a higher utility) and its concavity models diminishing marginal return.

We assume the following conditions:

C1: For all $s \in S$ and $l \in L$, F_s and G_l are nonnegative functions. F_s are continuously differentiable and $\partial F_s/\partial q_s \neq 0$ in $\{(x_s, q_s)|x_s > 0, q_s > 0\}$; moreover, f_s in (17) are strictly decreasing.

C2: $\mathbf{R}$ has full row rank.

C3: If $\lambda_l = G_l(y_l, \lambda_l, \mathbf{v}_l)$ and $\mathbf{v}_l = \mathbf{H}_l(y_l, \lambda_l, \mathbf{v}_l)$, then $y_l \leq c_l$, with equality if $\lambda_l > 0$.

Condition C1 guarantees that $(\mathbf{x}(t), \boldsymbol{\lambda}(t)) \geq 0$ and $(\mathbf{x}^*, \boldsymbol{\lambda}^*) \geq 0$, and that utility functions U_s exist and are strictly concave. C2 guarantees uniqueness of equilibrium price vector $\boldsymbol{\lambda}^*$. C3 guarantees the primal feasibility and complementary slackness of $(\mathbf{x}^*, \boldsymbol{\lambda}^*)$. We can regard congestion control algorithms (6)–(8) as distributed algorithms to solve the NUM (4) and its dual (5) [87].

Theorem 1: Suppose assumptions C1 and C2 hold. Then (6)–(8) has a unique equilibrium $(\mathbf{x}^*, \boldsymbol{\lambda}^*)$. Moreover, it solves the primal problem (4) and the dual problem (5) with utility function given by (20) if and only if C3 holds.

Hence, the various TCP/AQM protocols can be modeled as different distributed solutions $(\mathbf{F}, \mathbf{G}, \mathbf{H})$ to solve (4) and its dual (5), with different utility functions U_s. Theorem 1 characterizes a large class of protocols $(\mathbf{F}, \mathbf{G}, \mathbf{H})$ that admit such an interpretation. This interpretation is the consequence of end-to-end control: it holds as long as the end-to-end congestion measure to which the TCP algorithm reacts is the *sum* of the constituent link congestion measures, and that the link prices are independent of sources (this would not be true in the heterogeneous protocol case as in Section II-A4). Note that the definition of utility function U_s depends only on TCP algorithm F_s. The role of AQM $(\mathbf{G}, \mathbf{H})$ is to ensure that the complementary slackness condition (condition C3) of problem (6)–(8) is satisfied. The complementary slackness has a simple interpretation: AQM should match input rate to capacity to maximize utilization at every bottleneck link. Any AQM that stabilizes queues possesses this property and generates a Lagrange multiplier vector $\boldsymbol{\lambda}^*$ that solves the dual problem.

The utility functions of several proposed TCP algorithms turn out to belong to a simple class of functions defined in [96] that is parameterized by a scalar parameter $\alpha_s \geq 0$

$$U_s(x_s) = \begin{cases} w_s \log x_s, & \alpha_s = 1 \\ w_s(1-\alpha_s)^{-1} x_s^{1-\alpha_s}, & \alpha_s \neq 1 \end{cases}$$

where weight $w_s > 0$. In particular, it has been shown that TCP Vegas, FAST, and Scalable TCP correspond to $\alpha_s = 1$, HTCP to $\alpha_s = 1.2$, TCP Reno to $\alpha_s = 2$, and maxmin fairness to $\alpha_s = \infty$. Maximizing α-fair utility leads to optimizers that satisfy the definition of α-fair resource allocation in the economics literature.

Method 1: Reverse-Engineering Cooperative Protocol as a Distributed Algorithm Solving a Global Optimization Problem.

The potentials and risks of networks come from the interconnection of local algorithms. Often, interesting and counter-intuitive behaviors arise in such a setting where users interact through multiple shared links in intricate and surprising ways. Reverse-engineering of TCP/AQM has also led to a deeper understanding of throughput and fairness behavior in large scale TCP networks. For example, there is a general belief that one can design systems to be efficient or fair, but not both. Many papers in the networking, wireless, and economics literature provide concrete examples in support of this intuition. The work in [132] proves an exact condition under which this conjecture is true for general TCP networks using the duality model of TCP/AQM. This condition allows us to produce the first counter-example and trivially explains all the supporting examples found in the literature. Surprisingly, in some counter-examples, a fairer throughput allocation is always more efficient. It implies for example that maxmin fair allocation can achieve higher aggregate throughput on certain networks. Intuitively, we might expect that the aggregate throughput will always rise as long as some links increase their capacities and no links decrease theirs. This turns out not to be the case, and [132] characterizes exactly the condition under which this is true in general TCP networks. Not only can the aggregate throughput be reduced when some link increases its capacity, more strikingly, it can also be reduced even when all links increase their capacities by the same amount. Moreover, this holds for all fair bandwidth allocations. This paradoxical result seems less surprising in retrospect: according to the duality model of TCP/AQM, raising link capacities always increases the aggregate utility, but mathematically there is no *a priori* reason that it should also increase the aggregate throughput. If all links increase their capacities proportionally, however, the aggregate throughput will indeed increase, for α-fair utility functions.

3) Stability of Distributed Solution: Theorem 1 characterizes the equilibrium structure of congestion control algorithm (6)–(8). We now discuss its stability. We assume conditions C1 and C2 in this section so that there is a unique equilibrium $(\mathbf{x}^*, \boldsymbol{\lambda}^*)$. In this section, an algorithm is said to be locally *asymptotically stable* if it converges to the unique equilibrium starting from a neighborhood of the equilibrium, and globally asymptotically stable if it converges starting from any initial state. Global asymptotic stability in the presence of feedback delay is desirable but generally hard to prove. Most papers in the literature analyze global asymptotic stability in the absence of feedback delay, or local stability in the presence of feedback delay. Proof techniques that have been used for global asymptotic stability in the absence of feedback delay include Lyapunov stability theorem, gradient decent method, passivity technique, and singular perturbation theory. In the following, we summarize some representative algorithms and illustrate how these methods are used to prove their stability in the absence of feedback delay. For analysis with delay, see, e.g., [102], [103], [140], and [141] for local stability of linearized systems and [90], [106], [107], and [115] for global stability; see also surveys in [63] [91], and [125] for further references. In particular, unlike the Nyquist argument, [107] handles nonlinearity and delay with Lyapunov functionals.

Consider the algorithm (using a continuous-time model) of [64]

$$\dot{x}_s = \kappa_s x_s(t)\left(U_s'(x_s(t)) - q_s(t)\right) \tag{21}$$
$$\lambda_l(t) = g_l(y_l(t)) \tag{22}$$

where $\kappa_s > 0$ is a constant of gain parameter. This is called a *primal-driven* algorithm, which means that there is dynamics only in the source control law but not the link control law. To motivate (21) and (22), note that $q_s(t)$ is the unit price for bandwidth that source s faces end-to-end. The marginal utility $U_s'(x_s(t))$ can be interpreted as source its willingness to pay when it transmits at rate $x_s(t)$. Then, according to (21), source s increases its rate (demand for bandwidth) if the end-to-end bandwidth price is less than its willingness to pay, and decreases it otherwise. Since g_l is an increasing function, the price increases as the aggregate demand for bandwidth at link l is large. To prove that (21) and (22) are globally asymptotically stable, consider the function

$$V(\mathbf{x}) := \sum_s U_s(x_s) - \sum_l \int_0^{y_l} g_l(z)dz. \tag{23}$$

Using (21) and (22), it is easy to check that

$$\dot{V} := \frac{d}{dt}V(\mathbf{x}(t)) = \begin{cases} > 0, & \text{for all } \mathbf{x}(t) \neq \mathbf{x}^* \\ = 0, & \text{if } \mathbf{x}(t) = \mathbf{x}^* \end{cases}$$

where $\mathbf{x}^*$ is the unique maximizer of the strictly concave function $V(\mathbf{x})$. Hence $V(\mathbf{x})$ is a Lyapunov function for the dynamical system (21), (22), certifying its global asymptotic stability. The function $V(\mathbf{x})$ in (23) can be interpreted as the penalty-function version of the NUM (4). Hence the algorithm in (21) and (22) can also be thought of as a gradient ascent algorithm to solve the approximate NUM.

Method 2: Lyapunov Function Construction to Show Stability.

A dual-driven algorithm is proposed in [90].

$$\lambda_l(t+1) = [\lambda_l(t) + \beta_0(y_l(t) - c_l)]^+ \tag{24}$$
$$x_s(t) = U_s'^{-1}(q_s(t)) \tag{25}$$

where $U_s'^{-1}$ is the inverse of U_s'. The algorithm is derived as the gradient projection algorithm to solve the dual (5) of NUM. The source algorithm (25) is called the demand function in economics: the larger the end-to-end bandwidth price $q_s(t)$, the smaller the demand $x_s(t)$. The link algorithm (24) is the law of supply and demand (for variable demand and fixed supply in this case): if demand $y_l(t)$ exceeds supply, increase the price $\lambda_l(t)$; otherwise, decrease it. By showing that the gradient $\nabla D(\boldsymbol{\lambda})$ of the dual objective function in (5) is Lipschitz, it is proved in [90] that, provided the stepsize β_0 is small enough, $\mathbf{x}(t)$ converges to the unique primal optimal solution of NUM and $\boldsymbol{\lambda}(t)$ converges to its unique dual solution. The idea is to show that the dual objective function $D(\boldsymbol{\lambda}(t))$ strictly decreases in each step t. Hence, one can regard $D(\boldsymbol{\lambda})$ as a Lyapunov function in discrete time.[3] The same idea is extended in [90] to prove global asymptotic stability in an asynchronous environment where the delays between sources and links can be substantial, diverse, and time-varying, sources and links can communicate at different times and with different frequencies, and information can be outdated or out of order.

Method 3: Proving Convergence of Dual Descent Algorithm Through Descent Lemma.

Several variations of the primal and *dual-driven* algorithms above can all maintain local stability in the presence of feedback delay [102], [103], [140], [141]. They are complementary in the sense that the primal-driven algorithm has dynamics only at the sources, allows arbitrary utility functions and therefore arbitrary fairness, but typically has low link utilization, whereas the dual-driven algorithm has dynamics only at the links, achieves full link utilization, but requires a specific class of utility functions (fairness) to maintain local stability in the presence of arbitrary feedback delays. The next algorithm has dynamics at both. It allows arbitrary utility functions, achieves arbitrarily close to full link utilization, and can maintain local stability in the presence of feedback delay. Algorithms that have dynamics at both links and sources are called primal-dual-driven algorithms. The algorithm of [71] extends the primal-driven algorithm (21), (22) to a *primal-dual-driven* algorithm and the algorithm of [103] extends the dual-driven algorithm (24), (25) to a primal-dual-driven algorithm. The paper [103] focuses on local stability in the presence of feedback delay. We now summarize the proof technique in [71] for global stability in the absence of feedback delay.

[3]Indeed, for a continuous-time version of (24) and (25), it is trivial to show that $D(\boldsymbol{\lambda})$ is a Lyapunov function.

The algorithm of [71] uses a source algorithm that is similar to (21)

$$\dot{x}_i(t) = w_i - \frac{1}{U_i'(x_i(t))}\sum_l R_{li}\lambda_l(t). \tag{26}$$

Its link algorithm adaptive virtual queue (AVQ) maintains an internal variable at each link called the virtual capacity $\tilde{c}_l$ that is dynamically updated

$$\dot{\tilde{c}}_l = \begin{cases} \frac{\gamma}{\partial g_l/\partial \tilde{c}_l}(c_l - y_l(t)), & \text{if } \tilde{c}_l \geq 0 \\ 0, & \text{if } \tilde{c}_l = 0 \text{ and } y_l(t) > c_l \end{cases} \tag{27}$$

where $\gamma > 0$ is a gain parameter and g_l is a link "marking" function that maps aggregate flow rate $y_l(t)$ and virtual capacity $\tilde{c}_l$ into a price

$$\lambda_l(t) = g_l(y_l(t), c_l(t)). \tag{28}$$

Using singular perturbation theory, it is proved in [71] that, under (26)–(28), $x(t)$ converges exponentially to the unique solution of the basic NUM, provided γ is small enough. Furthermore, $\lambda(t)$ then converges to the optimum of the dual problem. The idea is to separately consider the stability of two approximating subsystems that are at different time scales when γ is small. The boundary-layer system approximates the source dynamics and assumes that the virtual capacity $\tilde{c}_l$ are constants at the fast time scale

$$\dot{x}_s = w_s - \frac{1}{U_s'(x_s(t))}\sum_l R_{ls} g_l(y(t), \tilde{c}_l). \tag{29}$$

The reduced system approximates the link dynamics and assumes the source rates x_s are the unique maximizers of (23)

$$\dot{\tilde{c}}_l = c_l - y_l \tag{30}$$

where $y_l = \sum_l R_{ls}x_s$ are constants and x_s are the unique maximizers of $V(\mathbf{x})$ defined in (23). Now we already know from above that the boundary-layer system (29) is asymptotically stable. In [71], it is further shown that it is exponentially stable uniformly in $\tilde{\mathbf{c}}$, and that the reduced system (30) is exponentially stable provided the trajectory remains in a compact set. Singular perturbation theory then implies that the original system (26)–(28) is globally exponentially stable provided γ is small enough (and the initial state $(\mathbf{x}(0), \boldsymbol{\lambda}(0))$ is in a compact set).

Method 4: Proving Stability by Singular Perturbation Theory.

A different approach is used in [148] to prove global asymptotic stability for primal-dual-driven algorithms based on passivity techniques. A system, described by its state $\mathbf{z}(t)$, input $\mathbf{u}(t)$ and output $\mathbf{v}(t)$, is called *passive* if there are positive semidefinite functions $V(\mathbf{x}) \geq 0$ and $W(\mathbf{x}) \geq 0$ such that

$$\dot{V}(\mathbf{x}(t)) \leq -W(\mathbf{x}(t)) + \mathbf{u}^T(t)\mathbf{v}(t).$$

$V(\mathbf{x})$ is called a storage function. The passivity theorem states that the feedback interconnection of two passive systems is globally asymptotically stable and

$$V(\mathbf{x}) := V_1(\mathbf{x}) + V_2(\mathbf{x})$$

is a Lyapunov function for the feedback system, provided one of the storage functions V_1, V_2 of the individual systems are positive definite and radially unbounded. Consider the following variants of the primal-driven algorithm (21), (22):

$$\dot{x}_s(t) = \kappa_s\left(U_s'(x_s(t)) - q_s(t)\right) \tag{31}$$
$$\lambda_l(t) = g_l(y_l(t)). \tag{32}$$

To show that it is the feedback interconnection of two passive systems, the trick is to consider the forward system from $\boldsymbol{\lambda}(t) - \boldsymbol{\lambda}^*$ to $\dot{\mathbf{y}}(t)$, and the backward system from $\dot{\mathbf{y}}(t)$ to $\boldsymbol{\lambda}(t) - \boldsymbol{\lambda}^*$. From $\boldsymbol{\lambda}(t) - \boldsymbol{\lambda}^*$ to $\dot{\mathbf{y}}(t)$, the storage function is

$$V_1(\mathbf{x}) = \sum_s x_s q_s^* - U_s(x_s).$$

Then $V_1(x)$ is a positive definite function since its Hessian is a positive definite matrix for all $\mathbf{x}$. Moreover, it can be shown, using $\mathbf{q}(t) = \mathbf{R}^T\boldsymbol{\lambda}(t)$, that

$$\dot{V}_1(\mathbf{x}) = -\sum_s \kappa_s\left(q_s(t) - U_s'(x_s(t))\right)^2 + (\boldsymbol{\lambda}(t) - \boldsymbol{\lambda}^*)^T\dot{\mathbf{y}}$$

hence the forward system from $\boldsymbol{\lambda}(t) - \boldsymbol{\lambda}^*$ to $\dot{\mathbf{y}}(t)$ is passive. For the reverse system, consider the storage function

$$V_2(\mathbf{y} - \mathbf{y}^*) = \sum_l \int_{y_l}^{y_l^*} g_l(z) - g_l(z^*)dz.$$

V_2 is positive semidefinite since its Hessian is a positive semidefinite matrix. Moreover

$$\dot{V}_2 = (\boldsymbol{\lambda}(t) - \boldsymbol{\lambda}^*)^T\dot{\mathbf{y}}$$

and, hence, the reverse system is passive. Then, $V(\mathbf{x}) := V_1(\mathbf{x}) + V_2(\mathbf{x})$ can be used as a Lyapunov function for the feedback system, because

$$\dot{V} = -\sum_s \kappa_s\left(q_s(t) - U_s'(x_s(t))\right)^2 < 0,$$
$$\text{except for } \mathbf{x}(t) \equiv \mathbf{x}^*.$$

This implies global asymptotic stability.

The same argument proves the global asymptotic stability of the dual-driven algorithm (24), (25) [148]. Moreover, since the primal source algorithm from $\boldsymbol{\lambda} - \boldsymbol{\lambda}^*$ to $\mathbf{y} - \mathbf{y}^*$ and the dual link algorithm from $\mathbf{y} - \mathbf{y}^*$ to $\boldsymbol{\lambda} - \boldsymbol{\lambda}^*$ are both passive, the passivity theorem asserts the global asymptotic stability of their feedback interconnection, i.e., that of the following primal-dual-driven algorithm:

$$\dot{x}_s = \kappa_s\left(U_s'(x_s(t)) - q_s(t)\right)_{x_s}^+$$
$$\dot{\lambda}_l = \gamma_l(y_l(t) - c_l)_{\lambda_l}^+$$

where $(h)_z^+ = 0$, if $z = 0$ and $h < 0$, and $= h$, otherwise. The global asymptotic stability of the AVQ algorithm (26) and (27) is similarly proved in [148].

Method 5: Proving Stability by Passivity Argument.

4) Heterogeneous Congestion Control Protocols: A key assumption in the current model (6)–(8) is that the link prices $\lambda_l(t)$ depend only on links but not sources, i.e., the sources are homogeneous in that, even though they may control their rates using different algorithms F_s, they all adapt to the same type of congestion signals, e.g., all react to loss probabilities, as in TCP Reno, or all to queueing delay, as in TCP Vegas or FAST. When sources with *heterogeneous* protocols that react to different congestion signals share the same network, the current convex optimization and duality framework is no longer applicable. This is modeled in [133] and [134] by introducing price mapping functions m_l^s that maps link prices λ_l to "effective prices" seen by source s. However, one can no longer interpret congestion control as a distributed solution of the basic NUM when there are heterogeneous protocols. In this section, we summarize the main results of [134] on the equilibrium structure of heterogeneous protocols. Dynamic properties have also recently been characterized.

Suppose there are J different protocols indexed by superscript j, and N^j sources using protocol j, indexed by (j,s), where $j=1,\ldots,J$ and $s=1,\ldots,N^j$. The total number of sources is $N := \sum_j N^j$. The $L \times N^j$ routing matrix R^j for type j sources is defined by $R_{ls}^j = 1$ if source (j,s) uses link l, and 0, otherwise. The overall routing matrix is denoted by

$$\mathbf{R} = [\mathbf{R}^1 \quad \mathbf{R}^2 \quad \cdots \quad \mathbf{R}^J].$$

Every link l has an "intrinsic price" λ_l. A type j source reacts to the "effective price" $m_l^j(\lambda_l)$ in its path, where m_l^j is a price mapping function, which can depend on both the link and the protocol type. By specifying functions m_l^j, we can let the link feed back different congestion signals to sources using different protocols, for example, Reno with packet losses and Vegas with queueing delay. Let $\mathbf{m}^j(\boldsymbol{\lambda}) = (m_l^j(\boldsymbol{\lambda}_l), l = 1, \ldots L)$ and $\mathbf{m}(\boldsymbol{\lambda}) = (\mathbf{m}^j(\boldsymbol{\lambda}_l), j = 1, \ldots J)$.

The aggregate prices for source (j,s) is defined as

$$q_s^j = \sum_l R_{ls}^j m_l^j(\lambda_l). \tag{33}$$

Let $\mathbf{q}^j = (q_s^j, s = 1, \ldots, N^j)$ and $\mathbf{q} = (\mathbf{q}^j, j = 1 \ldots, J)$ be vectors of aggregate prices. Then $\mathbf{q}^j = (\mathbf{R}^j)^T \mathbf{m}^j(\boldsymbol{\lambda})$ and $\mathbf{q} = \mathbf{R}^T \mathbf{m}(\boldsymbol{\lambda})$. Let $\mathbf{x}^j$ be a vector with the rate x_s^j of source (j,s) as its sth entry, and $\mathbf{x}$ be the vector of $\mathbf{x}^j$

$$\mathbf{x} = \left[(\mathbf{x}^1)^T, (\mathbf{x}^2)^T, \ldots, (\mathbf{x}^J)^T\right]^T.$$

Source (j,s) has a utility function $U_s^j(x_s^j)$ that is strictly concave increasing in its rate x_s^j. Let $\mathbf{U} = (U_s^j, s = 1, \ldots, N^j, j = 1, \ldots, J)$. We call $(\mathbf{c}, \mathbf{m}, \mathbf{R}, \mathbf{U})$ a *network* with heterogeneous congestion control protocols.

A network is in equilibrium, or the link prices $\boldsymbol{\lambda}$ and source rates $\mathbf{x}$ are in equilibrium, when each source (j,s) maximizes its net benefit (utility minus bandwidth cost), and the demand for and supply of bandwidth at each bottleneck link are balanced. Formally, a network equilibrium is defined as follows.

Given any prices $\boldsymbol{\lambda}$, we assume that the source rates x_s^j are uniquely determined by

$$x_s^j(q_s^j) = \left[\left(U_s^j\right)'^{-1}\left(q_s^j\right)\right]^+.$$

This implies that the source rates x_s^j uniquely solve $\max_{z \geq 0} [U_s^j(z) - z q_s^j]$. As usual, we use $\mathbf{x}^j(\mathbf{q}^j) = (x_s^j(q_s^j), s = 1, \ldots, N^j)$ and $\mathbf{x}(\mathbf{q}) = (\mathbf{x}^j(\mathbf{q}^j), j = 1, \ldots, J)$ to denote the vector-valued functions composed of x_s^j. Since $\mathbf{q} = \mathbf{R}^T \mathbf{m}(\boldsymbol{\lambda})$, we often abuse notation and write $x_s^j(\lambda)$, $\mathbf{x}^j(\boldsymbol{\lambda})$, $\mathbf{x}(\boldsymbol{\lambda})$. Define the aggregate source rates $\mathbf{y}(\boldsymbol{\lambda}) = (y_l(\boldsymbol{\lambda}), l = 1, \ldots, L)$ at links l by

$$\mathbf{y}^j(\boldsymbol{\lambda}) = \mathbf{R}^j \mathbf{x}^j(\boldsymbol{\lambda}), \quad \mathbf{y}(\boldsymbol{\lambda}) = \mathbf{R}\mathbf{x}(\boldsymbol{\lambda}). \tag{34}$$

In equilibrium, the aggregate rate at each link is no more than the link capacity, and they are equal if the link price is strictly positive. Formally, we call $\boldsymbol{\lambda}$ an *equilibrium price*, a *network equilibrium*, or just an *equilibrium* if it satisfies [from (33) and (34)]

$$\operatorname{diag}(\lambda_l)(\mathbf{y}(\boldsymbol{\lambda}) - \mathbf{c}) = 0, \quad \mathbf{y}(\boldsymbol{\lambda}) \leq \mathbf{c}, \quad \boldsymbol{\lambda} \geq 0. \tag{35}$$

The theory in Section II-A2 corresponds to $J = 1$. When there are $J > 1$ types of prices, it breaks down because there cannot be more than one Lagrange multiplier at each link. In general, an equilibrium no longer maximizes aggregate utility, nor is it unique. It is proved in [134] that, under mild assumptions, an equilibrium always exists. There can be networks $(\mathbf{R}, \mathbf{c}, \mathbf{m}, \mathbf{U})$ that have uncountably many equilibria, but except for a set of measure zero, all networks have finitely many equilibria. Moreover, the Poincare–Hopf index theorem implies that the number of equilibria is necessarily odd. Specifically, suppose the following assumptions hold:

C4: Price mapping functions m_l^j are continuously differentiable in their domains and strictly increasing with $m_l^j(0) = 0$.

C5: For any $\epsilon > 0$, there exists a number $\lambda_{\max}$ such that if $\lambda_l > \lambda_{\max}$ for link l, then

$$x_i^j(\boldsymbol{\lambda}) < \epsilon \qquad \text{for all} \quad (j,i) \quad \text{with} \quad \mathbf{R}_{li}^j = 1.$$

C6: Every link l has a single-link flow (j,i) with $(U_i^j)'(c_l) > 0$.

Assumption C6 can be relaxed; see [124]. We call an equilibrium $\boldsymbol{\lambda}^*$ *locally unique* if $\partial \mathbf{y}/\partial \boldsymbol{\lambda} \neq 0$ at $\boldsymbol{\lambda}^*$. We call a network $(\mathbf{c}, \mathbf{m}, \mathbf{R}, \mathbf{U})$ *regular* if all equilibrium points are locally unique.

Theorem 2:
1) There exists an equilibrium price $\boldsymbol{\lambda}^*$ for any network $(\mathbf{c}, \mathbf{m}, \mathbf{R}, \mathbf{U})$.
2) Moreover, the set of link capacities $\mathbf{c}$ for which not all equilibrium points are locally unique (i.e., the network is not regular) has Lebesgue measure zero in $\mathbf{R}_+^L$.
3) A regular network has a finite and odd number of equilibrium points.

Despite the lack of an underlying NUM, heterogeneous protocols are still Pareto efficient for general networks. Moreover, the loss of optimality can be bounded in terms of the slope of the price mapping functions m_l^j. Specifically, suppose we use the optimal objective value U^* of the following NUM as a measure of optimality for heterogeneous protocols:

$$\begin{aligned} \text{maximize} \quad & \sum_j \sum_s U_s^j\left(x_s^j\right) \\ \text{subject to} \quad & \mathbf{R}\mathbf{x} \leq \mathbf{c}. \end{aligned} \qquad (36)$$

Let $U(\boldsymbol{\lambda}^*) := \sum_j \sum_s U_s^j(x_s^j(\boldsymbol{\lambda}^*))$ be the utility achieved by any equilibrium $\boldsymbol{\lambda}^*$ of the heterogeneous protocol. Then it can be shown that, for any equilibrium $\boldsymbol{\lambda}^*$

$$\frac{U(\boldsymbol{\lambda}^*)}{U^*} \geq \frac{\min \dot{m}_l^j(\boldsymbol{\lambda})}{\max \dot{m}_l^j(\boldsymbol{\lambda})}$$

where $\dot{m}_l^j$ denotes the derivative of m_l^j, and the minimization and maximization are over all types j, all links l used by all type j flows, and all prices $\boldsymbol{\lambda}$. For common AQM schemes such as RED with (piecewise) linear m_l^j, the bound reduces to a simple expression in terms of their slopes.

For a homogeneous congestion control protocol, the utility functions determine how bandwidth is shared among all the flows. For heterogeneous protocols, how is bandwidth shared among these protocols (interprotocol fairness), and how is it shared among flows within each protocol (intraprotocol fairness)? It is shown in [133] (and a generalization of results there) that any desired degree of fairness among the different protocols is achievable by appropriate linear scaling of utility functions. Within each protocol, the flows would share the bandwidth among themselves as if they were in a single-protocol network according to their own utility functions, except that the link capacities are reduced by the amount consumed by the other protocols. In other words, intraprotocol fairness is unaffected by the presence of other protocols.

Theorem 2 guarantees local unique equilibrium points for almost all networks under mild conditions. If the *degree of heterogeneity*, as measured by the slopes $\dot{m}_l^j$ of the price mapping functions m_l^j, is small, then global uniqueness is guaranteed: if $\dot{m}_l^j$ do not differ much across source types at each link, or they do not differ much along links in every source's path, the equilibrium is globally unique. Moreover, under this condition, global uniqueness is equivalent to local stability. Specifically, consider the dual-driven algorithm (in continuous-time)

$$\begin{aligned} \dot{\lambda}_l &= \gamma(y_l(t) - c_l) \\ x_s^j(t) &= U_s'^{-1}\left(q_s^j(t)\right) \end{aligned}$$

where the effective prices $q_s^j(t)$ are defined by (33) (compare with (24) and (25) in the homogeneous case). The linearized system with a small perturbation $\delta\boldsymbol{\lambda}$ around an equilibrium point $\boldsymbol{\lambda}^*$ is, in vector form

$$\delta\dot{\boldsymbol{\lambda}} = \gamma \frac{\partial y}{\partial \boldsymbol{\lambda}}(\boldsymbol{\lambda}^*)\delta\boldsymbol{\lambda}. \qquad (37)$$

The equilibrium $\boldsymbol{\lambda}^*$ is called *locally stable* if all the eigenvalues of $\partial y/\partial\boldsymbol{\lambda}(\boldsymbol{\lambda}^*)$ are in the left-half plane. Given the price mapping functions m_l^j, we say their degree of heterogeneity is small if they satisfy any one of the following conditions:

1) For each $l = 1, \ldots, L$, $j = 1, \ldots, J$

$$\dot{m}_l^j(\boldsymbol{\lambda}^*) \in \left[a_l, 2^{\frac{1}{L}} a_l\right]$$
$$\text{for some} \quad a_l > 0 \quad \text{for any equilibrium} \quad \boldsymbol{\lambda}^*. \qquad (38)$$

2) For all $j = 1, \ldots, J$, $l = 1, \ldots, L$

$$\dot{m}_l^j(\boldsymbol{\lambda}^*) \in \left[a^j, 2^{\frac{1}{L}} a^j\right]$$
$$\text{for some} \quad a^j > 0 \quad \text{for any equilibrium} \quad \boldsymbol{\lambda}^*. \qquad (39)$$

Theorem 3: For almost all networks $(\mathbf{c}, \mathbf{m}, \mathbf{R}, \mathbf{U})$:
1) Suppose their degree of heterogeneity is small, then the equilibrium is globally unique. Moreover, it is locally stable.
2) Conversely, if all equilibrium points are locally stable, it is also globally unique.

Asymptotically when $L \to \infty$, both conditions (38) and (39) converge to a single point. Condition (38) reduces to $\dot{m}_l^j = a_l$ which essentially says that all protocols are the same ($J = 1$). Condition (39) reduces to $\dot{m}_l^j = a^j$, which is the case where price mapping functions m_l^j are linear and link independent. Various special cases are shown to have a globally unique equilibrium in [134].

Method 6: Proving Equilibrium Properties Through Vector Field Representation and Poincare–Hopf Index Theorem.

Recall that since a network of homogeneous protocols solves the basic NUM, it always has a unique equilibrium point as long as the routing matrix R has full row rank. The equilibrium source rates $\mathbf{x}^*$ does not depend on link parameters, such as buffer size, as long as the AQM guarantees complementary slackness condition for the basic NUM. Moreover, $\mathbf{x}^*$ does not depend on the flow arrival pattern. These properties no longer hold in the heterogeneous case. We now present a simulation using Network Simulator 2 (ns2) that shows that $\mathbf{x}^*$ can depend on the flow arrival pattern because of the existence of multiple equilibria.

The topology of this network is shown in Fig. 1. All links run the RED algorithm. Links 1 and 3 are each configured with 9.1 pkts/ms capacity (equivalent to 111 Mb/s), 30 ms one-way propagation delay and a buffer of 1500 packets. The RED parameter is set to $(\underline{b}, \overline{b}, \rho_1) = (300, 1500, 10^{-4})$. Link 2 has a capacity of 13.8 pkts per ms (166 Mb/s) with 30 ms one-way propagation delay and buffer size of 1500 packets. RED parameter is set to (0, 1500, 0.1). There are 8 Reno flows on path 3 utilizing all the three links, with one-way propagation delay of 90 ms. There are two FAST flows on each of paths 1 and 2. Both of them have one-way propagation delay of 60 ms. All FAST flows use a common parameter value $\alpha = 50$ packets. Two sets of simulations have been carried out with different starting times for Reno and FAST flows. One set of flows (Reno or FAST) starts at time zero, and the other set starts at the 100th seconds. Fig. 2 shows the sample throughput trajectories of one of FAST flows and one of Reno flows. The large difference in the rate allocations of FAST and Reno between these two scenarios results from that the network reaches two different equilibrium points, depending on which type of flows starts first.

The model introduced in [133] and [134] is critical in deepening our understanding of such complex behavior, and providing design guidelines to manage it in practice. Indeed, a distributed algorithm is proposed in [135] that can steer a heterogeneous network to the unique equilibrium point that maximizes aggregate utility. The basic idea is simple: Besides regulating their rates according to their congestion signals, sources also adapt a parameter in a *slow time scale* based on a common congestion signal. This allows a source to choose a particular congestion signal in a fast time scale (and therefore maintain benefits associated with it) while asymptotically reaching the optimal equilibrium. The theoretical foundation and empirical supports of the algorithm are provided in [135].

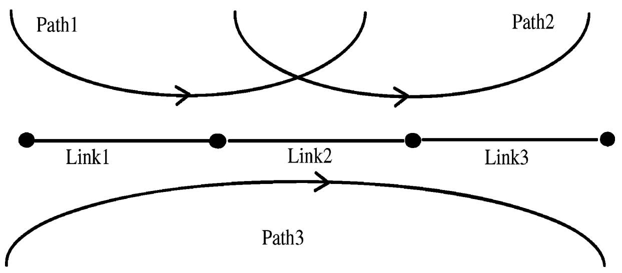


Fig. 1. ***Scenario with multiple equilibria with heterogeneous congestion control protocol.***

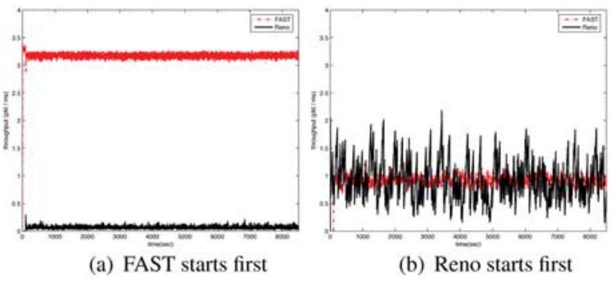


Fig. 2. ***Sample throughput trajectories of FAST and Reno. (a) FAST starts first; (b) Reno starts first.***

5) Forward-Engineering: FAST: The congestion control algorithm in the current TCP, which we refer to as Reno, was developed in 1988 [55] and has gone through several enhancements since. It has performed remarkably well and is generally believed to have prevented severe congestion as the Internet scaled up by six orders of magnitude in size, speed, load, and connectivity. It is also well-known, however, that as bandwidth-delay product continues to grow, TCP Reno will eventually become a performance bottleneck itself. Even though, historically, TCP Reno was designed, implemented, and deployed without any consideration of NUM, and its equilibrium, fairness, and dynamic properties were understood only as an afterthought, it indeed solves a NUM implicitly as shown in Section II-A2.

Several new algorithms have been proposed in the last few years to address the problems of Reno, including TCP Westwood, HSTCP [42], FAST TCP [56], [57], STCP [67], BIC TCP [155], HTCP [123], MaxNet [153], [154], XCP [62], and RCP [33], etc. (see [147] for other references). Some of these designs were explicitly guided by the emerging theory surveyed in this paper, which has become indispensable to the systematic design of new congestion control algorithms. It provides a framework to understand issues, clarify ideas and suggest directions, leading to more understandable and better performing implementations.

The congestion control mechanism of FAST TCP is separated into four components, as shown in Fig. 3. These four components are functionally independent so that they can be designed separately and upgraded asynchronously. The *data control* component determines *which* packets to

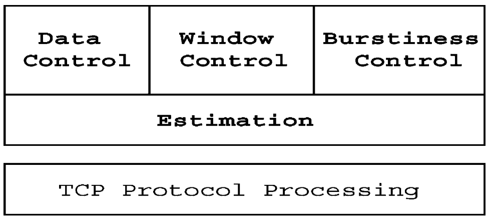


Fig. 3. ***Schematic of FAST TCP.***

transmit, *window control* determines *how many* packets to transmit, and *burstiness control* determines *when* to transmit these packets. These decisions are made based on information provided by the *estimation* component. More specifically, the estimation component computes two pieces of feedback information for each data packet sent—a multibit queueing delay and an one-bit loss-or-no-loss indication—which are used by the other three components. Data control selects the next packet to send from three pools of candidates: new packets, packets that are deemed lost (negatively acknowledged), and transmitted packets that are not yet acknowledged. Window control regulates packet transmission at the RTT time scale, while burstiness control smoothes out the transmission of packets at a smaller time scale. The theory surveyed in this paper forms the foundation of the window control algorithm. FAST periodically updates the congestion window based on the average RTT and average queueing delay provided by the estimation component, according to (15) in Section II-A1.

The equilibrium values of windows $\mathbf{W}^*$ and delays $\boldsymbol{\lambda}^*$ of the network defined by (15) and (16) are obtained from the unique solutions to the utility maximization problem over $\mathbf{x}$

$$\begin{aligned} \text{maximize} \quad & \sum_s w_s \log x_s \\ \text{subject to} \quad & \mathbf{R}\mathbf{x} \le \mathbf{c} \end{aligned}$$

and its Lagrangian dual problem over $\boldsymbol{\lambda}$

$$\text{minimize} \quad \sum_l c_l \lambda_l - \sum_s w_s \log \sum_l R_{ls}\lambda_l.$$

This implies that the equilibrium rate $\mathbf{x}^*$ is α_s-weighted proportionally fair. In equilibrium, source s maintains α_s packets in the buffers along its path. Hence, the total amount of buffering in the network must be at least $\sum_s \alpha_s$ packets in order to reach the equilibrium. FAST TCP is proved in [145] to be locally asymptotically stable for general networks if all flows have the same feedback delay, no matter how large the delay is. It is proved in [26] to be globally asymptotically stable in the presence of heterogeneous feedback delay at a single link.

We have implemented the insights from this series of theoretical work in a software prototype FAST TCP [56], [147] and have been working with our collaborators to test it in various networks around the world [57]. Physicists have been using FAST TCP to break various world records in data transfer in the last few years. Fig. 4 shows its performance in several experiments conducted during 2002–2005 over a high-speed trans-Atlantic network, over a home DSL, and over an emulated lossy link.

Table 5 Summary of Main Notation for Section II-B

Symbol	*Meaning*
W_l	Backoff window size at the transmitter of link l
CW_l	Current backoff window size
W_l^{max}	Maximum backoff window size
W_l^{min}	Minimum backoff window size
p_l	Persistence probability of link l
q_l	Conditional persistence probability
p_l^{max}	Maximum persistence probability
p_l^{min}	Minimum persistence probability
T_l	The event that link l transmits
C_l	The event that the transmission by link l is collided
β_l	Backoff multiplier
$S(\mathbf{p})$	Probability of successful transmission
$F(\mathbf{p})$	Probability of failed transmission
$R(p_l)$	Reward for successful transmission
$C(p_l)$	Cost of failed transmission
K	Maximum number of contending links
r_l	Receiving node of link l
t_l	Transmitting node of link l
$L_{out}(n)$	Set of egress links from node n
$L_{in}(n)$	Set of ingress links to node n
$L_{from}^I(n)$	Set of links interferred by node n
$N_{to}^I(l)$	Set of nodes whose transmission interfere link l
P^n	Persistence probability of node n
C_{CL}	Capacity of clique

B. MAC

1) Reverse-Engineering: MAC as Noncooperative Game: If contentions among transmissions on the same link in wired networks, or across different links in wireless networks, are not appropriately controlled, a large number of collisions may occur, resulting in waste of resources such as bandwidth and energy, as well as loss of system efficiency and fairness. There are two major types of MAC: scheduling-based contention-free mode and random-access-based contention-prone mode. The first is often shown to solve certain maximum weight matching or graph coloring problems. The second has been extensively studied through the perspective of queuing-theoretic performance evaluation, but was only recently reverse-engineered to recover the underlying utility maximization structure [75], [78].

In TCP reverse-engineering considered in the last section, the utility function of each source depends only on its data rate that can be directly controlled by the source itself. TCP/AQM can be modeled as a distributed algorithm that solves the basic NUM problem and its Lagrange dual problem.

In contrast, in the exponential-backoff (EB) MAC protocol, the utility of each link directly depends not only on its own transmission (e.g., persistence probability) but also transmissions of other links due to collisions. We show that the EB protocol can be reverse-engineered through a noncooperative game in which each link tries to maximize, using a stochastic subgradient formed by local information, its own utility function in the form of expected net reward for successful transmission. While the existence of the Nash equilibrium can be proved, neither convergence nor social welfare optimality is guaranteed. We then provide sufficient conditions on user density and backoff aggressiveness that guarantee uniqueness and stability of the Nash equilibrium (i.e., convergence of the standard best response strategy).

Consider an *ad hoc* network represented by a directed graph $G(V, E)$, e.g., as in Fig. 5, where V is the set of nodes and E is the set of logical links. We define $L_{\text{out}}(n)$ as a set of outgoing links from node n, $L_{\text{in}}(n)$ as a set of incoming links to node n, t_l as the transmitter node of link l, and r_l as the receiver node of link l. We also define $N^I_{\text{to}}(l)$ as the set of nodes whose transmission cause interference to the

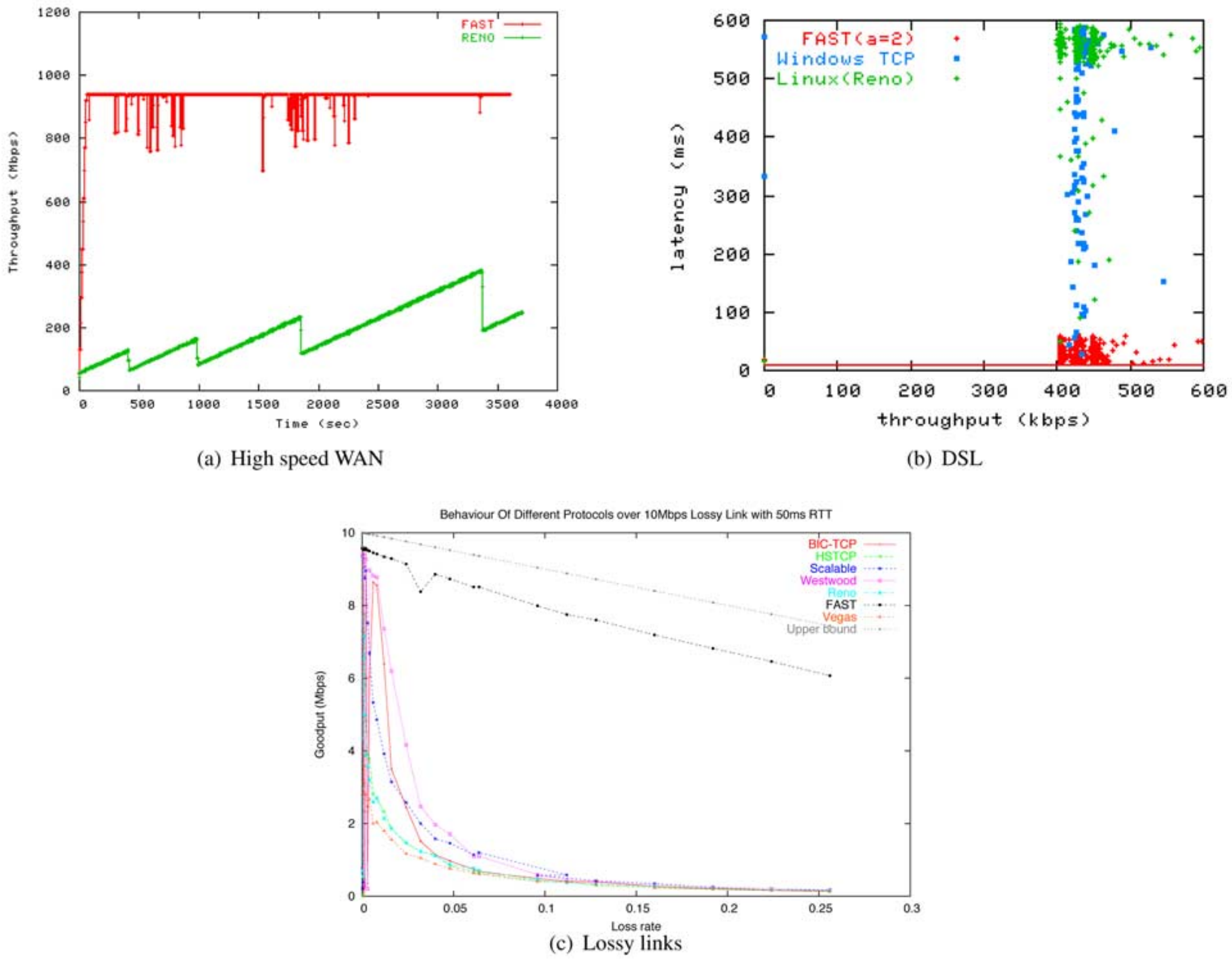


Fig. 4. ***Performance of FAST TCP. (a) At 1 Gb/s, FAST TCP utilized 95% of a trans-Atlantic network bandwidth while maintaining a fairly constant throughput. Linux TCP on average used 19% of the available bandwidth, while producing a throughput that fluctuates from 100 to 400 Mb/s. (b) At an 512-Kb/s DSL uplink, data transfer using FAST TCP increased the latency from 10 ms to around 50 ms, while Linux and Windows TCP increased it to as high as 600 ms, an order of magnitude larger. (c) Over an emulated lossy link, FAST TCP achieved close to optimal data rate while other (loss-based) TCP variants collapsed when loss rate exceeded 5%. Figure from unpublished work by B. Wydrowski, S. Hegde, and C. Jin.***

receiver of link l, excluding the transmitter node of link l (i.e., t_l), and $L^I_{\text{from}}(n)$ as the set of links whose transmission suffers interference from the transmission of node n, excluding outgoing links from node n (i.e., $l \in L_{\text{out}}(n)$). Hence, if the transmitter of link l and a node in set $N^I_{\text{to}}(l)$ transmit data simultaneously, the transmission of link l fails. If node n and the transmitter of link l in set $L^I_{\text{from}}(n)$ transmit data simultaneously, the transmission of link l also fails.

Random-access protocols in such wireless networks usually consist of two phases: contention avoidance and contention resolution. We focus only on the second phase here. The EB protocol is a prototypical contention resolution protocol. For example, in the IEEE 802.11 DCF (Distributed Coordination Function) implementation, the EB protocol is window-based: each link l maintains its contention window size W_l, current window size CW_l, and minimum and maximum window sizes $W_l^{\min}$ and $W_l^{\max}$. After each transmission, contention window size and current window size are updated. If transmission is successful, the contention window size is reduced to the minimum window size (i.e., $W_l = W_l^{\min}$), otherwise it is doubled until reaching the maximum window size $W_l^{\max}$ (i.e., $W_l = \min\{2W_l, W_l^{\max}\}$). Then, the current window size CW_l is chosen to be a number between $(0, W_l)$ uniformly at random. It decreases in every time slot, and when it becomes zero, the link transmits data. Since the window size is doubled after each transmission failure, the random access protocol in DCF is called the binary exponential backoff (BEB) protocol, which is a special case of EB protocols.

We study the window-based EB MAC protocol through a persistence probabilistic model, an approach analogous to the source rate model for the window-based TCP congestion control protocol in Section II-A2. Here each link l transmits data with a probability p_l, which we refer to as the persistence probability of link l. After each transmission attempt, if the transmission is successful without collisions, then link l sets its persistence probability to be its maximum value $p_l^{\max}$. Otherwise, it multiplicatively reduces its persistence probability by a factor β_l $(0 < \beta_l < 1)$ until reaching its minimum value $p_l^{\min}$. This persistence probability model is a memoryless one that approximates the average behavior of EB protocol.

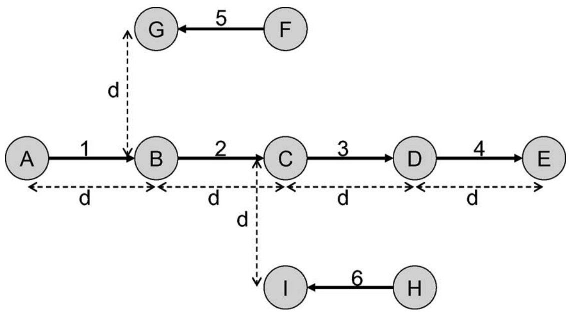


Fig. 5. ***Logical topology graph of a network illustrating contention.***

Since in the window-based EB protocol the current window size CW_l of link l is randomly selected between $(0, W_l)$, when its window size is W_l, we may think that link l transmits data in a time slot with an attempt probability $1/W_l$, which corresponds to the persistence probability p_l in our model for the average behavior of the EB protocols. In the window-based protocol, after every transmission success, the attempt probability is set to be its maximum value (i.e., $1/W_l^{\min}$), which corresponds to $p_l^{\max}$ in our model, and after every transmission failure, the attempt probability is set to be a fraction of its current value until it reaches its minimum value, which corresponds to reducing the persistence probability by a factor of $\beta = 0.5$ in BEB (and in general $\beta \in (0,1)$ in EB) until reaching the minimum persistence probability $p_l^{\min}$.

The update algorithm for the persistence probability described above can be written as

$$p_l(t+1) = \max\Big\{p_l^{\min}, p_l^{\max}\mathbf{1}_{\{T_l(t)=1\}}\mathbf{1}_{\{C_l(t)=0\}} + \beta_l p_l(t)\mathbf{1}_{\{T_l(t)=1\}}\mathbf{1}_{\{C_l(t)=1\}} + p_l(t)\mathbf{1}_{\{T_l(t)=0\}}\Big\} \quad (40)$$

where $p_l(t)$ is a persistence probability of link l at time slot t, $\mathbf{1}_a$ is an indicator function of event a, and $T_l(t)$ and $C_l(t)$ are the events that link l transmits data at time slot t and that there is a collision to link l's transmission given that link l transmits data at time slot t, respectively. In the rest of this section, we will examine the case when $p_l^{\min} = 0$. Given $\mathbf{p}(t)$, we have

$$\text{Prob}\{T_l(t) = 1 | \mathbf{p}(t)\} = p_l(t)$$

and

$$\text{Prob}\{C_l(t) = 1 | \mathbf{p}(t)\} = 1 - \prod_{n \in L^I_{\text{to}}(l)} (1 - p_n(t)).$$

Since the update of the persistence probabilities for the next time slot depends only on the current persistence probabilities, we will consider the update conditioning on the current persistence probabilities. Note that $p_l(t)$ is a random process whose transitions depend on events $T_l(t)$ and $C_l(t)$. We first study its expected trajectory and will return to (40) later in this section. Slightly abusing the notation, we still use $p_l(t)$ to denote the expected persistence probability. From (40), we have

$$\begin{aligned} p_l(t+1) =& p_l^{\max} E\left\{\mathbf{1}_{\{T_l(t)=1\}}\mathbf{1}_{\{C_l(t)=0\}}|\mathbf{p}(t)\right\} \\ &+ \beta_l E\left\{p_l(t)\mathbf{1}_{\{T_l(t)=1\}}\mathbf{1}_{\{C_l(t)=1\}}|\mathbf{p}(t)\right\} \\ &+ E\left\{p_l(t)\mathbf{1}_{\{T_l(t)=0\}}|\mathbf{p}(t)\right\} \\ =& p_l^{\max} p_l(t) \prod_{n\in L_{\text{to}}^I(l)} (1-p_n(t)) \\ &+ \beta_l p_l(t) p_l(t) \left(1 - \prod_{n\in L_{\text{to}}^I(l)} (1-p_n(t))\right) \\ &+ p_l(t)(1-p_l(t)) \end{aligned} \qquad (41)$$

where $E\{a|b\}$ is the expected value of a given b and $\mathbf{1}$ denotes the indicator function of probabilistic events.

We now reverse-engineer the update algorithm in (41) as a game, in which each link l updates its strategy, i.e., its persistence probability p_l, to maximize its utility U_l based on strategies of the other links, i.e., $\mathbf{p}_{-l} = (p_1, \cdots, p_{l-1}, p_{l+1}, \cdots, p_{|E|})$. Formally, the game is $G_{\text{EB-MAC}} = [E, \times_{l\in E} A_l, \{U_l\}_{l\in E}]$, where E is a set of players, i.e., links, $A_l = \{p_l | 0 \le p_l \le p_l^{\max}\}$ is an action set of player l, and U_l is a utility function of player l to be determined through reverse-engineering.

Theorem 4: The utility function is the following expected net reward (expected reward minus expected cost) that the link can obtain from its transmission:

$$U_l(\mathbf{p}) = R(p_l)S(\mathbf{p}) - C(p_l)F(\mathbf{p}), \quad \forall l \qquad (42)$$

where $S(\mathbf{p}) = p_l \prod_{n\in L_{\text{to}}^I(l)}(1-p_n)$ is the probability of transmission success, $F(\mathbf{p}) = p_l(1-\prod_{n\in L_{\text{to}}^I(l)}(1-p_n))$ is the probability of transmission failure, and $R(p_l) \overset{\text{def}}{=} p_l((1/2)p_l^{\max} - (1/3)p_l)$ can be interpreted as the reward for transmission success, $C(p_l) \overset{\text{def}}{=} (1/3)(1-\beta_l)p_l^2$ can be interpreted as the cost for transmission failure.

Furthermore, there exists a Nash equilibrium in the EB-MAC Game $G_{\text{EB-MAC}} = [E, \times_{l\in E} A_l, \{U_l\}_{l\in E}]$ characterized by the following system of equations:

$$p_l^* = \frac{p_l^{\max} \prod_{n\in L_{\text{to}}^I(l)}\left(1-p_n^*\right)}{1-\beta_l\left(1-\prod_{n\in L_{\text{to}}^I(l)}\left(1-p_n^*\right)\right)}, \quad \forall l. \qquad (43)$$

Note that the expressions of $S(\mathbf{p})$ and $F(\mathbf{p})$ come directly from the definitions of success and failure probabilities, while the expressions of $R(p_l)$ and $C(p_l)$ (thus exact form of U_l) are in fact *derived* in the proof by reverse-engineering the EB protocol description.

In the EB protocol, there is no explicit message passing among links, and the link cannot obtain the exact information to evaluate the gradient of its utility function. Instead of using the exact gradient of its utility function as in (41), each link attempts to approximate it using (40). It can be shown [76], [78] that the EB protocol described by (40) is a stochastic subgradient algorithm to maximize utility (42).

Method 7: Reverse-Engineer a Noncooperative Protocol as a Game.

The next step is to investigate uniqueness of the Nash equilibrium together with the convergence of a natural strategy for the game: the best response strategy, commonly used to study stability of the Nash equilibrium. In best response, each link updates its persistence probability for the next time slot such that it maximizes its utility based on the persistence probabilities of the other links in the current time slot

$$p_l^*(t+1) = \underset{0\le p_l\le p_l^{\max}}{\arg\max}\ U_l\left(p_l, \mathbf{p}_{-l}^*(t)\right). \qquad (44)$$

Hence, $p_l^*(t+1)$ is the best response of link l given $\mathbf{p}_{-l}^*(t)$. The connection between the best response strategy and stochastic subgradient update strategy has been quantified for the EB MAC Game [78].

Let $K = \max_l\{|L_{\text{to}}^I(l)|\}$, which captures the amount of potential contention among links. We have the following theorem that relates three key quantities: amount of potential contention K, backoff multiplier β (speed of backoff), and $p^{\max}$ that corresponds to the minimum contention window size (minimum amount of backoff).

Theorem 5: If $p^{\max}K/4\beta(1-p^{\max}) < 1$, then
1) the Nash equilibrium is unique;
2) starting from any initial point, the iteration defined by best response converges to the unique equilibrium.

There are several interesting engineering implications from the above theorem. For example, it provides guidance on choosing parameters in the EB protocols, and quantifies the intuition that with a large enough β (i.e., links do not decrease the probabilities suddenly) and a small enough $p^{\max}$ (i.e., links backoff aggressively enough), uniqueness and stability can be ensured. The higher the amount of contention (i.e., a larger value of K), the smaller $p^{\max}$ needs to be. The key idea in the proof is to show the updating rule from $p(t)$ to $p(t+1)$ is a contraction mapping by verifying the infinity norm of the Jacobian of the update dynamics in the game is less than one.

Method 8: Verifying Contraction Mapping by Bounding the Jacobian's Norm.

Reverse-engineering for the vertical interaction between TCP Reno congestion control and 802.11 DCF random access has also been carried out [168].

As will be discussed in Section V, session level stochastic effects need to be incorporated in the above reverse-engineering model to include the arrival statistics of finite-duration sessions. Then MAC protocols can be analyzed and designed through a union of stochastic stability results in traditional queuing models and optimality results in the utility maximization models.

2) Forward-Engineering: Utility-Optimal MAC Protocol: The Nash equilibrium attained by existing EB MAC protocols may not be socially optimal. This motivates forward-engineering where adequate feedback is generated to align selfish utility maximization by each logical link to maximize the social welfare in terms of total network utility. By imposing different utility functions, different types of services and different efficiency-fairness tradeoffs can be provisioned. Two suites of protocols are possible: scheduling-based and random-access-based. We again focus on the second in this subsection on forward-engineering.

Contentions among links can be modeled by using a contention graph first proposed in [98]. An example is shown in Fig. 6, which is obtained from Fig. 5 assuming that if the distance between the receiver of one link and the transmitter of the other link is less than $2d$, there is interference between those two links. Each vertex in the contention graph corresponds to a link in the network topology graph. If two links' transmissions interfere with each other, the vertices corresponding to them in the contention graph are connected with an edge. Only one link at a time among links in the same maximal clique in the contention graph can transmit data without collision. This constraint can be visualized by using a bipartite graph, as in Fig. 7, where one partition of vertices corresponds to links in the network (i.e., nodes in the contention graph) and the other corresponds to maximal cliques in the contention graph. An edge is established in the bipartite graph if a node in the contention graph belongs to a maximal clique. Hence, only network links represented by the nodes in the bipartite graph that are covered by a matching can transmit data simultaneously without collisions.

In [16], [39], and [98], a fluid approximation approach is used where each maximum clique is defined as a resource with a finite capacity that is shared by the links belonging to the clique. Capacity of a clique is defined as the maximum value of the sum of time fractions such that each link in the clique can transmit data without collision. Consequently, a generalized NUM problem has been formulated as follows, with capacity constraint C_{CL_i} at each maximal clique CL_i:

$$\begin{aligned} \text{maximize} \quad & \sum_l U_l(x_l) \\ \text{subject to} \quad & \sum_{l \in L(CL_i)} \frac{x_l}{c_l} \le C_{CL_i} \quad \forall i. \end{aligned} \tag{45}$$

This problem formulation essentially takes the same structure as the basic NUM (4) for TCP congestion control, and can be solved following the same dual-decomposition algorithm. We refer to this as the deterministic approximation approach.

An alternative approach is to explicitly model collision probabilities, as shown in [61] for log utility and in [76] and for general concave utility. Consider a random-access-based MAC protocol in which each node n adjusts its own persistence probability and also the persistence probability of each of its outgoing links. Since persistent transmission decisions are made distributively at each node, we need a shift from graph models based on logical links to graph models that incorporate nodes as well. Let P^n be the transmission probability of node n, and p_l be that of link l. The appropriate generalized NUM thus formulated is as follows, with variables $\{x_l\}$, $\{P^n\}$, $\{p_l\}$:

$$\begin{aligned} \text{maximize} \quad & \sum_l U_l(x_l) \\ \text{subject to} \quad & x_l = c_l p_l \prod_{k \in N_{\text{to}}^I(l)} (1 - P^k), \quad \forall l \\ & \sum_{l \in L_{\text{out}}(n)} p_l = P^n, \quad \forall n \\ & 0 \le P^n \le 1, \quad \forall n \\ & 0 \le p_l \le 1, \quad \forall l. \end{aligned} \tag{46}$$

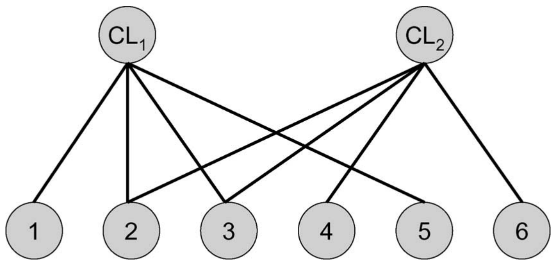


Fig. 7. ***Bipartite graph between maximal cliques and links in the contention graph.***

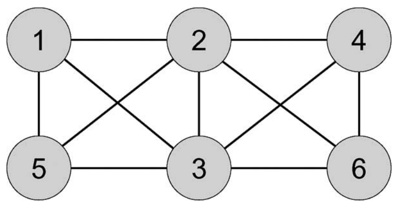


Fig. 6. ***Contention graph derived from the logical topology graph.***

Without loss of generality, we can replace the equality in the first constraint with an inequality. This is because such an inequality will always be achieved with an equality at optimality. The next step of problem transformation is to take the log of both sides of the first constraint in problem (46) and a log change of variables and constants: $x'_l = \log x_l$, $U'_l(x'_l) = U_l(e^{x'_l})$, and $c'_l = \log c_l$. This reformulation turns the problem into

$$\begin{aligned}
\text{maximize} \quad & \sum_{l \in L} U'_l(x'_l) \\
\text{subject to} \quad & c'_l + \log p_l + \sum_{k \in N^I_{\text{to}}(l)} \log(1 - P^k) - x'_l \geq 0, \quad \forall l \\
& \sum_{l \in L_{\text{out}}(n)} p_l = P^n, \quad \forall n \\
& 0 \leq P^n \leq 1, \quad \forall n \\
& 0 \leq p_l \leq 1, \quad \forall l.
\end{aligned} \tag{47}$$

Note that problem (47) is now separable but still may not be a convex optimization problem, since the objective $U'_l(x'_l)$ may not be a strictly concave function, even though $U_l(x_l)$ is a strictly concave function. However, the following simple sufficient condition guarantees its concavity:

$$\frac{\partial^2 U_l(x_l)}{\partial x_l^2} < -\frac{\partial U_l(x_l)}{x_l \partial x_l}$$

which states that the curvature (degree of concavity) of the utility function needs to be not just nonpositive but bounded away from zero by as much as $-(\partial U_l(x_l)/x_l \partial x_l)$, i.e., the application represented by this utility function must be elastic enough.

Method 9: Log Change of Variables for Decoupling, and Computing Minimum Curvature Needed for Concavity After the Change of Variables.

Following dual decomposition and the subgradient[4] method, the NUM problem (46) for random access MAC protocol design can be solved by the following algorithm.

Algorithm 1: **Utility Optimal Random Access Algorithm**

Each node n constructs its local interference graph to obtain sets $L_{\text{out}}(n)$, $L_{\text{in}}(n)$, $L^I_{\text{from}}(n)$, and $N^I_{\text{to}}(l)$, $\forall l \in L_{\text{out}}(n)$.

Each node n sets $t = 0$, $\lambda_l(1) = 1$, $\forall l \in L_{\text{out}}(n)$, $P^n(1) = |L_{\text{out}}(n)|/(|L_{\text{out}}(n)| + |L^I_{\text{from}}(n)|)$, and $p_l(1) = 1/(|L_{\text{out}}(n)| + |L^I_{\text{from}}(n)|)$, $\forall l \in L_{\text{out}}(n)$.

For each node n, do

1) Set $t \leftarrow t + 1$.

2) Inform $\lambda_l(t)$ to all nodes in $N^I_{\text{to}}(l)$, $\forall l \in L_{\text{out}}(n)$ and $P^n(t)$ to t_l, $\forall l \in L^I_{\text{from}}(n)$.

3) Set $k_n(t) = \sum_{l \in L_{\text{out}}(n)} \lambda_l(t) + \sum_{k \in L^I_{\text{from}}(n)} \lambda_k(t)$ and $\beta(t) = 1/t$.

4) Solve the following problems to obtain $P^n(t+1)$, and $x'_l(t+1)$, $p_l(t+1)$, and $\lambda_l(t+1)$, $\forall l \in L_{\text{out}}(n)$:

$$P^n(t+1) = \begin{cases} \dfrac{\sum_{l \in L_{\text{out}}(n)} \lambda_l(t)}{\sum_{l \in L_{\text{out}}(n)} \lambda_l(t) + \sum_{k \in L^I_{\text{from}}(n)} \lambda_k(t)}, & \text{if } k_n(t) \neq 0 \\[2ex] \dfrac{|L_{\text{out}}(n)|}{|L_{\text{out}}(n)| + |L^I_{\text{from}}(n)|}, & \text{if } k_n(t) = 0 \end{cases}$$

$$p_l(t+1) = \begin{cases} \dfrac{\lambda_l(t)}{\sum_{l \in L_{\text{out}}(n)} \lambda_l(t) + \sum_{k \in L^I_{\text{from}}(n)} \lambda_k(t)}, & \text{if } k_n(t) \neq 0 \\[2ex] \dfrac{1}{|L_{\text{out}}(n)| + |L^I_{\text{from}}(n)|}, & \text{if } k_n(t) = 0 \end{cases}$$

$$x'_l(t+1) = \underset{x'^{\min}_l \leq x' \leq x'^{\max}_l}{\arg\max} \left\{ U'_l(x'_l) - \lambda_l(t) x'_l \right\}$$

and

$$\lambda_l(t+1) = \left[\lambda_l(t) - \beta(t) \left(c'_l + \log p_l(t) + \sum_{k \in N^I_{\text{to}}(l)} \log\left(1 - P^k(t)\right) - x'_l(t) \right) \right].$$

5) Set its persistence probability $P^{n*} = P^n(t)$ and the conditional persistence probability of each of its outgoing links $q^*_l = p_l(t)/P^n(t)$.

6) Decide if it will transmit data with a probability P^{n*}, in which case it chooses to transmit on one of its outgoing links with a probability q^*_l, $\forall l \in L_{\text{out}}(n)$.

while (1).

Note that the above algorithm is conducted at each node n to calculate P^n, and p_l, λ_l, and x'_l for its outgoing link l (i.e., $\forall l \in L_{\text{out}}(n)$). Hence, it is conducted at the transmitter node of each link. If we assume that two nodes within interference range can communicate with each other (i.e., if nodes within distance $2d$ in Fig. 5 can establish a communication link), in the above algorithm each node requires information from nodes within two-hop distance from it. To calculate P^n and p_l for its outgoing link l (i.e., $\forall l \in L_{\text{out}}(n)$), node n needs λ_m from the transmitter node t_m of link m that is interfered from the transmission of node n (i.e., from t_m, $\forall m \in L^I_{\text{from}}(n)$). Note that t_m is within two-hop from node n.

[4]A subgradient of a (possibly nondifferentiable) function $f : \mathbf{R}^n \rightarrow \mathbf{R}$ at point $\mathbf{x}$ is a vector $\mathbf{g}$ such that $f(\mathbf{y}) \geq f(\mathbf{x}) + \mathbf{g}^T(\mathbf{y} - \mathbf{x})$, $\forall \mathbf{y}$.

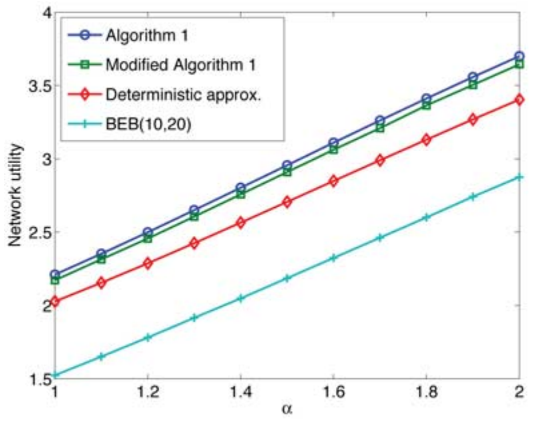


Fig. 8. ***Comparison of network utilities in a numerical example.***

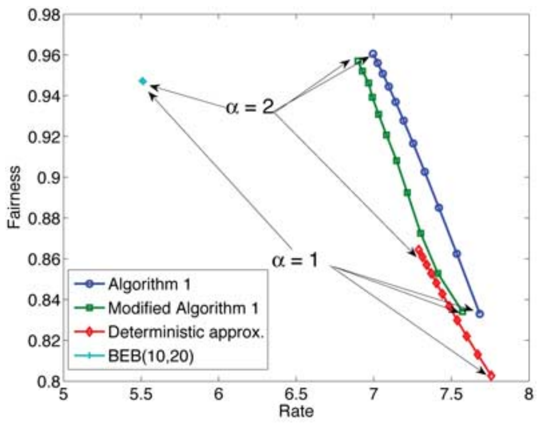


Fig. 9. ***Comparison of rate-fairness tradeoff in a numerical example.***

Alternatively, if λ_l and x_l' for each link l are calculated at its receiver node r_l instead of its transmitter node t_l, a modified version of Algorithm 1 can be devised in which each node requires information only within *one-hop* distance [76].

Theorem 6: Algorithm 1 converges to a globally optimal solution of (46) for sufficiently concave utility functions.

We now show a numerical example of the desired tradeoff between efficiency and fairness that can be achieved by appropriately adjusting the parameters of utility functions. In this experiment, the utility function for each link l, $U_l(x_l)$ is in the following standard form of concave utility parameterized by α, shifted such that $U_l(x_l^{\min}) = 0$ and $U_l(x_l^{\max}) = 1$

$$U_l(x_l) = \frac{x_l^{(1-\alpha)} - x_l^{\min(1-\alpha)}}{x_l^{\max(1-\alpha)} - x_l^{\min(1-\alpha)}}.$$

We set $x_l^{\min} = 0.5$ and $x_l^{\max} = 5$, $\forall l$, varying the value of α from 1 to 2 with a step size 0.1.

We compare the performances of Algorithm 1 and its one-hop message passing variant (modified Algorithm 1, not shown here) with deterministic fluid approximation and the BEB protocol in IEEE 802.11 standard.[5]

In Fig. 8, we compare the network utility achieved by each protocol. We show the tradeoff curve of rate and fairness for each protocol in Fig. 9. Here, the fairness index is $((\sum_l x_l^2)/|L| \sum_l x_l^2)$. For each protocol shown in the graph, the area to the left and below of the tradeoff curve is the achievable region (i.e., every (rate, fairness) point in this region can be obtained), and the area to the right and above of the tradeoff curve is the infeasible region (i.e., it is impossible to have any combination of (rate, fairness) represented by points in this region). It is impossible to operate in the infeasible region and inferior to operate in the interior of the achievable region. Operating on the boundary of the achievable region, i.e., the Pareto optimal tradeoff curve, is the best. Points on the Pareto optimal tradeoff curve are not comparable: which point is better depends on the desired tradeoff between efficiency and fairness. Since the BEB protocol is a static protocol, it always provides the same efficiency (rate) and fairness regardless of the choice of utility functions. Hence, we cannot flexibly control the efficiency-fairness tradeoff in the BEB protocol. Algorithm 1 and its variant achieve higher network utilities and wider dynamic ranges of rate-fairness tradeoff.

Further discussions on distributed, suboptimal scheduling algorithms for different interference models can be found in Sections III-D and V-A.

III. VERTICAL DECOMPOSITION

In this section, we turn to vertical decomposition across the protocol stack. Following is a *nonexhaustive* list of some of the recent publications using "Layering as Optimization Decomposition" for vertical decomposition.[6] Almost all of the following papers start with some generalized NUM formulations, and use either dual decomposition or primal penalty function approach to modularize and distribute the solution algorithm, followed by proofs of optimality, stability, and fairness. The individual modules in the

[5]The performance of the BEB protocol highly depends on the choice of maximum and minimum window sizes, $W_l^{\max}$ and $W_l^{\min}$. It turns out that for the network in Fig. 5, the average-performance parameters are: $W_l^{\max} = 20$ and $W_l^{\min} = 10$.

[6]Note that there are many more publications on joint design across these layers that did not use NUM modeling or decomposition theory. In addition, we apologize in advance for missing any references we should have included and would appreciate any information about such citations.

holistic solution range from adaptive routing and distributed matching to information-theoretic source coding and video signal processing, coupled through implicit or explicit message passing of functions of appropriate "layering prices": variables that coordinate the layers.

- joint congestion control and adaptive coding or power control [18], [21], [75];
- joint congestion and contention control [16], [61], [74], [143], [168]–[170];
- joint congestion control and scheduling [1], [12], [35], [92], [129];
- joint routing and power control [58], [100], [158];
- joint congestion control, routing, and scheduling [17], [35], [36], [82], [83], [99], [129];
- joint routing, scheduling, and power control [27], [156];
- joint routing, resource allocation, and source coding [53], [167];
- TCP/IP interactions [50], [49], [112], [144] and HTTP/TCP interactions [13];
- joint congestion control and routing [46], [49], [60], [65], [69], [101];
- network lifetime maximization [97].

Four case studies and the associated illustrative numerical examples are summarized here, each picked mainly to convey several key messages. We present more details on the first two cases, which mainly illustrate the applications to the analysis and design aspects, respectively. In all case studies, we first formulate generalized NUM problems to capture the interactions across such functional modules. These can be decomposed into subproblems, each of which is solved by a layer, and the interface between the layers represented by some function of the optimization variables. Then in Section IV-C, we will show that these case studies have only spanned a subset of alternative layering architectures.

Table 6 Summary of Main Notation for Section III

Symbol	*Meaning*
$\mathbf{H}^s, \mathbf{H}$	Physical connectivity matrices
$\mathbf{W}^s, \mathbf{W}$	Load balancing matrices
$\mathcal{W}_n$	Set of load balancing matrices with single-path routing
$\mathcal{W}_m$	Set of load balancing matrices with multi-path routing
$\mathcal{R}_n$	Set of routing matrices with single-path routing
$\mathcal{R}_m$	Set of routing matrices with multi-path routing
r^s	Set of routes available to source s
V_{np}	Optimized primal value of TCP/IP NUM with single-path routing
V_{nd}	Optimized dual value of TCP/IP NUM with single-path routing
V_{mp}	Optimized primal value of TCP/IP NUM with multi-path routing
V_{md}	Optimized dual value of TCP/IP NUM with multi-path routing
τ_l	Constant term in link weight
$f(y_l/c_l)$	Link congestion penalty function determined by network operators
P_j	Transmit power on link j
m_j	Message generated by node j
SIR_j	Signal to Interference Ratio received on link j
G_{ij}	Channel gain from transmitter on link j to receiver on link i
ρ_s	Reliability at source s
$\rho^s(t)$	Offered reliability at source s at time t
p^s	Probability of decoding error at source s
$p_{l,s}$	Probability of decoding error on link l for source s
$t_{l,s}$	Transmission rate on link k for source s
$r_{l,s}$	Code rate on link l for source s
c_l^{max}	Maximum capacity on link l
M	Codeword length
E_l	Error function mapping code rate to decoding error probability
μ_s	Signal quality price generated by source s
$c_{l,s}$	Capacity on link l allocated to source s
x'_s	Log transformed x_s
U'_s	Utility function of log transformed x_s
v	Parameter weighting preference between rate and reliability
κ	Penalty function weight
ϵ, δ	Message passing for joint congestion and contention control
$\mathbf{\Pi}$	Schedulability constraint
$\mathbf{f}$	Flow vector
x_i^k	Rate for source destination pair (i, k)
f_{ij}^k	Flow on link (i, j) for destination k
w_{ij}	Weighting for maximum weight matching

There are obviously many more case studies in the rapidly developing research literature in this area. Many of these are not covered in this survey, in part because of space limitation, and in part because we hope to highlight the concepts of the top-down approach to design layered architecture from first principles, rather than any set of specific cross-layer schemes or their performance enhancements. Case studies are surveyed here only to illustrate the *conceptual simplicity* in the *structured thinking* of "layering as decomposition," a simplicity that we hope will not be buried under the rich details in all these recent publications.

Even for these selected illustrative examples, there are many related works by various research groups. Our presentation is inevitably somewhat biased towards relying on the materials from publications by ourselves and coauthors.

A. Case Study 1: Jointly Optimal Congestion Control and Routing

The word "routing" carries different meanings in different parts of the research literature. It can refer to dynamic or static routing, single-path or multipath routing, distance-vector or link-state-based routing, inter-domain or intra-domain routing, fully distributed routing or centralized-computation-aided routing, and other types of routing in wireless *ad hoc* networks, optical networks, and the Internet. Several notions of routing will be used in the models in this section.

1) TCP/IP Interaction: Suppose that there are K^s acyclic paths from source s to its destination, represented by a $L \times K^s$ 0–1 matrix $\mathbf{H}^s$, where

$$H^s_{lj} = \begin{cases} 1, & \text{if path } j \text{ of source } s \text{ uses link } l \\ 0, & \text{otherwise.} \end{cases}$$

Let $\mathcal{H}^s$ be the set of all columns of $\mathbf{H}^s$ that represents all the available paths to s. Define the $L \times K$ matrix $\mathbf{H}$ as

$$\mathbf{H} = [\mathbf{H}^1 \quad \dots \quad \mathbf{H}^N]$$

where $K := \sum_s K^s$. $\mathbf{H}$ defines the physical topology of the network.

Let $\mathbf{w}^s$ be a $K^s \times 1$ vector where the jth entry represents the fraction of i's flow on its jth path such that

$$w^s_j \geq 0 \ \forall j \quad \text{and} \quad \mathbf{1}^T\mathbf{w}^s = 1$$

where $\mathbf{1}$ is a vector of an appropriate dimension with the value 1 in every entry. We require $w^s_j \in \{0,1\}$ for single path routing, and allow $w^s_j \in [0,1]$ for multipath routing. Collect the vectors $\mathbf{w}^s$, $s = 1, \dots, N$, into a $K \times N$ block-diagonal matrix $\mathbf{W}$. Let $\mathcal{W}_n$ be the set of all such matrices corresponding to single path routing, defined as

$$\left\{\mathbf{W} | \mathbf{W} = \operatorname{diag}(\mathbf{w}^1, \dots, \mathbf{w}^N) \in \{0,1\}^{K\times N}, \mathbf{1}^T\mathbf{w}^s = 1, \ \forall s.\right\}.$$

Define the corresponding set $\mathcal{W}_m$ for multipath routing as

$$\left\{\mathbf{W} | \mathbf{W} = \operatorname{diag}(\mathbf{w}^1, \dots, \mathbf{w}^N) \in [0,1]^{K\times N}, \mathbf{1}^T\mathbf{w}^s = 1, \ \forall s.\right\}. \tag{48}$$

As mentioned above, $\mathbf{H}$ defines the set of acyclic paths available to each source, and $\mathbf{W}$ defines how the sources load balance across these paths. Their product defines an $L \times N$ routing matrix $\mathbf{R} = \mathbf{HW}$ that specifies the fraction of s's flow at each link l. The set of all single-path routing matrices is

$$\mathcal{R}_n = \{\mathbf{R} | \mathbf{R} = \mathbf{HW}, \mathbf{W} \in \mathcal{W}_n\} \tag{49}$$

and the set of all multipath routing matrices is

$$\mathcal{R}_m = \{\mathbf{R} | \mathbf{R} = \mathbf{HW}, \mathbf{W} \in \mathcal{W}_m\}. \tag{50}$$

The difference between single-path routing and multipath routing is the integer constraint on $\mathbf{W}$ and $\mathbf{R}$. A single-path routing matrix in $\mathcal{R}_n$ is an 0-1 matrix:

$$R_{ls} = \begin{cases} 1, & \text{if link } l \text{ is in the path of source } s \\ 0, & \text{otherwise.} \end{cases}$$

A multipath routing matrix in $\mathcal{R}_m$ is one whose entries are in the range [0, 1]

$$R_{ls} \begin{cases} > 0, & \text{if link } l \text{ is in a path of source } s \\ = 0, & \text{otherwise.} \end{cases}$$

The path of source s is denoted by $\mathbf{r}^s = [R_{1s} \ \dots \ R_{Ls}]^T$, the sth column of the routing matrix $\mathbf{R}$. We now model the interaction of congestion control at the transport layer and shortest-path routing at the network layer.

We first consider the situation where TCP-AQM operates at a faster time scale than routing updates. We assume for now a *single* path is selected for each source-destination pair that minimizes the sum of the link costs in the path, for some appropriate definition of link cost. In particular, traffic is not split across multiple paths from the source to the destination even if they are available. We focus on the time scale of the route changes, and assume TCP-AQM is stable and converges instantly to equilibrium after a route change. As explained in the last section, we interpret the equilibria of various TCP and AQM algorithms as solutions of NUM and its dual.

Specifically, let $\mathbf{R}(t) \in \mathcal{R}_n$ be the (single-path) routing in period t. Let the equilibrium rates $\mathbf{x}(t) = \mathbf{x}(\mathbf{R}(t))$ and prices $\boldsymbol{\lambda}(t) = \boldsymbol{\lambda}(\mathbf{R}(t))$ generated by TCP-AQM in period t, respectively, be the optimal primal and dual solutions, i.e.,

$$\mathbf{x}(t) = \arg\max_{\mathbf{x}\geq 0} \sum_s U_s(x_s) \quad \text{subject to } \mathbf{R}(t)\mathbf{x} \leq \mathbf{c} \tag{51}$$

$$\boldsymbol{\lambda}(t) = \arg\min_{\boldsymbol{\lambda}\geq 0} \sum_s \max_{x_s\geq 0}\left(U_s(x_s) - x_s\sum_l R_{ls}(t)\lambda_l\right) + \sum_l c_l\lambda_l. \tag{52}$$

The link costs used in routing decision in period t are the congestion prices $\lambda_l(t)$. Each source computes its new route $\mathbf{r}^s(t+1) \in \mathcal{H}^s$ individually that minimizes the total cost on its path

$$\mathbf{r}^s(t+1) = \arg\min_{\mathbf{r}^s\in\mathcal{H}^s} \sum_l \lambda_l(t) r_l^s. \tag{53}$$

We say that $(\mathbf{R}^*, \mathbf{x}^*, \boldsymbol{\lambda}^*)$ is an *equilibrium of TCP/IP* if it is a fixed point of (51)–(53), i.e., starting from routing $\mathbf{R}^*$ and associated $(\mathbf{x}^*, \boldsymbol{\lambda}^*)$, the above iterations yield $(\mathbf{R}^*, \mathbf{x}^*, \boldsymbol{\lambda}^*)$ in the subsequent periods.

We now characterize the condition under which TCP/IP as modeled by (51)–(53) has an equilibrium. Consider the following generalized NUM:

$$\text{maximize}_{\mathbf{R}\in\mathcal{R}_n} \text{maximize}_{\mathbf{x}\geq 0} \sum_s U_s(x_s) \quad \text{subject to } \mathbf{R}\mathbf{x} \leq \mathbf{c} \tag{54}$$

and its Lagrange dual problem

$$\text{minimize}_{\boldsymbol{\lambda}\geq 0} \sum_s \max_{x_s\geq 0}\left(U_s(x_s) - x_s \min_{\mathbf{r}^s\in\mathcal{H}^s}\sum_l R_{ls}\lambda_l\right) + \sum_l c_l\lambda_l \tag{55}$$

where $\mathbf{r}^s$ is the sth column of $\mathbf{R}$ with $r_l^s = R_{ls}$. While (51) maximizes utility over source rates only, problem (54) maximizes utility over both rates and routes. While (51) is a convex optimization problem without duality gap, problem (54) is nonconvex because the variable $\mathbf{R}$ is discrete, and generally has a duality gap.[7] The interesting feature of the dual problem (55) is that the maximization over $\mathbf{R}$ takes the form of minimum-cost routing with congestion prices $\boldsymbol{\lambda}$ generated by TCP-AQM as link costs. This suggests that TCP/IP might turn out to be a distributed algorithm that attempts to maximize utility, with a proper choice of link costs. This is indeed true, provided that an equilibrium of TCP/IP actually exists.

Theorem 7: An equilibrium $(\mathbf{R}^*, \mathbf{x}^*, \boldsymbol{\lambda}^*)$ of TCP/IP exists if and only if there is no duality gap between (54) and (55). In this case, the equilibrium $(\mathbf{R}^*, \mathbf{x}^*, \boldsymbol{\lambda}^*)$ is a solution of (54) and (55).

Method 10: Analyzing a Given Cross-Layer Interaction Through Generalized NUM.

Hence, one can regard the layering of TCP and IP as a decomposition of the NUM problem over source rates and routes into a distributed and decentralized algorithm, carried out on two different time scales, in the sense that an equilibrium of the TCP/IP iteration (51)–(53), if it exists, solves (54) and (55). However, an equilibrium may not exist. Even if it does, it may not be stable [144].

The duality gap can be interpreted as a measure of "cost for not splitting." To elaborate, consider the Lagrangian

$$L(\mathbf{R}, \mathbf{x}, \boldsymbol{\lambda}) = \sum_s \left(U_s(x_s) - x_s\sum_l R_{ls}\lambda_l\right) + \sum_l c_l\lambda_l.$$

The primal (54) and dual (55) can then be expressed, respectively, as

$$V_{np} = \max_{\mathbf{R}\in\mathcal{R}_n,\ \mathbf{x}\geq 0} \min_{\boldsymbol{\lambda}\geq 0} L(\mathbf{R}, \mathbf{x}, \boldsymbol{\lambda})$$

$$V_{nd} = \min_{\boldsymbol{\lambda}\geq 0} \max_{\mathbf{R}\in\mathcal{R}_n,\ \mathbf{x}\geq 0} L(\mathbf{R}, \mathbf{x}, \boldsymbol{\lambda}).$$

[7]The nonlinear constraint $\mathbf{R}\mathbf{x} \leq \mathbf{c}$ can be converted into a linear constraint (see proof of Theorem 8 in [144]), so the integer constraint on $\mathbf{R}$ is the real source of difficulty.

If we allow sources to distribute their traffic among multiple paths available to them, then the corresponding problems for multipath routing are

$$V_{mp} = \max_{\mathbf{R}\in\mathcal{R}_m,\ \mathbf{x}\geq 0} \min_{\boldsymbol{\lambda}\geq 0} L(\mathbf{R}, \mathbf{x}, \boldsymbol{\lambda})$$
$$V_{md} = \min_{\boldsymbol{\lambda}\geq 0} \max_{\mathbf{R}\in\mathcal{R}_m,\ \mathbf{x}\geq 0} L(\mathbf{R}, \mathbf{x}, \boldsymbol{\lambda}). \tag{56}$$

Since $\mathcal{R}_n \subseteq \mathcal{R}_m$, $V_{np} \leq V_{mp}$. The next result clarifies the relation among these four problems.

Theorem 8: $V_{sp} \leq V_{sd} = V_{mp} = V_{md}$.

According to Theorem 7, TCP/IP has an equilibrium exactly when there is no duality gap in the single-path utility maximization, i.e., when $V_{np} = V_{nd}$. Theorem 8 then says that in this case, there is no penalty in not splitting the traffic, i.e., single-path routing performs as well as multipath routing, $V_{np} = V_{mp}$. Multipath routing achieves a strictly higher utility V_{mp} precisely when TCP/IP has no equilibrium, in which case the TCP/IP iteration (51)–(53) cannot converge, let alone solve the single-path utility maximization problem (54) or (55). In this case the problem (54) and its dual (55) do not characterize TCP/IP, but their gap measures the loss in utility in restricting routing to single-path and is of independent interest.

Even though shortest-path routing is polynomial, the single-path utility maximization is NP-hard.

Theorem 9: The primal problem (54) is NP-hard.

Theorem 9 is proved [144] by reducing all instances of the integer partition problem to some instances of the primal problem (54). Theorem 8, however, implies that the subclass of the utility maximization problems with no duality gap are in *P*, since they are equivalent to multipath problems which are convex optimization problems and hence polynomial-time solvable. Informally, the hard problems are those with nonzero duality gap.

Theorem 7 suggests using pure prices $\boldsymbol{\lambda}(t)$ generated by TCP-AQM as link costs, because in this case, an equilibrium of TCP/IP, when it exists, maximizes aggregate utility over both rates and routes. It is shown in [144], however, that such an equilibrium can be unstable, and hence not attainable by TCP/IP.

Routing can be stabilized by including a strictly positive traffic-insensitive component in the link cost, in addition to congestion price. Stabilization, however, reduces the achievable utility. There thus seems to be an inevitable tradeoff between achievable utility and routing stability, when link costs are fixed. If the link capacities are optimally provisioned, however, pure *static* routing, which is necessarily stable, is enough to maximize utility. Moreover, it is optimal even within the class of multipath routing: again, there is no penalty in not splitting traffic across multiple paths.

Indeed, pure dynamic routing that uses only congestion prices as link cost was abandoned in APARNet precisely because of routing instability [3]. In practice, a weighted sum of congestion price and a traffic insensitive component is often used as link cost in shortest-path routing, i.e., (53) is replaced by

$$\mathbf{r}^s(t+1) = \arg\min_{\mathbf{r}^s\in\mathcal{H}^s} \sum_l (a\lambda_l(t) + b\tau_l) r_l^s \tag{57}$$

for some positive constant τ_l. We will interpret τ_l as the propagation delay over link l. The parameters a, b determine the responsiveness of routing to network traffic: the larger the ratio of a/b, the more responsive routing is. The result summarized above corresponds to pure dynamic routing $b = 0$ which is never used in practical networks. When $b > 0$, however, it can be shown that for any delay-insensitive utility function $U_s(x_s)$, there exists a network with sources using this utility function where TCP/IP equilibrium exists but does not solve the joint utility maximization problem (54) and its dual (55). It turns out that when $b > 0$, TCP/IP equilibrium, if it exists, maximizes a class of delay-sensitive utility functions and their dual [112].

Specifically, Theorems 7 and 8 generalize directly to the case with $a > 0$ and $b > 0$ when utility functions $U_s(x_s)$ in (51), (52), (53) are replaced by

$$U_s(x_s, \tau^s) := V_s(x_s) - \frac{b}{a} x_s \tau^s \tag{58}$$

where

$$\tau^s := \sum_l R_{ls}\tau_l$$

is the end-to-end propagation delay, and $V_s(x_s)$ is a strictly concave increasing and continuously differentiable function. This is an example of general delay-sensitive utility functions $U_s(x_s, \tau^s)$ where the utility of source s depends not only on its throughput x_s, but also on its end-to-end propagation delay τ^s. Note that τ^s is determined by routing. The particular class of utility functions in (58) has two distinct components: $V_s(x_s)$ which is strictly increasing in throughput x_s and $(b/a)x_s\tau^s$ which is strictly decreasing in delay. The weights a, b in the link cost in the routing decision translate directly into a weight in the utility function that determines how sensitive utility is to delay.

In [112], some counter-intuitive properties are also proved for any class of delay-sensitive utility functions optimized by TCP/IP with $a, b > 0$, as well as a sufficient condition for global stability of routing updates for general networks.

In [50], three alternative time-scale separations are further considered for the joint congestion control and shortest-path routing dynamics based on congestion price. Analytical characterizations and simulation experiments demonstrate how the step size of the congestion-control algorithm affects the stability of the system models, and how the time scale of each control loop and homogeneity of link capacities affect system stability and optimality. In particular, the stringent conditions on capacity configuration for TCP/IP interaction to remain stable suggest that congestion price, on its own, would be a poor "layering price" for TCP and (dynamic routing-based) IP.

In a different routing model capturing today's operational practice by service providers, [49] considers the following interaction between congestion control and traffic engineering. For a given routing configuration, the utilization of link l is $u_l = y_l/c_l$, where $y_l = \sum_s R_{ls}x_s$. To penalize routing configurations that congest the links, candidate routing solutions are evaluated based on an increasing, convex cost function $f(u_l)$ that increases steeply as u_l approaches 1. The following optimization problem over $\mathbf{R}$, for a fixed $\mathbf{x}$ and $\mathbf{c}$, captures the traffic-engineering practice:

$$\text{minimize} \quad \sum_l f\left(\sum_s R_{ls}x_s/c_l\right). \tag{59}$$

This optimization problem avoids solutions that operate near the capacity of the links and consequently tolerates temporary traffic bursts. The resulting routing configuration can, therefore, be considered robust. It is proved [49] that, for certain classes of cost function f, the interaction between end-user congestion control and the above traffic engineering (at the same time scale) converges for sufficiently concave utilities (i.e., sufficiently elastic traffic): $(\partial^2 U_s(x_s)/\partial x_s^2) \leq -\partial U_s(x_s)/x_s\partial x_s)$.

2) *Joint Congestion Control and Traffic Engineering:* Researchers have also carried out forward-engineering of joint congestion control and traffic engineering over multiple paths. Various designs have been presented based on somewhat different NUM formulations and decomposition methods: MATE in [34], TeXCP in [59], distributed adaptive traffic engineering (DATE) in [49], Overlay TCP in [46], and others [60], [69], [84], [101], [142].

For example, in the DATE algorithm [49], edge and core routers work together to balance load, limit the incoming traffic rate, and route around failures. The core routers compute prices based on local information and feed it back to the edge routers which adjust the end-to-end throughput on paths. Using the decomposition approach of "consistency pricing" (presented in Section IV-B), an algorithm is developed to update both congestion prices and consistency prices at core routers and feedback to edge routers for multipath load splitting. It is shown to be stochastically stable (more discussions in Section V-D) and converge to the joint and global optimum of the following NUM over *both* $\mathbf{R}$ and $\mathbf{x}$:

$$\begin{aligned} &\text{maximize} \quad \sum_s U_s(x_s) - \sum_l f\left(\sum_s R_{ls}x_s/c_l\right) \\ &\text{subject to} \quad \mathbf{Rx} \leq \mathbf{c}, \ \mathbf{x} \geq 0. \end{aligned} \tag{60}$$

Note that the objective function above favors a solution that provides both high aggregate utility to end-users and a low overall network congestion to the network operator, in order to satisfy the need for both performance (reflected through the utility function) and robustness (reflected through the cost function).

Other related works have studied different NUM formulations, e.g., without the linear capacity constraint or without the link congestion penalty term in the objective function in problem (60), using different distributed solution approaches. This is one of the cases where alternative decompositions naturally arise and lead to different implementation implications. More discussions on alternative vertical decompositions will appear in Section IV-C.

B. Case Study 2: Jointly Optimal Congestion Control and Physical Resource Allocation

Adaptive resource allocation per link, such as power control and error correction coding considered in this section, produces intriguing interactions with end-to-end congestion control.

1) *Power Control:* First consider a wireless multihop network with an established logical topology represented by $\mathbf{R}$ or equivalently $\{S(l)\}, \forall l$, where some nodes are sources of transmission and some nodes act as relay nodes. Revisiting the basic NUM (4), we observe that in an interference-limited wireless network, data rates attainable on wireless links are not fixed numbers $\mathbf{c}$ as in (4), and instead can be written as global and nonlinear functions of the transmit power vector $\mathbf{P}$ and channel conditions

$$c_l(\mathbf{P}) = \frac{1}{T}\log(1 + K\text{SIR}_l(\mathbf{P})), \quad \forall l.$$

Here constant T is the symbol period, which will be assumed to be one unit without loss of generality, and constant $K = (-\phi_1/\log(\phi_2\text{BER}))$, where ϕ_1 and ϕ_2 are constants depending on the modulation and BER is the required bit-error rate. The signal-to-interference ratio for link l is defined as $\text{SIR}_l(\mathbf{P}) = P_lG_{ll}/(\sum_{k\neq l} P_kG_{lk} + n_l)$ for a given set of path losses G_{lk} (from the transmitter on logical link k to the receiver on logical link l) and a given set of noises n_l (for the receiver on logical link l). The G_{lk} factors incorporate propagation loss, spreading gain, and other normalization constants. Notice that G_{ll} is the path gain on link l (from the transmitter on logical link l to the intended receiver on the same logical link). With reasonable spreading gain, G_{ll} is much larger than G_{lk}, $k \neq l$, and assuming that not too many close-by nodes transmit at the same time, KSIR is much larger than 1. In this high-SIR regime, c_l can be approximated as $\log(K\mathrm{SIR}_l(\mathbf{P}))$.

With the above assumptions, we have specified the following generalized NUM with "elastic" link capacities:

$$\begin{aligned} \text{maximize} \quad & \sum_s U_s(x_s) \\ \text{subject to} \quad & \sum_{s \in S(l)} x_s \leq c_l(\mathbf{P}), \quad \forall l \\ & \mathbf{x}, \mathbf{P} \geq 0 \end{aligned} \tag{61}$$

where the optimization variables are both source rates $\mathbf{x}$ and transmit powers $\mathbf{P}$. The key difference from the standard utility maximization (4) is that each link capacity c_l is now a function of the new optimization variables: the transmit powers $\mathbf{P}$. The design space is enlarged from $\mathbf{x}$ to both $\mathbf{x}$ and $\mathbf{P}$, which are clearly coupled in (61). Linear flow constraints on $\mathbf{x}$ become nonlinear constraints on $(\mathbf{x}, \mathbf{P})$. In practice, problem (61) is also constrained by the maximum and minimum transmit powers allowed at each transmitter on link $l : P_{l,\min} \leq P_l \leq P_{l,\max}$, $\forall l$.

The major challenges are the two global dependencies in (61).

- Source rates $\mathbf{x}$ and link capacities $\mathbf{c}$ are globally coupled across the network, as reflected in the range of summation $\{s \in S(l)\}$ in the constraints in (61).
- Each link capacity $c_l(\mathbf{P})$, in terms of the attainable data rate under a given power vector, is a global function of all the interfering powers.

We present the following distributive algorithm and later prove that it converges to the global optimum of (61). To make the algorithm and its analysis concrete, we focus on delay-based price and TCP Vegas window update (as reflected in items 1 and 2 in the algorithm, respectively), and the corresponding logarithmic utility maximization over $(\mathbf{x}, \mathbf{P})$, where α_s is a constant parameter in TCP Vegas (not as the α-fairness parameter here)

$$\begin{aligned} \text{maximize} \quad & \sum_s \alpha_s \log x_s \\ \text{subject to} \quad & \sum_{s \in S(l)} x_s \leq c_l(\mathbf{P}), \quad \forall l \\ & \mathbf{x}, \mathbf{P} \geq 0. \end{aligned} \tag{62}$$

Algorithm 2: **Joint Congestion Control and Power Control Algorithm**

During each time slot t, the following four updates are carried out simultaneously, until convergence:

1) At each intermediate node, a weighted queuing delay λ_l is implicitly updated, where $\beta_1 > 0$ is a constant

$$\lambda_l(t+1) = \left[\lambda_l(t) + \frac{\beta_1}{c_l(t)}\left(\sum_{s \in S(l)} x_s(t) - c_l(t)\right)\right]^+. \tag{63}$$

2) At each source, total delay D_s is measured and used to update the TCP window size w_s. Consequently, the source rate x_s is updated

$$w_s(t+1) = \begin{cases} w_s(t) + \frac{1}{D_s(t)}, & \text{if } \frac{w_s(t)}{d_s} - \frac{w_s(t)}{D_s(t)} < \alpha_s \\ w_s(t) - \frac{1}{D_s(t)}, & \text{if } \frac{w_s(t)}{d_s} - \frac{w_s(t)}{D_s(t)} > \alpha_s \\ w_s(t), & \text{else.} \end{cases}$$

$$x_s(t+1) = \frac{w_s(t+1)}{D_s(t)}. \tag{64}$$

3) Each transmitter j calculates a message $m_j(t) \in \mathbf{R}_+$[8] based on locally measurable quantities, and passes the message to all other transmitters by a flooding protocol

$$m_j(t) = \frac{\lambda_j(t)\mathrm{SIR}_j(t)}{P_j(t)G_{jj}}.$$

4) Each transmitter updates its power based on locally measurable quantities and the received messages, where $\beta_2 > 0$ is a constant

$$P_l(t+1) = P_l(t) + \frac{\beta_2 \lambda_l(t)}{P_l(t)} - \beta_2 \sum_{j \neq l} G_{lj} m_j(t). \tag{65}$$

With the minimum and maximum transmit power constraint $(P_{l,\min}, P_{l,\max})$ on each transmitter, the updated power is projected onto the interval $[P_{l,\min}, P_{l,\max}]$.

[8]Note that here m_j does not denote price-mapping functions as in Section II-A4.

Item 2 is simply the TCP Vegas window update [10]. Item 1 is a modified version of queuing delay price update [89] (and the original update [10] is an approximation of item 1). Items 3 and 4 describe a new power control using message passing. Taking in the current values of $\lambda_j(t)\mathrm{SIR}_j(t)/P_j(t)G_{jj}$ as the messages from other transmitters indexed by j, the transmitter on link l adjusts its power level in the next time slot in two ways: first increases power directly proportional to the current price and inversely proportional to the current power level, then decreases power by a weighted sum of the messages from all other transmitters, where the weights are the path losses G_{lj}.[9] Intuitively, if the local queuing delay is high, transmit power should increase, with a more moderate increase when the current power level is already high. If queuing delays on other links are high, transmit power should decrease in order to reduce interference on those links.

To compute m_j, the values of queuing delay λ_j, signal-interference-ratio SIR_j, and received power level P_jG_{jj} can be directly measured by node j locally. This algorithm only uses the resulting message m_j but *not* the individual values of λ_j, SIR_j, P_j and G_{jj}. Each message is a real number to be explicitly passed. To conduct the power update, G_{lj} factors are assumed to be estimated through training sequences.

It is important to note that there is no need to change the existing TCP congestion control and queue management algorithms. All that is needed to achieve the joint and global optimum of (62) is to utilize the values of weighted queuing delay in designing power control algorithm in the physical layer. The stability and optimality of this layering price can be stated through the following.

Theorem 10: For small enough constants β_1 and β_2, Algorithm 2 (63), (64), (65) converges to the global optimum of the joint congestion control and power control problem (62).

The key steps of this vertical decomposition, which uses congestion price as the layering price, are again through dual decomposition. We first associate a Lagrange multiplier λ_l for each of the constraints $\sum_{s\in S(l)} x_s \le c_l(\mathbf{P})$. Using the KKT optimality conditions [4], [9], solving problem (62) [or (61)] is equivalent to satisfying the complementary slackness condition and finding the stationary points of the Lagrangian.

Complementary slackness condition states that at optimality, the product of the dual variable and the associated primal constraint must be zero. This condition is satisfied since the equilibrium queuing delay must be zero if the total equilibrium ingress rate at a router is strictly smaller than the egress link capacity. We also need to find the stationary points of the Lagrangian: $L_{\text{system}}(\mathbf{x},\mathbf{P},\boldsymbol{\lambda}) = \sum_s U_s(x_s) - \sum_l \lambda_l \sum_{s\in S(l)} x_s + \sum_l \lambda_l c_l(\mathbf{P})$. By linearity of the differentiation operator, this can be decomposed into two separate maximization problems

$$\text{maximize}_{\mathbf{x}\succeq 0}\; L_{\text{congestion}}(\mathbf{x},\boldsymbol{\lambda}) = \sum_s U_s(x_s) - \sum_s \sum_{l\in L(s)} \lambda_l x_s$$

$$\text{maximize}_{\mathbf{P}\succeq 0}\; L_{\text{power}}(\mathbf{P},\boldsymbol{\lambda}) = \sum_l \lambda_l c_l(\mathbf{P}).$$

The first maximization is already implicitly solved by the congestion control mechanism for different U_s (e.g., TCP Vegas for $U_s(x_s) = \alpha_s \log x_s$). But we still need to solve the second maximization, using the Lagrange multipliers $\boldsymbol{\lambda}$ as the shadow prices to allocate exactly the right power to each transmitter, thus increasing the link data rates and reducing congestion at the network bottlenecks. Although the data rate on each wireless link is a global function of all the transmit powers, distributed solution is still feasible through distributed gradient method with the help of message passing. Issues arising in practical implementation, such as asynchronous update and reduced message passing, and their impacts on convergence and optimality, are discussed in [18].

Method 11: Dual Decomposition for Jointly Optimal Cross-Layer Design.

The logical topology and routes for four multihop connections are shown in Fig. 10 for a numerical example. The path losses G_{ij} are determined by the relative physical distances d_{ij}, which we vary in different experiments, by $G_{ij} = d_{ij}^{-4}$. The target BER is 10^{-3} on each logical link.

Transmit powers, as regulated by the proposed distributed power control, and source rates, as regulated through TCP Vegas window update, are shown in Fig. 11. The initial conditions of the graphs are based on the equilibrium states of TCP Vegas with fixed power levels of 2.5 mW. With power control, the transmit powers $\mathbf{P}$ distributively adapt to induce a "smart" capacity $\mathbf{c}$ and queuing delay $\boldsymbol{\lambda}$ configuration in the network, which in turn lead to increases in end-to-end throughput as indicated by the rise in all the allowed source rates. Notice that some link capacities actually decrease while the

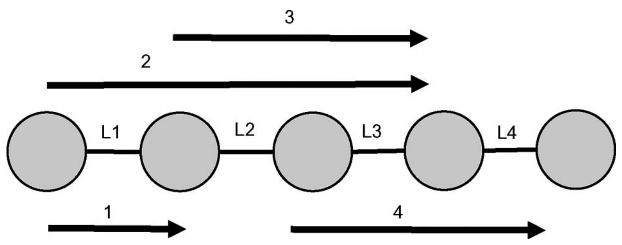


Fig. 10. ***Logical topology and connections for a numerical example of joint congestion control power control.***

[9]This facilitates a graceful reduction of message passing scope since messages from far-away neighbors are weighted much less.

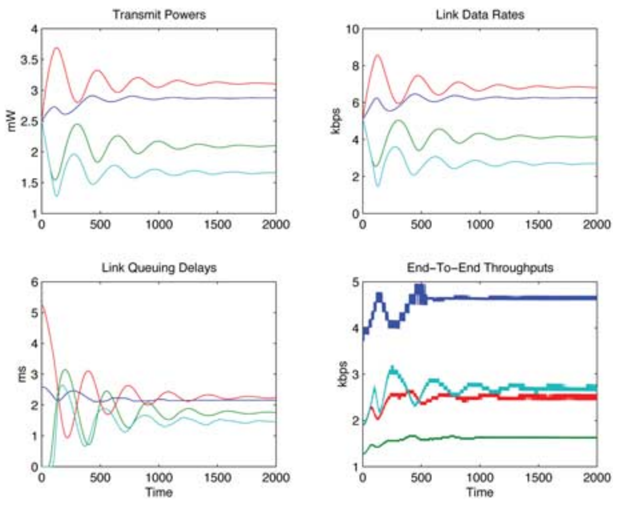


Fig. 11. ***Typical numerical example of joint TCP Vegas congestion control and power control. The top left graph shows the primal variables lP. The lower left graph shows the dual variables λ. The lower right graph shows the primal variables x, i.e., the end-to-end throughput. In order of their y-axis values after convergence, the curves in the top left, top right, and bottom left graphs are indexed by the third, first, second, and fourth links in Fig. 10. The curves in the bottom right graph are indexed by flows 1, 4, 3, 2.***

capacities on the bottleneck links rise to maximize the total network utility. This is achieved through a distributive adaptation of power, which lowers the power levels that cause the most interference on the links that are becoming a bottleneck in the dynamic demand-supply matching process. Confirming our intuition, such a "smart" allocation of power tends to reduce the spread of queuing delays, thus preventing any link from becoming a bottleneck. Queuing delays on the four links do not become the same though, due to the asymmetry in traffic load on the links and different weights in the logarithmic utility objective functions.

2) Adaptive Coding: In the second half of this section, we discuss the interaction of per-hop adaptive channel coding with end-to-end congestion control. At the end hosts, the utility for each user depends on both transmission rate and signal quality, with an intrinsic tradeoff between the two. At the same time, each link may also provide a "fatter" (or "thinner") transmission "pipe" by allowing a higher (or lower) decoding error probability.

In the basic NUM, the convexity and decomposability properties of the optimization problem readily lead to a distributed algorithm that converges to the globally optimal rate allocation. The generalized NUM problems for joint rate-reliability provisioning turn out to be nonseparable and nonconvex. We review a price-based distributed algorithm, and its convergence to the globally optimal rate-reliability tradeoff under readily verifiable sufficient conditions on link coding block lengths and user utility curvatures. In contrast to standard price-based rate control algorithms for the basic NUM, in which each link provides the same congestion price to each of its users and each user provides its willingness to pay for rate allocation to the network, in the joint rate-reliability algorithms each link provides a possibly different congestion price to each of its users and each user also provides its willingness to pay for its own reliability to the network.

On some communication links, physical layer's adaptive channel coding (i.e., error correction coding) can change the information "pipe" sizes and decoding error probabilities, e.g., through adaptive channel coding in DSL broadband access networks or adaptive diversity-multiplexing control in MIMO wireless systems. Then each link capacity is a function of the signal quality (i.e., decoding reliability) attained on that link. A higher throughput can be obtained on a link at the expense of lower decoding reliability, which in turn lowers the end-to-end signal quality for sources traversing the link and reduces users' utilities. This leads to an intrinsic tradeoff between rate and reliability. This tradeoff also provides an additional degree of freedom for improving each user's utility as well as system efficiency. For example, if we allow lower decoding reliability, thus higher information capacity, on the more congested links, and higher decoding reliability, thus lower information capacity, on the less congested links, we may improve the end-to-end rate and reliability performance of each user. Clearly, rate-reliability tradeoff is globally coupled across the links and users.

In the case where the rate-reliability tradeoff is controlled through the code rate of each source on each link, there are two possible policies: integrated dynamic reliability policy and differentiated dynamic reliability policy. In integrated policy, a link provides the same error probability (i.e., the same code rate) to each of the sources traversing it. Since a link provides the same code rate to each of its sources, it must provide the lowest code rate that satisfies the requirement of the source with the highest reliability. This motivates a more general approach called differentiated policy to fully exploit the rate-reliability tradeoff when there exist multiclass sources (i.e., sources with different reliability requirements) in the network. Under the differentiated dynamic reliability policy, a link can provide a different error probability (i.e., a different code rate) to each of the sources using this link.

We assume that each source s has a utility function $U_s(x_s, \rho_s)$, where x_s is an information data rate and ρ_s is reliability of source s. We assume that the utility function is a continuous, increasing, and strictly concave function of x_s and ρ_s. Each source s has a minimum reliability requirement $\rho_s^{\min}$. The reliability of source s is defined as

$$\rho_s = 1 - p^s$$

where p^s is the end-to-end error probability of source s. Each link l has its maximum transmission capacity $c_l^{\max}$.

After link l receives the data of source s from the upstream link, it first decodes it to extract the information data of the source and encodes it again with its own code rate, $r_{l,s}$, where the code rate is defined by the ratio of the information data rate x_s at the input of the encoder to the transmission data rate $t_{l,s}$ at the output of the encoder. This allows a link to adjust the transmission rate and the error probability of the sources, since the transmission rate of source s at link l can be defined as

$$t_{l,s} = \frac{x_s}{r_{l,s}}$$

and the error probability of source s at link l can be defined as a function of $r_{l,s}$ by

$$p_{l,s} = E_l(r_{l,s})$$

which is assumed to be an increasing function of $r_{l,s}$. Rarely are there analytic formulas for $E_l(r_{l,s})$, and we will use various upper bounds on this function. The end-to-end error probability for each source s is

$$p^s = 1 - \prod_{l\in L(s)} (1 - p_{l,s}) = 1 - \prod_{l\in L(s)} \left(1 - E_l(r_{l,s})\right).$$

Assuming that the error probability of each link is small (i.e., $p_{l,s} \ll 1$), we can approximate the end-to-end error probability of source s as

$$p^s \approx \sum_{l\in L(s)} p_{l,s} = \sum_{l\in L(s)} E_l(r_{l,s}).$$

Hence, the reliability of source s can be expressed as

$$\rho_s \approx 1 - \sum_{l\in L(s)} E_l(r_{l,s}).$$

Since each link l has a maximum transmission capacity $C_l^{\max}$, the sum of transmission rates of sources that are traversing each link cannot exceed $C_l^{\max}$

$$\sum_{s\in S(l)} t_{l,s} = \sum_{s\in S(l)} \frac{x_s}{r_{l,s}} \le C_l^{\max}, \quad \forall l.$$

For (the more general) differentiated dynamic reliability policy, in which a link may provide a different code rate to each of the sources traversing it, the associated generalized NUM becomes the following problem with variables $\mathbf{x}$, $\boldsymbol{\rho}$, $\mathbf{r}$:

$$\begin{aligned} \text{maximize} \quad & \sum_s U_s(x_s, \rho_s) \\ \text{subject to} \quad & \rho_s \le 1 - \sum_{l\in L(s)} E_l(r_{l,s}), \quad \forall s \\ & \sum_{s\in S(l)} \frac{x_s}{r_{l,s}} \le C_l^{\max}, \quad \forall l \\ & \rho_s^{\min} \le \rho_s \le 1, \quad \forall s \\ & 0 \le r_{l,s} \le 1, \quad \forall l,\ s \in S(l). \end{aligned} \tag{66}$$

There are two main difficulties in distributively and globally solving the above problem. The first one is the convexity of $E_l(r_{l,s})$. If random coding based on binary coded signals is used, a standard upper bound on the error probability is

$$p_l < \frac{1}{2} 2^{-M(R_0 - r_l)}$$

where M is the block length and R_0 is the cutoff rate. In this case, $E_l(r_l) = (1/2)2^{-M(R_0 - r_l)}$ is a convex function for given M and R_0. A more general approach for discrete memoryless channel models is to use the random code ensemble error exponent that upper bounds the decoding error probability

$$p_l \le \exp(-ME_r(r_l))$$

where M is the codeword block length and $E_r(r_l)$ is the random coding exponent function, which is defined as

$$E_r(r_l) = \max_{0\le\mu\le1} \max_{\mathbf{Q}} [E_o(\mu, \mathbf{Q}) - \mu r_l]$$

where

$$E_o(\mu, \mathbf{Q}) = -\log \sum_{j=0}^{J-1} \left[\sum_{k=0}^{K-1} Q_k \bar{P}_{jk}^{1/(1+\mu)} \right]^{1+\mu}$$

K is the size of input alphabet, J is the size of output alphabet, Q_k is the probability that input letter k is chosen, and $\bar{P}_{jk}$ is the probability that output letter j is received given that input letter k is transmitted.

In general, $E_l(r_l) = \exp(-ME_r(r_l))$ may not be convex (even though it is known that $E_r(r_l)$ is a convex function). However, the following lemma provides a sufficient condition for its convexity.

Lemma 1: If the absolute value of the first derivatives of $E_r(r_l)$ is bounded away from 0 and absolute value of the second derivative of $E_r(r_l)$ is upper bounded, then for a large enough codeword block length M, $E_l(r_l)$ is a convex function.

Method 12: Computing Conditions Under Which a General Constraint Set is Convex.

The second difficulty is the global coupling of constraints $\sum_{s\in S(l)}(x_s/r_{l,s}) \le C_l^{\max}$. This problem is tackled by first introducing auxiliary variables $c_{l,s}$, which can be interpreted as the allocated transmission capacity to source s at link l

$$\begin{aligned}
\text{maximize} \quad & \sum_s U_s(x_s,\rho_s) \\
\text{subject to} \quad & \rho_s \le 1 - \sum_{l\in L(s)} E_l(r_{l,s}), \quad \forall s \\
& \frac{x_s}{r_{l,s}} \le c_{l,s}, \quad \forall l,\ s\in S(l) \\
& \sum_{s\in S(l)} c_{l,s} \le C_l^{\max}, \quad \forall l \\
& \rho_s^{\min} \le \rho_s \le 1, \quad \forall s \\
& 0 \le r_{l,s} \le 1, \quad \forall l,\ s\in S(l) \\
& 0 \le c_{l,s} \le C_l^{\max}, \quad \forall l,\ s\in S(l). \qquad (67)
\end{aligned}$$

Note that effectively a new "scheduling layer" has been introduced into the problem: scheduling of flows by deciding bandwidth sharing on each link $\{c_{l,s}\}$.

Method 13: Introducing a New Layer to Decouple a Generalized NUM.

A log change of variables $x'_s = \log x_s$ can be used to decouple the above problem for horizontal decomposition. Define a modified utility function $U'_s(x'_s,\rho_s) = U_s(e^{x'_s},\rho_s)$, which needs to be concave in order for the transformed problem to remain a convex optimization problem, similar to the curvature condition on utility function in Section II-B2.

Define

$$\begin{aligned}
g_s(x_s,\rho_s) &= \frac{\partial^2 U_s(x_s,\rho_s)}{\partial x_s^2}x_s + \frac{\partial U_s(x_s)}{\partial x_s} \\
h_s(x_s,\rho_s) &= \left(\left(\frac{\partial^2 U_s(x_s,\rho_s)}{\partial x_s \partial \rho_s}\right)^2 - \frac{\partial^2 U_s(x_s,\rho_s)}{\partial x_s^2}\frac{\partial^2 U_s(x_s,\rho_s)}{\partial \rho_s^2}\right)x_s \\
&\quad - \frac{\partial^2 U_s(x_s,\rho_s)}{\partial \rho_s^2}\frac{\partial U_s(x_s,\rho_s)}{\partial x_s} \\
q_s(x_s,\rho_s) &= \frac{\partial^2 U_s(x_s,\rho_s)}{\partial \rho_s^2}.
\end{aligned}$$

Lemma 2: If $g_s(x_s,\rho_s) < 0$, $h_s(x_s,\rho_s) < 0$, and $q_s(x_s,\rho_s) < 0$, then $U'_s(x'_s,\rho_s)$ is a concave function of x'_s and ρ_s.

Now the joint rate-reliability problem (66) can be solved distributively through dual decomposition.

Algorithm 3: **Differentiated Dynamic Reliability Policy Algorithm**

In each iteration t, by solving (68) over (x'_s,ρ_s), each source s determines its information data rate and requested reliability (i.e., $x'_s(t)$ or equivalently, $x_s(t) = e^{x'_s(t)}$, and $\rho_s(t)$) that maximize its net utility based on the prices in the current iteration. Furthermore, by price update (69), the source adjusts its offered price per unit reliability for the next iteration.

Source problem and reliability price update at source s:

- Source problem

$$\begin{aligned}
\text{maximize} \quad & U_s(x'_s,\rho_s) - \lambda^s(t)x'_s - \mu_s(t)\rho_s \\
\text{subject to} \quad & \rho_s^{\min} \le \rho_s \le 1 \qquad (68)
\end{aligned}$$

  where $\lambda^s(t) = \sum_{l\in L(s)} \lambda_{l,s}(t)$ is the end-to-end congestion price at iteration t.
- Price update (where step size can be set to $\beta(t) = \beta_0/t$ for some $\beta_0 > 0$)[10]

$$\mu_s(t+1) = [\mu_s(t) - \beta(t)(\rho^s(t) - \rho_s(t))]^+ \qquad (69)$$

  where $\rho^s(t) = 1 - \sum_{l\in L(s)} E_l(r_{l,s}(t))$ is the end-to-end reliability at iteration t.

[10]Diminishing stepsizes, e.g., $\beta(t) = \beta_0/t$, can guarantee convergence when the primal optimization problem's objective function is concave but not strictly concave in all the variables, whereas a constant stepsize cannot.

Concurrently in each iteration t, by solving problem (70) over $(c_{l,s}, r_{l,s})$, $\forall s \in S(l)$, each link l determines the allocated transmission capacity $c_{l,s}(t)$ and the code rate $r_{l,s}(t)$ of each of the sources using the link, so as to maximize the "net revenue" of the network based on the prices in the current iteration. In addition, by price update (71), the link adjusts its congestion price per unit rate for source s during the next iteration.

Link problem and congestion price update at link l:

- Link problem:

$$\begin{aligned} \text{maximize} \quad & \sum_{s\in S(l)} \lambda_{l,s}(t)(\log c_{l,s} + \log r_{l,s}) \\ & \quad - \mu_s(t)E_l(r_{l,s}) \sum_{s\in S(l)} c_{l,s} \le C_l^{\max} \\ \text{subject to} \quad & 0 \le c_{l,s} \le C_l^{\max}, \quad s \in S(l) \\ & 0 \le r_{l,s} \le 1, \quad s \in S(l). \end{aligned} \tag{70}$$

- Price update (where step size can be set to $\beta(t) = \beta_0/t$ for some $\beta_0 > 0$)

$$\begin{aligned} \lambda_{l,s}(t+1) &= \left[\lambda_{l,s}(t) - \beta(t)\left(\log c_{l,s}(t) + \log r_{l,s}(t) - x_s'(t)\right)\right]^+ \\ &= \left[\lambda_{l,s}(t) - \beta(t)\left(\log c_{l,s}(t) + \log r_{l,s}(t) - \log x_s(t)\right)\right]^+ \\ & \quad s \in S(l). \end{aligned} \tag{71}$$

In the above algorithm, to solve problem (68), source s needs to know $\lambda^s(t)$, the sum of congestion prices $\lambda_{l,s}(t)$'s of links that are along its path $L(s)$. This can be obtained by the notification from the links, e.g., through acknowledgment packets. To carry out price update (69), the source needs to know the sum of error probabilities of the links that are along its path (i.e., its own reliability that is provided by the network, $\rho^s(t)$). This can be obtained by the notification either from the links that determine the code rate for the source [by solving problem (70)] or from the destination that can measure its end-to-end reliability. To solve the link problem (70), each link l needs to know $\mu_s(t)$ from each of sources using this link l. This can be obtained by the notification from these sources. To carry out price update (71), the link needs to know the information data rate of each of the sources that are using it (i.e., $x_s(t)$). This can be measured by the link itself.

Method 14: End User Generated Pricing for Distributed Update of Metrics in User Utility Function.

Theorem 11: For sufficiently concave utilities and sufficiently strong codes, and diminishing stepsizes, the dual variables $\boldsymbol{\lambda}(t)$ and $\boldsymbol{\mu}(t)$ converge to the optimal dual solutions $\boldsymbol{\lambda}^*$ and $\boldsymbol{\mu}^*$ and the corresponding primal variables $\mathbf{x}'^*$, $\boldsymbol{\rho}^*$, $\mathbf{c}^*$, and $\mathbf{r}^*$ are the globally optimal primal solutions of the joint rate-reliability problem, i.e., $\mathbf{x}^* = (e^{x_s'^*})_{\forall s}$, $\boldsymbol{\rho}^*$, $\mathbf{c}^*$, and $\mathbf{r}^*$ are the globally optimal primal solutions of problem (66).

We now present numerical examples for the proposed algorithms by considering a simple network, shown in Fig. 12, with a linear topology consisting of four links and eight users. Utility function for user s is $U_s(x_s, \rho_s)$ in the following α-fair form, shifted such that $U_s(x_s^{\min}, \rho_s^{\min}) = 0$ and $U_s(x_s^{\max}, \rho_s^{\max}) = 1$ (where $x_s^{\min}$, $\rho_s^{\min}$, $x_s^{\max}$, $\rho_s^{\max}$ are constants), and with utility on rate and utility on reliability summed up with a given weight θ_s between rate and reliability utilities

$$U_s(x_s, \rho_s) = \theta_s \frac{x_s^{1-\alpha} - x_s^{\min(1-\alpha)}}{x_s^{\max(1-\alpha)} - x_s^{\min(1-\alpha)}} + (1-\theta_s)\frac{\rho_s^{(1-\alpha)} - \rho_s^{\min(1-\alpha)}}{\rho_s^{\max(1-\alpha)} - \rho_s^{\min(1-\alpha)}}.$$

Different weights θ_s are given to the eight users as follows:

$$\theta_s = \begin{cases} 0.5 - v, & \text{if } s \text{ is an odd number} \\ 0.5 + v, & \text{if } s \text{ is an even number} \end{cases} \tag{72}$$

and vary v from 0 to 0.5 in step size of 0.05.

The decoding error probability on each link l is assumed to be of the following form, with constant M:

$$p_l = \frac{1}{2}\exp(-M(1 - r_l)).$$

We trace the globally optimal tradeoff curve between rate and reliability using differentiated and integrated dynamic reliability policies, and compare the network utility achieved by the following three schemes:

- Static reliability: each link provides a fixed error probability 0.025. Only rate control is performed to maximize the network utility.
- Integrated dynamic reliability: each link provides the same adjustable error probability to all its users.

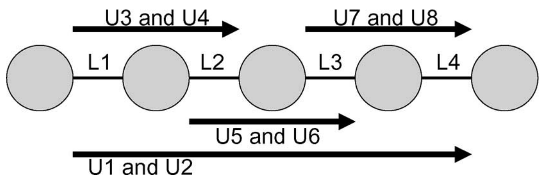


Fig. 12. ***Network topology and flow routes for a numerical example of rate-reliability tradeoff.***

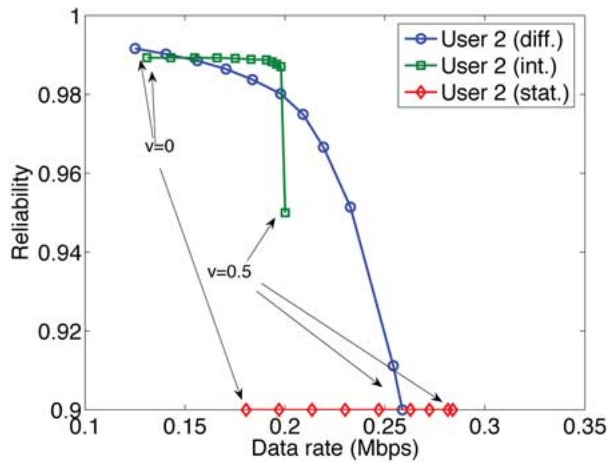


Fig. 13. ***Comparison of data rate and reliability tradeoff in a numerical example by each policy for User 2, when θ_s are changed according to (72).***

- Differentiated dynamic reliability: each link provides a possibly different error probability to each of its users.

Fig. 13 shows the globally optimal tradeoff curves between rate and reliability for a particular user, under the three policies of static reliability, integrated dynamic reliability, and differentiated dynamic reliability, respectively. The differentiated scheme shows a much larger dynamic range of tradeoff than both the integrated and static schemes. The gain in total network utility through joint rate and reliability control is shown in Fig. 14.

C. Case Study 3: Jointly Optimal Congestion and Contention Control

Following the notation in Section II-B, for joint end-to-end rate allocation and per-hop MAC, the generalized NUM problem for random-access-based MAC and TCP can be formulated as the following optimization over $(\mathbf{x}, \mathbf{P}, \mathbf{p})$:

$$\begin{aligned} \text{maximize} \quad & \sum_s U_s(x_s) \\ \text{subject to} \quad & \sum_{s \in S(l)} x_s \le c_l p_l \prod_{k \in N^I_{\text{to}}(l)} (1 - P^k), \quad \forall l \\ & \sum_{l \in L_{\text{out}}(n)} p_l = P^n, \quad \forall n \\ & x_s^{\min} \le x_s \le x_s^{\max}, \quad \forall s \\ & 0 \le P^n \le 1, \quad \forall n \\ & 0 \le p_l \le 1, \quad \forall l. \end{aligned} \tag{73}$$

Similar to the discussions on MAC forward-engineering and jointly optimal rate reliability control, for sufficiently concave utilities, problem (73) is a convex optimization after a log change of variables $\{p_l, P^k\}$. Its solution can now be distributively carried out using either the penalty function approach or the dual-decomposition-based Lagrangian relaxation approach. Both have standard convergence properties but now producing different implications to the *time scale* of TCP/MAC interaction (e.g., [74], [143], [169]), as shown in the rest of this section.

First the penalty function approach is pursued. We first define $h_l(\mathbf{p}, \mathbf{x}') = \log(\sum_{s \in S(l)} e^{x'_s}) - c'_l - \log p_l - \sum_{k \in N^I_{\text{to}}(l)} \log(1 - \sum_{m \in L_{\text{out}}(k)} p_m)$ and $w_n(\mathbf{p}) = \sum_{m \in L_{\text{out}}(n)} p_m - 1$. Then, problem (73) can be rewritten as

$$\begin{aligned} \text{maximize} \quad & \sum_s U'_s(x'_s) \\ \text{subject to} \quad & h_l(\mathbf{p}, \mathbf{x}') \le 0, \quad \forall l \\ & w_n(\mathbf{p}) \le 0, \quad \forall n \\ & x'^{\min}_s \le x'_s \le x'^{\max}_s, \quad \forall s \\ & 0 \le p_l \le 1, \quad \forall l. \end{aligned} \tag{74}$$

Instead of solving problem (74) directly, we apply the penalty function method and consider the following problem:

$$\begin{aligned} \text{maximize} \quad & V(\mathbf{p}, \mathbf{x}') \\ \text{subject to} \quad & x'^{\min}_s \le x'_s \le x'^{\max}_s, \quad \forall s \\ & 0 \le p_l \le 1, \quad \forall l \end{aligned} \tag{75}$$

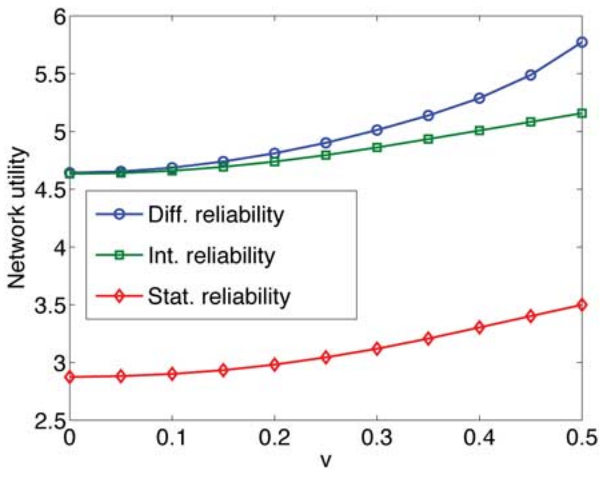


Fig. 14. ***Comparison of the achieved network utility attained in a numerical example by the differentiated dynamic policy, the integrated dynamic policy, and the static policy, when θ_s are changed according to (72).***

where $V(\mathbf{p},\mathbf{x}') = \sum_s U'_s(x'_s) - \kappa \sum_l \max\{0, h_l(\mathbf{p},\mathbf{x}')\} - \kappa \sum_n \max\{0, w_n(\mathbf{p})\}$ and κ is a positive constant.

Since the objective function of problem (75) is concave, problem (75) is convex optimization with simple, decoupled constraints, which can be solved by using a subgradient projection algorithm. We can easily show that

$$\frac{\partial V(\mathbf{p},\mathbf{x}')}{\partial p_l} = \kappa \left(\frac{\epsilon_l}{p_l} - \frac{\sum_{k \in L^I_{\text{from}}(t_l)} \epsilon_k}{1 - \sum_{m \in L_{\text{out}}(t_l)} p_m} - \delta_{t_l} \right) \tag{76}$$

and

$$\frac{\partial V(\mathbf{p},\mathbf{x}')}{\partial x'_s} = \frac{\partial U'_s(x'_s)}{\partial x'_s} - \kappa e^{x'_s} \sum_{l \in L(s)} \frac{\epsilon_l}{\sum_{k \in S(l)} e^{x'_k}} \tag{77}$$

where

$$\epsilon_l = \begin{cases} 0, & \text{if } \sum_{n \in S(l)} e^{x'_n} \le c_l p_l \prod_{k \in N^I_{\text{to}}(l)} \left(1 - \sum_{m \in L_{\text{out}}(k)} p_m\right) \\ 1, & \text{otherwise} \end{cases}$$

and

$$\delta_n = \begin{cases} 0, & \text{if } \sum_{m \in L_{\text{out}}(n)} p_m \le 1 \\ 1, & \text{otherwise.} \end{cases}$$

Then, an iterative subgradient projection algorithm, with iterations indexed by t, that solves problem (75) is obtained as follows.

Algorithm 4: **Joint End-to-End Congestion Control and Local Contention Control Algorithm**

On each logical link, transmission is decided to take place with persistence probability

$$p_l(t+1) = \left[p(t) + \alpha(t) \frac{\partial V(\mathbf{p},\mathbf{x}')}{\partial p_l} \Big|_{\mathbf{p}=\mathbf{p}(t),\mathbf{x}'=\mathbf{x}'(t)} \right]_0^1, \quad \forall l \tag{78}$$

and concurrently at each source, the end-to-end rate is adjusted

$$x'_s(t+1) = \left[x'_s(t) + \alpha(t) \frac{\partial V(\mathbf{p},\mathbf{x}')}{\partial x'_s} \Big|_{\mathbf{p}=\mathbf{p}(t),\mathbf{x}'=\mathbf{x}'(t)} \right]_{x'^{\min}_s}^{x'^{\max}_s}, \quad \forall s \tag{79}$$

where $[a]_c^b = \max\{\min\{a,b\},c\}$.

The joint control algorithm (78) and (79) can be implemented as follows. Each link l (or its transmission node t_l) updates its persistence probability $p_l(t)$ using (78), and concurrently, each source updates its data rate $x_s(t)$ using (79). To calculate the subgradient in (76), each link needs information only from link k, $k \in L^I_{\text{from}}(t_l)$, i.e., from links whose transmissions are interfered from the transmission of link l, and those links are in the neighborhood of link l. To calculate the subgradient in (77), each source needs information only from link l, $l \in L(s)$, i.e., from links on its routing path. Hence, to perform the algorithm, each source and link need only local information through limited message passing and the algorithm can be implemented in a distributed way. In particular, note that δ_n is calculated at the transmitter node of each link to update the persistence probability of that link, and does not need to be passed among the nodes. There is no need to explicitly pass around the values of persistence probabilities, since their effects are included in $\{\epsilon_l\}$. Quantities such as $\sum_{m \in L_{\text{out}}(t_l)} p_m$ and $\sum_{k \in S(l)} \exp(x'_k)$ can be measured locally by each node and each link.

To implement a dual-decomposition-based algorithm instead, we can decompose problem (73) into two problems, using a standard technique of dual decomposition also used in [16] and [143]

$$\begin{aligned} &\text{maximize} && \sum_s U_s(x_s) \\ &\text{subject to} && \sum_{s \in S(l)} x_s \le y_l, \quad \forall l \\ &&& x_s^{\min} \le x_s \le x_s^{\max}, \quad \forall s \end{aligned} \tag{80}$$

where y_l is the average data rate of link l, and

$$\begin{aligned} &\text{maximize} && \hat{U}(\mathbf{p}) \\ &\text{subject to} && \sum_{m \in L_{\text{out}}(n)} p_m \le 1, \quad \forall n \\ &&& 0 \le p_l \le 1, \quad \forall l \end{aligned} \tag{81}$$

where

$$\hat{U}(\mathbf{p}) = \max\left\{\sum_s U_s(x_s) \middle| \sum_{s\in S(l)} x_s \le y_l(\mathbf{p}),\quad \forall l,\right.$$
$$y_l(\mathbf{p}) = c_l p_l \prod_{k\in N_{\text{to}}^I(l)} \left(1 - \sum_{m\in L_{\text{out}}(k)} p_m\right),\quad \forall l$$
$$\left. x_s^{\min} \le x_s \le x_s^{\max},\quad \forall s\right\}.$$

For a given $\mathbf{y}$, problem (80) can be solved by dual decomposition and distributed subgradient method just as before.

We now solve problem (81). To this end, we first add a penalty function to the objective function of the problem as

$$\begin{aligned} &\text{maximize} && \hat{V}(\mathbf{p}) \\ &\text{subject to} && 0 \le p_l \le 1,\quad \forall l \end{aligned} \tag{82}$$

where $\hat{V}(\mathbf{p}) = \hat{U}(\mathbf{p}) - \kappa \max\{0, \sum_n (1 - \sum_{m\in L_{\text{out}}(n)} p_m)\}$ and κ is a positive constant. Since problem (82) is a convex problem with simple constraints, we can solve it by using a subgradient projection algorithm as

$$p_l(t+1) = \left[p(t) + \beta(t) \frac{\partial \hat{V}(\mathbf{p})}{\partial p_l}\bigg|_{\mathbf{p}=\mathbf{p}(t)} \right]_0^1,\quad \forall l \tag{83}$$

where $\partial \hat{V}(\mathbf{p})/\partial p_l$ is a subgradient of $\hat{V}(\mathbf{p})$ with respect to p_l. It can be readily verified that $\partial \hat{V}(\mathbf{p})/\partial p_l$ is obtained as

$$\begin{aligned} \frac{\partial \hat{V}(\mathbf{p})}{\partial p_l} =& \lambda_l^*(t) c_l \prod_{k\in N_{\text{to}}^I(l)} \left(1 - \sum_{m\in L_{\text{out}}(k)} p_m\right) \\ &- \sum_{n\in L_{\text{from}}^I(t_l)} \lambda_n^*(t) c_n p_n \\ &\times \prod_{k\in N_{\text{to}}^I(n), k\ne t_l} \left(1 - \sum_{m\in L_{\text{out}}(k)} p_m\right) - \kappa\delta_{t_l} \end{aligned} \tag{84}$$

where

$$\delta_n = \begin{cases} 0, & \text{if } \sum_{m\in L_{\text{out}}(n)} p_m \le 1 \\ 1, & \text{otherwise} \end{cases}$$

and $\boldsymbol{\lambda}^*(t)$ is the optimal dual solution to dual problem of (80) with $\mathbf{y} = \mathbf{y}(\mathbf{p}(t))$.

This dual-decomposition-based algorithm can also be implemented in a distributed way. In each time slot, each link determines its persistence probability by solving (83) with the help of local message passing to obtain the expression in (19). Then, within the time slot, based on $\mathbf{y}(\mathbf{p}(t))$, each source and link use standard dual-decomposition-based algorithm to solve (80) and determine the data rate of each source in the time slot.

Unlike the penalty-function-based algorithm, this dual-decomposition-based algorithm clearly decomposes TCP and MAC layers through the vertical decomposition (80) and (81). However, it needs an embedded loop of iterations [i.e., the convergence of a distributed subgradient algorithm to solve (80) in each time slot]. Hence, it may require longer convergence time than the penalty-function-based algorithm. This comparison between two alternative decompositions, together with yet another decomposition in [169] for the same NUM formulation, highlights the engineering implication of different decompositions to the time scale of the interaction between functional modules.

Method 15: Providing Different Timescales of Protocol Stack Interactions Through Different Decomposition Methods.

D. Case Study 4: Jointly Optimal Congestion Control, Routing, and Scheduling

A generalized NUM is formulated in [17], [35], [36], [82], [83], [99], and [129] where the key additional feature is the optimization over not just source rates but also scheduling of medium access and the incorporation of scheduling constraint. The standard dual decomposition decomposes it vertically into subproblems that can be solved through congestion control, routing, and scheduling.

Consider an *ad hoc* wireless network with a set N of nodes and a set L of logical links. We assume some form of power control so that each logical link l has a fixed capacity c_l when it is active. The feasible rate region at the link layer is the convex hull of the corresponding rate vectors of independent sets of the conflict graph. Let $\boldsymbol{\Pi}$ denote the feasible rate region. Let x_i^k be the flow rate generated at node i for destination k. We assume there is a queue for each destination k at each link (i,j). Let f_{ij}^k be the amount of capacity of link (i,j) allocated to the flows on that link for final destination k. Consider the following generalized NUM in variables $x_s \ge 0$, $f_{ij}^k \ge 0$:

$$\begin{aligned} &\text{maximize} && \sum_s U_s(x_s) \\ &\text{subject to} && x_i^k \le \sum_{j:(i,j)\in L} f_{ij}^k - \sum_{j:(j,i)\in L} f_{ji}^k,\quad \forall i,k \\ &&& \mathbf{f} \in \boldsymbol{\Pi} \end{aligned} \tag{85}$$

where x_s is a shorthand for x_i^k. The first constraint is a flow balance equation: the flow originated from node i for final destination k plus total capacity allocated for transit flows through node i for final destination k should be no more than the total capacity going out of node i for final destination k. The second constraint is on schedulability. The dual problem of (85) decomposes into minimizing the sum of the resulting values of the following two subproblems:

$$D_1(\boldsymbol{\lambda}) := \max_{x_s \geq 0} \sum_s (U_s(x_s) - x_s \lambda_s) \tag{86}$$

$$D_2(\boldsymbol{\lambda}) := \max_{f_{ij}^k \geq 0} \sum_{i,k} \lambda_i^k \sum_j \left(f_{ij}^k - f_{ji}^k\right) \quad \text{subject to} \quad \mathbf{f} \in \boldsymbol{\Pi}. \tag{87}$$

The first subproblem is congestion control where λ_s is the congestion price locally at source $s = (i, k)$. The second subproblem corresponds to a joint problem of multipath routing and allocation of link capacities. Thus, by dual decomposition, the flow optimization problem decomposes into separate local optimization problems that interact through congestion prices.

The congestion control problem (86) admits a unique maximizer $x_s(\boldsymbol{\lambda}) = U_s'^{-1}(\lambda_s)$. The joint routing and scheduling problem (87) is equivalent to

$$\sum_{i,j} \sum_k \max_{f_{ij}^k \geq 0} f_{ij}^k \left(\lambda_i^k - \lambda_j^k\right) \quad \text{subject to} \quad \mathbf{f} \in \boldsymbol{\Pi}.$$

Hence, an optimal schedule is to have $f_{ij}^k = c_{ij}$, if k maximizes $(\lambda_i^k - \lambda_j^k)$ and 0, otherwise. This motivates the following joint congestion control, scheduling, and routing algorithm:

Algorithm 5: **Joint Congestion Control, Routing, and Scheduling Algorithm**

1) Congestion control: the source of flow s sets its rate as $x_s(\lambda) = U_s'^{-1}(\lambda_s)$.
2) Scheduling:
   - For each link (i, j), find destination k^* such that $k^* \in \arg\max_k (\lambda_i^k - \lambda_j^k)$ and define $w_{ij}^* := \lambda_i^{k^*} - \lambda_j^{k^*}$.
   - Choose an $\tilde{\mathbf{f}} \in \arg\max_{\mathbf{f} \in \boldsymbol{\Pi}} \sum_{(i,j)\in L} w_{ij}^* f_{ij}$ such that $\tilde{\mathbf{f}}$ is an extreme point. Those links (i, j) with $\tilde{f}_{ij} > 0$ will transmit and other links (i, j) (with $\tilde{f}_{ij} = 0$) will not.
3) Routing: over link $(i, j) \in L$ with $\tilde{f}_{ij} > 0$, send data for destination k^* at full link capacity c_{ij}.
4) Price update: each node i updates the price on the queue for destination k according to

$$\lambda_i^k(t+1) = \left[\lambda_i^k(t) + \beta\left(x_i^k(\boldsymbol{\lambda}(t)) - \sum_{j:(i,j)\in L} f_{ij}^k(\boldsymbol{\lambda}(t)) + \sum_{j:(j,i)\in L} f_{ji}^k(\boldsymbol{\lambda}(t))\right)\right]^+. \tag{88}$$

The w_{ij}^* values represent the maximum differential congestion price of destination k between nodes i and j, and was introduced in [138]. The above algorithm uses back pressure to perform optimal scheduling and hop-by-hop routing. This is an illustrating case study on the potential interactions between back-pressure-based scheduling and dual decomposition for protocol stack design, where the "pressures" are the congestion prices.

Method 16: Maximum Differential Congestion Pricing for Node-Based Back-Pressure Scheduling.

There are several variations that have lead to an array of alternative decompositions, as will be discussed in Section IV-C. For example, instead of using a dual-driven algorithm, as in [17] and [99], where the congestion control part (Step 1 above) is static, the algorithms in [35] and [129] are primal-dual-driven, where the source congestion control algorithm can be interpreted as an ascent algorithm for the primal problem.

Method 17: Architectural Implication Due to Dual Decomposition: Absorb Routing Functionality Into Congestion Control and Scheduling.

Starting with a different set of optimization variables and a given NUM formulation, another family of variations uses a link-centric formulation, rather than the node-centric one studied in this section so far. Compared to the dual-decomposition of the link-centric formulation, Algorithm 5 has the following two major advantages. First, it can accommodate hop-by-hop routing without further modeling of multipath routing, and the routing is absorbed into end-to-end congestion control and per-contention-region scheduling. Second, it only requires the congestion price at the source, rather than the sum of congestion prices along the path, to accomplish congestion control. However, it also suffers from a more complicated set of dual variables: each node has to maintain a per-destination queue. More discussions on the tradeoffs between node-centric and link-centric formulations can be found in [85].

Now back to generalized NUM (85). It is known that the algorithm converges statistically to a *neighborhood* of the optimal point using *constant* stepsize, in the sense that the time averages tend to the optimal values arbitrarily closely. Specifically, let the primal function (the total achieved network utility) be $P(\mathbf{x})$ and let $\mathbf{x}^*$ be the optimum. Let $\overline{\mathbf{x}}(t) := (1/t)\sum_{\tau=0}^{t}\mathbf{x}(\tau)$ be the running average rates. Similarly, let $D(\boldsymbol{\lambda})$ be the dual objective function, $\boldsymbol{\lambda}^*$ be an optimal value of the dual variable, and $\overline{\boldsymbol{\lambda}}(t) := (1/t)\sum_{\tau=1}^{t}\boldsymbol{\lambda}(\tau)$ be the running average prices.

Theorem 12: Consider the dual of (85) and suppose the subgradient of the dual objective function is uniformly bounded. Then, for any $\delta > 0$, there exists a sufficiently small stepsize β in (88) such that

$$\liminf_{t\to\infty} P(\overline{\mathbf{x}}(t)) \geq P(\mathbf{x}^*) - \delta$$
$$\limsup_{t\to\infty} D\big(\overline{\boldsymbol{\lambda}}(t)\big) \leq D(\boldsymbol{\lambda}^*) + \delta.$$

The most difficult step in Algorithm 5 is scheduling. Solving it exactly requires a centralized computation which is clearly impractical in large-scale networks. Various scheduling algorithms and distributed heuristics have been proposed in the context of joint rate allocation, routing, and scheduling. The effects of imperfect scheduling on cross-layer design have recently been characterized in [83], for both the case when the number of users in the system is fixed and the case with dynamic arrivals and departures of the users.

IV. DECOMPOSITION METHODS

Various decomposition methods have been used in the last two sections on horizontal and vertical decompositions. In this section, we provide a more comprehensive discussion on the theory of optimization decomposition, first on primal and dual decomposition for decoupling constraints, then on consistency pricing for decoupling objective function, and finally on alternative decompositions.

A. Decoupling Coupled Constraints

The basic idea of a decomposition is to decompose the original large problem into subproblems which are then coordinated by a master problem by means of some kind of signalling, often without the need to solve the master problem centrally either [5]. Many of the existing decomposition techniques can be classified into *primal decomposition* and *dual decomposition* methods. The former (also called partitioning of variables, decomposition by right-hand side allocation, or decomposition with respect to variables) is based on decomposing the original primal problem, whereas the latter (also termed Lagrangian relaxation of the coupling constraints or decomposition with respect to constraints) is based on decomposing the dual of the problem. As illustrated in Fig. 15, primal decomposition methods have the interpretation that the master problem directly gives each subproblem an amount of resources that it can use; the role of the master problem is then to properly allocate the existing resources. In computer engineering terminology, the master problem adapts the *slicing* of resources among competing demands. In dual decomposition methods, the master problem sets the price for the resources to each subproblem which has to decide the amount of resources to be used depending on the price; the role of the master problem is then to obtain the best pricing strategy. In many cases, it is desirable and possible to solve the master problem distributively through message passing, which can be local or global, implicit or explicit. In summary, the engineering mechanism realizing dual decomposition is *pricing feedback* while that realizing primal decomposition is *adaptive slicing*.

Note that the terminology of "primal-dual" has a number of different meanings. For example, "primal-dual interior-point method" is a class of algorithms for centralized computation of an optimum for convex optimization, and "primal-dual distributed algorithm" is sometimes used to describe any algorithm that solves the primal and dual problems simultaneously. Two other sets of related

Table 7 Summary of Main Notation for Section IV

Symbol	*Meaning*
$\tilde{U}$	Utility function of subproblem
f	Primal objective function or constraint function
h	Primal constraint function
g	Lagrange dual function
$\boldsymbol{\lambda}, \gamma$	Dual variables
$\mathbf{s}$	Subgradients

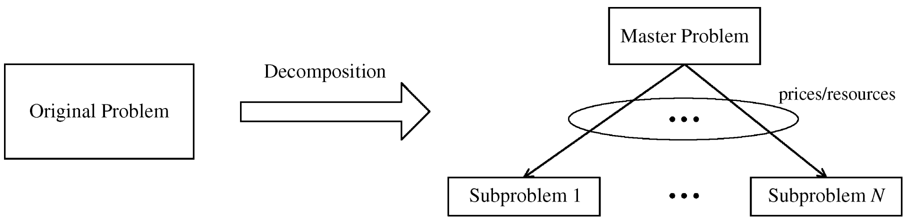


Fig. 15. *Schematic illustrating optimization problem decomposition.*

terminology have been used in previous sections. First, "primal-driven," "dual-driven," and "primal-dual-driven" algorithms are used to differentiate where is the update dynamics carried out: over the primal variables, or over the dual variables, or over both. Second, "penalty-function-based algorithms" refer to those distributed algorithms obtained by moving the coupled constraints to the augmented objective function in the primal problem through a penalty term. This is in contrast to "dual-decomposition-based algorithms" that are obtained through dual decomposition. In this section, primal and dual decompositions have yet a different set of meanings: decomposing coupling constraints through direct resource allocation and indirect pricing control, respectively.

It is also important to note that a given *decomposition method* may in turn lead to more than one *distributed algorithm*. Primal and dual decompositions leverage decomposability structures in a given optimization problem to turn it into subproblems coordinated by a master problem. Different distributed algorithms may then be developed based on the same decomposition, e.g., depending on the choice of update method (e.g., gradient or cutting plane or ellipsoid method), the ordering of variable updates (e.g., Jacobi or Gauss-Siedel), and the time scale of nested loops.

1) Dual Decomposition of the Basic NUM: We first illustrate how the dual decomposition approach can be applied to the basic NUM problem to produce the standard dual-decomposition-based distributed algorithm. Assume that the utility functions are concave, and possibly linear functions. The Lagrange dual problem of (4) is readily derived. We first form the Lagrangian

$$L(\mathbf{x},\boldsymbol{\lambda}) = \sum_s U_s(x_s) + \sum_l \lambda_l \left(c_l - \sum_{s\in S(l)} x_s \right)$$

where $\lambda_l \geq 0$ is the Lagrange multiplier (i.e., link price) associated with the linear flow constraint on link l. Additivity of total utility and linearity of flow constraints lead to a Lagrangian dual decomposition into individual source terms

$$\begin{aligned} L(\mathbf{x},\boldsymbol{\lambda}) &= \sum_s \left[U_s(x_s) - \left(\sum_{l\in L(s)} \lambda_l \right) x_s \right] + \sum_l c_l \lambda_l \\ &= \sum_s L_s(x_s, q_s) + \sum_l c_l \lambda_l \end{aligned}$$

where $q_s = \sum_{l\in L(s)} \lambda_l$. For each source s, $L_s(x_s, q_s) = U_s(x_s) - q_s x_s$ only depends on local rate x_s and the path price q_s (i.e., sum of λ_l on links used by source s).

The Lagrange dual function $g(\boldsymbol{\lambda})$ is defined as the maximized $L(\mathbf{x},\boldsymbol{\lambda})$ over $\mathbf{x}$ for a given $\boldsymbol{\lambda}$. This "net utility" maximization obviously can be conducted distributively by each source

$$x_s^*(q_s) = \operatorname{argmax}[U_s(x_s) - q_s x_s], \quad \forall s. \tag{89}$$

Such Lagrangian maximizer $\mathbf{x}^*(\boldsymbol{\lambda})$ will be referred to as price-based rate allocation (for a given price $\boldsymbol{\lambda}$). The Lagrange dual problem of (4) is

$$\begin{aligned} &\text{minimize} \quad g(\boldsymbol{\lambda}) = L(\mathbf{x}^*(\boldsymbol{\lambda}), \boldsymbol{\lambda}) \\ &\text{subject to} \quad \boldsymbol{\lambda} \geq 0 \end{aligned} \tag{90}$$

where the optimization variable is $\boldsymbol{\lambda}$. Since $g(\boldsymbol{\lambda})$ is the pointwise supremum of a family of affine functions in $\boldsymbol{\lambda}$, it is convex and (90) is a convex minimization problem. Since $g(\boldsymbol{\lambda})$ may be nondifferentiable, an iterative *subgradient* method can be used to update the dual variables $\boldsymbol{\lambda}$ to solve the dual problem (90)

$$\lambda_l(t+1) = \left[\lambda_l(t) - \beta(t) \left(c_l - \sum_{s\in S(l)} x_s(q_s(t)) \right) \right]^+, \quad \forall l \tag{91}$$

where $c_l - \sum_{s\in S(l)} x_s(q_s(t))$ is the lth component of a subgradient vector of $g(\boldsymbol{\lambda})$, t is the iteration number, and $\beta(t) > 0$ are step sizes. Certain choices of step sizes, such as $\beta(t) = \beta_0/t$, $\beta > 0$, guarantee that the sequence of dual variables $\boldsymbol{\lambda}(t)$ converges to the dual optimal $\boldsymbol{\lambda}^*$ as $t \to \infty$. It can be shown that the primal variable $\mathbf{x}^*(\boldsymbol{\lambda}(t))$ also converges to the primal optimal variable $\mathbf{x}^*$. For a primal problem that is a convex optimization, the convergence is towards a global optimum.

In summary, the sequence of source and link algorithms (89), (91) forms a standard dual-decomposition-based distributed algorithm that globally solves NUM (4) and the dual problem (90), i.e., computes an optimal rate vector $\mathbf{x}^*$ and optimal link price vector $\boldsymbol{\lambda}^*$. Note that no explicit signaling is needed. This is because the subgradient is precisely the difference between the fixed link capacity and the varying traffic load on each link, and the subgradient update equation has the interpretation of weighted queuing delay update.

The general methodology of primal and dual decompositions is now presented. A more comprehensive tutorial can be found in [104]. It turns out that primal and dual decompositions are also interchangeable through alternative representation of the optimization problem.

2) Primal Decomposition: A primal decomposition is appropriate when the problem has a coupling variable such that, when fixed to some value, the rest of the optimization problem decouples into several subproblems. Consider, for example, the following problem over $\mathbf{y}$, $\{\mathbf{x}_i\}$:

$$\begin{aligned} \text{maximize} \quad & \sum_i f_i(\mathbf{x}_i) \\ \text{subject to} \quad & \mathbf{x}_i \in \mathcal{X}_i, \quad \forall i \\ & \mathbf{A}_i \mathbf{x}_i \le \mathbf{y}, \quad \forall i \\ & \mathbf{y} \in \mathcal{Y}. \end{aligned} \tag{92}$$

If variable $\mathbf{y}$ were fixed, then the problem would decouple. This suggests separating the optimization in (92) into two levels of optimization. At the lower level, we have the subproblems, one for each i over $\mathbf{x}_i$, in which (92) decouples when $\mathbf{y}$ is fixed

$$\begin{aligned} \text{maximize} \quad & f_i(\mathbf{x}_i) \\ \text{subject to} \quad & \mathbf{x}_i \in \mathcal{X}_i \\ & \mathbf{A}_i \mathbf{x}_i \le \mathbf{y}. \end{aligned} \tag{93}$$

At the higher level, we have the master problem in charge of updating the coupling variable $\mathbf{y}$ by solving

$$\begin{aligned} \text{maximize} \quad & \sum_i f_i^*(\mathbf{y}) \\ \text{subject to} \quad & \mathbf{y} \in \mathcal{Y} \end{aligned} \tag{94}$$

where $f_i^*(\mathbf{y})$ is the optimal objective value of problem (93) for a given $\mathbf{y}$.

A subgradient for each $f_i^*(\mathbf{y})$ is given by

$$\mathbf{s}_i(\mathbf{y}) = \boldsymbol{\lambda}_i^*(\mathbf{y}) \tag{95}$$

where $\boldsymbol{\lambda}_i^*(\mathbf{y})$ is the optimal Lagrange multiplier corresponding to the constraint $\mathbf{A}_i\mathbf{x}_i \le \mathbf{y}$ in problem (93). The global subgradient is then $\mathbf{s}(\mathbf{y}) = \sum_i \mathbf{s}_i(\mathbf{y}) = \sum_i \boldsymbol{\lambda}_i^*(\mathbf{y})$. The subproblems in (93) can be locally and independently solved with the knowledge of $\mathbf{y}$.

3) Dual Decomposition: A dual decomposition is appropriate when the problem has a coupling constraint such that, when relaxed, the optimization problem decouples into several subproblems. Consider, for example, the following problem:

$$\begin{aligned} \text{maximize} \quad & \sum_i f_i(\mathbf{x}_i) \\ \text{subject to} \quad & \mathbf{x}_i \in \mathcal{X}_i \quad \forall i \\ & \sum_i \mathbf{h}_i(\mathbf{x}_i) \le \mathbf{c}. \end{aligned} \tag{96}$$

If the constraint $\sum_i \mathbf{h}_i(\mathbf{x}_i) \le \mathbf{c}$ were absent, then the problem would decouple. This suggests relaxing the coupling constraint in (96) as

$$\begin{aligned} \text{maximize} \quad & \sum_i f_i(\mathbf{x}_i) - \boldsymbol{\lambda}^T\left(\sum_i \mathbf{h}_i(\mathbf{x}_i) - \mathbf{c}\right) \\ \text{subject to} \quad & \mathbf{x}_i \in \mathcal{X}_i \quad \forall i \end{aligned} \tag{97}$$

such that the optimization separates into two levels of optimization. At the lower level, we have the subproblems, one for each i over $\mathbf{x}_i$, in which (97) decouples

$$\begin{aligned} \text{maximize} \quad & f_i(\mathbf{x}_i) - \boldsymbol{\lambda}^T \mathbf{h}_i(\mathbf{x}_i) \\ \text{subject to} \quad & \mathbf{x}_i \in \mathcal{X}_i. \end{aligned} \tag{98}$$

At the higher level, we have the master dual problem in charge of updating the dual variable $\boldsymbol{\lambda}$ by solving the dual problem

$$\begin{aligned} \text{minimize} \quad & g(\boldsymbol{\lambda}) = \sum_i g_i(\boldsymbol{\lambda}) + \boldsymbol{\lambda}^T \mathbf{c} \\ \text{subject to} \quad & \boldsymbol{\lambda} \ge \mathbf{0} \end{aligned} \tag{99}$$

where $g_i(\boldsymbol{\lambda})$ is the dual function obtained as the maximum value of the Lagrangian solved in (98) for a given $\boldsymbol{\lambda}$. This approach is in fact solving the dual problem instead of the original primal one. Hence, it will only give appropriate results if strong duality holds.

A subgradient for each $g_i(\boldsymbol{\lambda})$ is given by

$$\mathbf{s}_i(\boldsymbol{\lambda}) = -\mathbf{h}_i\left(\mathbf{x}_i^*(\boldsymbol{\lambda})\right) \tag{100}$$

where $\mathbf{x}_i^*(\boldsymbol{\lambda})$ is the optimal solution of problem (98) for a given $\boldsymbol{\lambda}$. The global subgradient is then $\mathbf{s}(\boldsymbol{\lambda}) = \sum_i \mathbf{s}_i(\boldsymbol{\lambda}) + \mathbf{c} = \mathbf{c} - \sum_i \mathbf{h}_i(\mathbf{x}_i^*(\boldsymbol{\lambda}))$. The subproblems in (98) can be locally and independently solved with knowledge of $\boldsymbol{\lambda}$.

Method 18: Primal and Dual Decomposition for Coupling Constraints.

Not all coupling constraints can be readily decomposed through primal or dual decompositions. For example, the feasibility set of SIR in wireless cellular network power control problems is coupled in a way with no obvious decomposability structure. A reparametrization of the constraint set is required before dual decomposition can be applied [47]. Sometimes, the coupling is time-invariant as in some broadband access networks, and very efficient "static pricing" can be used to decouple such "static coupling" [52].

B. Decoupling Coupled Objective

Examining the dual decomposition of the basic NUM reveals the following reasons why distributed and end-to-end algorithms can solve the basic NUM (4):

1) Separability in objective function: The network utility is a sum of individual source utilities.
2) Additivity in constraint functions: The linear flow constraints are summing over the individual flows.
3) Interchangeability of summation index: $\sum_l \lambda_l \sum_{s\in S(l)} x_s = \sum_s x_s \sum_{l\in L(s)} \lambda_l$.
4) Zero duality gap.

Property 3 is trivial. When Property 2 is violated, decomposition is much harder and usually involves some reparametrization of the constraint set. When Property 4 does not hold, recent works have provided three alternative solutions, as will be outlined in Section V-E.

For cases where Property 1 fails, recent progress on coupled utility formulations has been made [23], [130]. In many communication systems, utilities are indeed coupled. An example of the cooperation model can be found in networks where some nodes form a cluster and the utility obtained by each of them depends on the rate allocated to others in the same cluster (this can be interpreted as a hybrid model of selfish and nonselfish utilities). An example of the competition model is in wireless power control and DSL spectrum management, where the utilities are functions of SIRs that are dependent on the transmit powers of other users.

The generalized NUM problem considered in this subsection is

$$\begin{aligned} \text{maximize} \quad & \sum_k U_k\left(\mathbf{x}_k, \{\mathbf{x}_l\}_{l\in\mathcal{L}(k)}\right) \\ \text{subject to} \quad & \mathbf{x}_k \in \mathcal{X}_k \quad \forall k \\ & \sum_{k=1}^{K} \mathbf{h}_k(\mathbf{x}_k) \le \mathbf{c} \end{aligned} \qquad (101)$$

where the (strictly concave) utilities U_k depend on a vector local variable $\mathbf{x}_k$ and on variables of other utilities $\mathbf{x}_l$ for $l \in \mathcal{L}(k)$ (i.e., coupled utilities), $\mathcal{L}(k)$ is the set of nodes coupled with the kth utility, the sets $\mathcal{X}_k$ are arbitrary convex sets, and the coupling constraining function $\sum_k \mathbf{h}_k(\mathbf{x}_k)$ is not necessarily linear, but still convex. Note that this model has two types of coupling: coupled constraints and coupled utilities.

One way to tackle the coupling problem in the utilities is to introduce auxiliary variables and additional equality constraints, thus transferring the coupling in the objective function to coupling in the constraints, which can be decoupled by dual decomposition and solved by introducing additional *consistency pricing*. It is reasonable to assume that if two nodes have their individual utilities dependent on each other's local variables, then there must be some communication channels in which they can locally exchange pricing messages. It turns out that the global link congestion price update of the standard dual-decomposition-based distributed algorithm is not affected by the local consistency price updates, which can be conducted via these local communication channels among the nodes.

The first step is to introduce in problem (101) the auxiliary variables $\mathbf{x}_{kl}$ for the coupled arguments in the utility functions and additional equality constraints to enforce consistency

$$\begin{aligned} \text{maximize} \quad & \sum_k U_k\left(\mathbf{x}_k, \{\mathbf{x}_{kl}\}_{l\in\mathcal{L}(k)}\right) \\ \text{subject to} \quad & \mathbf{x}_k \in \mathcal{X}_k \quad \forall k \\ & \sum_k \mathbf{h}_k(\mathbf{x}_k) \le \mathbf{c} \\ & \mathbf{x}_{kl} = \mathbf{x}_l, \quad \forall k, l \in \mathcal{L}(k). \end{aligned} \qquad (102)$$

Next, to obtain a distributed algorithm, we take a dual decomposition approach by relaxing all the coupling constraints in problem (102)

$$\begin{aligned} \text{maximize} \quad & \sum_k U_k\left(\mathbf{x}_k, \{\mathbf{x}_{kl}\}_{l\in\mathcal{L}(k)}\right) + \boldsymbol{\lambda}^T\left(\mathbf{c} - \sum_k \mathbf{h}_k(\mathbf{x}_k)\right) \\ & + \sum_{k,l\in\mathcal{L}(k)} \boldsymbol{\gamma}_{kl}^T(\mathbf{x}_l - \mathbf{x}_{kl}) \\ \text{subject to} \quad & \mathbf{x}_k \in \mathcal{X}_k \quad \forall k \\ & \mathbf{x}_{kl} \in \mathcal{X}_l \quad \forall k, l \in \mathcal{L}(k) \end{aligned} \qquad (103)$$

where $\boldsymbol{\lambda}$ are the *congestion prices* and the $\boldsymbol{\gamma}_{kl}$'s are the *consistency prices*. By exploiting the additivity structure of the Lagrangian, the Lagrangian is separated into many subproblems where maximization is done using local variables (the kth subproblem uses *only* variables with the first subscript index k). The optimal value of (103) for a given set of $\boldsymbol{\gamma}_{kl}$'s and $\boldsymbol{\lambda}$ defines the dual function $g(\{\boldsymbol{\gamma}_{kl}\}, \boldsymbol{\lambda})$. The dual problem is then

$$\begin{aligned} \underset{\{\boldsymbol{\gamma}_{kl}\},\boldsymbol{\lambda}}{\text{minimize}} \quad & g(\{\boldsymbol{\gamma}_{kl}\}, \boldsymbol{\lambda}) \\ \text{subject to} \quad & \boldsymbol{\lambda} \ge \mathbf{0}. \end{aligned} \qquad (104)$$

It is worthwhile noting that (104) is equivalent to

$$\begin{aligned} \underset{\boldsymbol{\lambda}}{\text{minimize}} \quad & \left(\underset{\{\boldsymbol{\gamma}_{kl}\}}{\text{minimize}} \quad g(\{\boldsymbol{\gamma}_{kl}\}, \boldsymbol{\lambda}) \right) \\ \text{subject to} \quad & \boldsymbol{\lambda} \ge \mathbf{0}. \end{aligned} \qquad (105)$$

Problem (104) is easily solved by simultaneously updating the prices (both the congestion prices and the consistency prices) using a subgradient algorithm. In problem (105), however, the inner minimization is fully performed (by repeatedly updating the $\{\gamma_{kl}\}$) for each update of $\boldsymbol{\lambda}$. This latter approach implies two time scales: a fast time scale in which each cluster updates the corresponding consistency prices and a slow time scale in which the network updates the link prices; whereas the former approach has just one time scale.

Therefore, in problem (101), where the utilities U_k are strictly concave, the sets $\mathcal{X}_k$ are arbitrary convex sets, and the constraining functions $\mathbf{h}_k$ are convex, can be optimally solved by the following distributed algorithm:

- links update the congestion prices (the following vector equation can be carried out by each link autonomously as before):

$$\boldsymbol{\lambda}(t+1) = \left[\boldsymbol{\lambda}(t) - \beta_1\left(\mathbf{c} - \sum_k \mathbf{h}_k(\mathbf{x}_k)\right)\right]^+ \tag{106}$$

  where β_1 is the stepsize;
- the kth node, for all k, updates the consistency prices (at a faster or same time scale as the update of $\boldsymbol{\lambda}(t)$) as

$$\gamma_{kl}(t+1) = \gamma_{kl}(t) - \beta_2(\mathbf{x}_l(t) - \mathbf{x}_{kl}(t)), l \in \mathcal{L}(k) \tag{107}$$

  where β_2 is the stepsize, and then broadcast them to the coupled nodes within the cluster; and
- the kth node, for all k, locally solves the problem

$$\begin{array}{ll}\underset{\mathbf{x}_k,\{\mathbf{x}_{kl}\}_l}{\text{maximize}} & U_k\left(\mathbf{x}_k, \{\mathbf{x}_{kl}\}_{l\in\mathcal{L}(k)}\right) - \boldsymbol{\lambda}^T\sum_k \mathbf{h}_k(\mathbf{x}_k) \\ & +\left(\sum_{l:k\in\mathcal{L}(l)}\gamma_{lk}\right)^T\mathbf{x}_k - \sum_{l\in\mathcal{L}(k)}\gamma_{kl}^T\mathbf{x}_{kl} \\ \text{subject to} & \mathbf{x}_k \in \mathcal{X}_k \\ & \mathbf{x}_{kl} \in \mathcal{X}_l \quad \forall l \in \mathcal{L}(k)\end{array} \tag{108}$$

  where $\{\mathbf{x}_{kl}\}_{l\in\mathcal{L}(k)}$ are auxiliary local variables for the kth node.

Summarizing, all the links must advertise their local variables $\mathbf{x}_k$ (not the auxiliary ones $\mathbf{x}_{kl}$); congestion prices $\boldsymbol{\lambda}$ are updated as before, each link can update the corresponding γ_{kl}'s (with knowledge of the variables $\mathbf{x}_k$ of the coupled links) and signal it to the coupled links; each link can update the local variable $\mathbf{x}_k$ as well as the auxiliary ones $\mathbf{x}_{kl}$. The only additional price due to the coupled utilities is limited signaling between the coupled links within each cluster.

Method 19: Using Consistency Pricing to Decouple Coupled Utility Objective Functions.

C. Alternative Decompositions

Decomposition of a generalized NUM has significant implications to network protocol design along two directions: vertical (functional) decomposition into layers and horizontal (geographical) decomposition into distributed computation by network elements. There are many ways to decompose a given NUM formulation along both directions, providing a choice of different distributed algorithms and layering schemes. A systematic exploration of alternative decompositions is more than just an intellectual curiosity, it also derives different network architectures with a wide range of possibilities of communication overhead, computation load, and convergence behavior. This has been illustrated through some case studies in Sections III-C and III-D.

Alternative horizontal decomposition (i.e., distributed control across geographically disparate network elements) has been studied in [104], with applications to resource-constrained and direct-control rate allocation, and rate allocation among QoS classes with multipath routing.

Recent results on alternative vertical decomposition for a given NUM model (i.e., modularized control over multiple functional modules or layers) scatter in an increasingly large research literature. For example, on the topic of joint congestion control and multipath traffic engineering, different decompositions have been obtained in [34], [46], [49], [59], [69], [84], and [101]. On the topic of joint congestion control, routing, and scheduling, different decompositions have been obtained in [1], [17], [36], [83], and [129]. On the topic of joint congestion control and random access, different decompositions have been obtained in [74], [143], and [169]. On the topic of rate control for network coding-based multicast, different decompositions have been obtained in [6], [15], [86], [151], [152], and [157]. Some of these have been briefly discussed in Section III. A systematic treatise on this variety of vertical decompositions is an interesting research direction that will contribute to a rigorous understanding of the architectural choices of allocating functionalities to control modules.

One of the techniques that lead to alternatives of distributed architectures is to apply primal and dual decompositions recursively, as illustrated in Fig. 16. The basic decompositions are repeatedly applied to a problem to obtain smaller and smaller subproblems. For example, consider the following problem over $\mathbf{y}$, $\{\mathbf{x}_i\}$ which includes both a coupling variable and a coupling constraint

$$\begin{array}{ll}\text{maximize} & \sum_i f_i(\mathbf{x}_i, \mathbf{y}) \\ \text{subject to} & \mathbf{x}_i \in \mathcal{X}_i, \quad \forall i \\ & \sum_i \mathbf{h}_i(\mathbf{x}_i) \le \mathbf{c} \\ & \mathbf{A}_i\mathbf{x}_i \le \mathbf{y}, \quad \forall i \\ & \mathbf{y} \in \mathcal{Y}.\end{array} \tag{109}$$

One way to decouple this problem is by first taking a primal decomposition with respect to the coupling variable $\mathbf{y}$ and then a dual decomposition with respect to the coupling constraint $\sum_i \mathbf{h}_i(\mathbf{x}_i) \leq \mathbf{c}$. This would produce a two-level optimization decomposition: a master primal problem, a secondary master dual problem, and the subproblems. An alternative approach would be to first take a dual decomposition and then a primal one.

Another example that shows flexibility in terms of different decompositions is the following problem with two sets of constraints:

$$\begin{aligned} \text{maximize} \quad & f_0(\mathbf{x}) \\ \text{subject to} \quad & f_i(\mathbf{x}) \leq 0, \quad \forall i \\ & h_i(\mathbf{x}) \leq 0, \quad \forall i. \end{aligned} \tag{110}$$

One way to deal with this problem is via the dual problem with a full relaxation of both sets of constraints to obtain the dual function $g(\boldsymbol{\lambda}, \boldsymbol{\mu})$. At this point, instead of minimizing g directly with respect to $\boldsymbol{\lambda}$ and $\boldsymbol{\mu}$, it can be minimized over only one set of Lagrange multipliers first and then over the remaining one: $\min_{\boldsymbol{\lambda}} \min_{\boldsymbol{\mu}} g(\boldsymbol{\lambda}, \boldsymbol{\mu})$. This approach corresponds to first applying a full dual decomposition and then a primal one on the dual problem. The following lemma [105] characterizes the subgradient of the master problem at the top level.

Lemma 3: Consider the following partial minimization of the dual function:

$$g(\boldsymbol{\lambda}) = \inf_{\boldsymbol{\mu}} g(\boldsymbol{\lambda}, \boldsymbol{\mu}) \tag{111}$$

where $g(\boldsymbol{\lambda}, \boldsymbol{\mu})$ is the dual function defined as

$$g(\boldsymbol{\lambda}, \boldsymbol{\mu}) \triangleq \sup_{\mathbf{x} \in \mathcal{X}} \left\{ f_0(\mathbf{x}) - \sum_i \lambda_i f_i(\mathbf{x}) - \sum_i \mu_i h_i(\mathbf{x}) \right\}. \tag{112}$$

Then, $g(\boldsymbol{\lambda})$ is convex and a subgradient, denoted by $\mathbf{s}_{\boldsymbol{\lambda}}(\boldsymbol{\lambda})$, is given by

$$s_{\lambda_i}(\boldsymbol{\lambda}) = -f_i\left(\mathbf{x}^*(\boldsymbol{\lambda}, \boldsymbol{\mu}^*(\boldsymbol{\lambda}))\right) \tag{113}$$

where $\mathbf{x}^*(\boldsymbol{\lambda}, \boldsymbol{\mu})$ is the value of $\mathbf{x}$ that achieves the supremum in (112) for a given $\boldsymbol{\lambda}$ and $\boldsymbol{\mu}$, and $\boldsymbol{\mu}^*(\boldsymbol{\lambda})$ is the value of $\boldsymbol{\mu}$ that achieves the infimum in (111).

Alternatively, problem (110) can be approached via the dual but with a partial relaxation of only one set of constraints, say $f_i(\mathbf{x}) \leq 0, \forall i$, obtaining the dual function $g(\boldsymbol{\lambda})$ to be minimized by the master problem. Observe now that in order to compute $g(\boldsymbol{\lambda})$ for a given $\boldsymbol{\lambda}$, the partial Lagrangian has to be maximized subject to the remaining constraints $g_i(\mathbf{x}) \leq 0, \forall i$, for which yet another relaxation can be used. This approach corresponds to first applying a partial dual decomposition, and then, for the subproblem, another dual decomposition.

On top of combinations of primal and dual decompositions, there can also be different orderings of update, including the choice of parallel (Jacobi) or sequential (Gauss-Siedel) updates [5]. When there are more than one level of decomposition, and all levels conduct some type of iterative algorithms, such as the subgradient method, convergence and stability are guaranteed if the lower level master problem is solved on a faster time scale than the higher level master problem, so that at each iteration of a

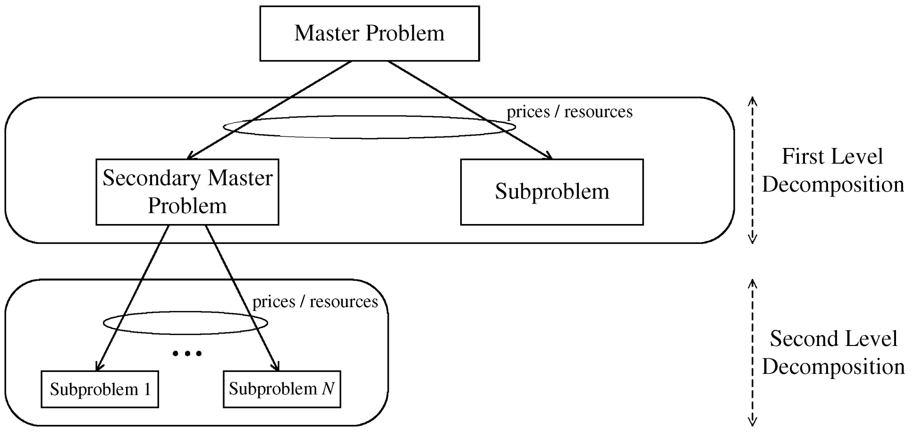


Fig. 16. ***Schematic illustrating multilevel decomposition.***

master problem all the problems at a lower level have already converged. If the updates of the different subproblems operate on similar time scales, convergence of the overall system may still be possible but requires more proof techniques [5], [118].

Method 20: Partial and Hierarchical Decompositions for Architectural Alternatives of the Protocol Stack.

As a more concrete example, consider the following special case of NUM in variables $(\mathbf{x}, \mathbf{y})$:

$$\begin{aligned} \text{maximize} \quad & \sum_i U_i(x_i) \\ \text{subject to} \quad & f_i(x_i, y_i) \le 0, \quad \forall i \\ & y_i \in \mathcal{Y}_i, \quad \forall i \\ & \sum_i h_i(x_i, y_i) \le 0 \end{aligned} \tag{114}$$

where $\mathbf{x}$ models the performance metrics that users utilities depend on and $\mathbf{y}$ models some resources that are globally coupled (the third constraint above) and have impacts on performance (the first constraint above). This problem has applications in distributed waterfilling algorithms in DSL spectrum management and distributed power control algorithms in wireless cellular networks, and can be decomposed in at least seven different ways following three general approaches below. Each decomposition results in a new possibility in striking the most appropriate tradeoff between computation and communication.

1) *A primal decomposition approach.* Problem (114) decouples if the y_i's are fixed. We can decompose the original problem into the master problem over $\mathbf{y}$

$$\begin{aligned} \text{maximize} \quad & \sum_i \tilde{U}_i(y_i) \\ \text{subject to} \quad & y_i \in \mathcal{Y}_i \quad \forall i \\ & \sum_i h_i(y_i) \le 0 \end{aligned} \tag{115}$$

where each $\tilde{U}_i(y_i)$ is the optimal objective value of the subproblem over x_i

$$\begin{aligned} \text{maximize} \quad & U_i(x_i) \\ \text{subject to} \quad & x_i \in \mathcal{X}_i \\ & f_i(x_i, y_i) \le 0. \end{aligned} \tag{116}$$

Each of the subproblems can be solved in parallel and only needs to know its local information (i.e., the local functions U_i, f_i and the local set $\mathcal{X}_i$) and the corresponding y_i (given by the master problem). Once each subproblem is solved, the optimal value $U_i(y_i)$ and possibly a subgradient can be communicated to the master problem. In this case, the master problem needs to communicate to each of the subproblems the available amount of resources y_i allocated.

2) *A full dual decomposition approach* with respect to all coupling constraints $f_i(x_i, y_i) \le 0$ and $\sum_i h_i(y_i) \le 0$. The master dual problem is to

$$\text{minimize} \quad g(\boldsymbol{\lambda}, \gamma) \tag{117}$$

over $\boldsymbol{\lambda}$, $\gamma \ge 0$, where $g(\boldsymbol{\lambda}, \gamma)$ is given by the sum of the optimal objective values of the following subproblems over (x_i, y_i) for each i

$$\begin{aligned} \text{maximize} \quad & U_i(x_i) - \lambda_i f_i(x_i, y_i) - \gamma h_i(y_i) \\ \text{subject to} \quad & x_i \in \mathcal{X}_i. \end{aligned} \tag{118}$$

Each of the subproblems can be solved in parallel and only needs to know its local information and the Lagrange multipliers λ_i and γ (given by the master problem). Once each subproblem is solved, the optimal value and possibly a subgradient (given by $-f_i(x_i, y_i)$ and $-h_i(y_i)$) can be communicated to the master problem. In this case, the master dual problem needs to communicate to each of the subproblems the private price λ_i and the common price γ.

3) *A partial dual decomposition approach* only with respect to the global coupling constraint $\sum_i h_i(y_i) \le 0$. The master dual problem over $\gamma \ge 0$

$$\text{minimize} \quad g(\gamma) \tag{119}$$

where $g(\gamma)$ is given by the sum of the optimal objective values of the following subproblems for all i:

$$\begin{aligned} \text{maximize} \quad & U_i(x_i) - \gamma h_i(y_i) \\ \text{subject to} \quad & x_i \in \mathcal{X}_i \\ & f_i(x_i, y_i) \le 0. \end{aligned} \tag{120}$$

Each of the subproblems can be solved in parallel and only needs to know its local information and

Table 8 Summary of Signalling Between the Master Problem and the Subproblems in the Three Decompositions Considered

	From master problem to ith subproblem	From ith subproblem to master problem
Approach 1 (primal decomp.)	amount of resources y_i	subgrad. of $\tilde{U}_i(y_i)$
Approach 2 (full dual decomp.)	prices λ_i and γ	subgrads. $-f_i(x_i, y_i)$ and $-h_i(y_i)$
Approach 3 (partial dual decomp.)	price γ	subgrad. $-h_i(y_i)$

the Lagrange multiplier γ (given by the master problem). Once each subproblem is solved, the optimal value and (possibly) a subgradient, given by $-h_i(y_i)$, can be communicated to the master problem. In this case, the master dual problem needs to communicate to each of the subproblems simply the common price γ.

We consider two of the metrics that can be used to compare alternative decompositions: the tradeoff between local computation and global communication through message passing, and the convergence speed.

The amount of signalling of the three decomposition methods is summarized in Table 8. In this particular problem, Approach 1 requires the largest amount of signalling in both directions. Approach 3 only requires a single common price from the master problem to the subproblems and a single number from each subproblem to the master problem. Approach 2 is intermediate, requiring the same amount as Approach 3 plus an additional individual price from the master problem to each subproblem.

For a particular instance of problem (114), Fig. 17 shows the convergence behavior in a numerical example for various distributed algorithms obtained by adding the choice of Jacobi or Gauss-Siedel update order on top of the three approaches in Table 8. The top two curves correspond to variations under Approaches 3 and 2, respectively.

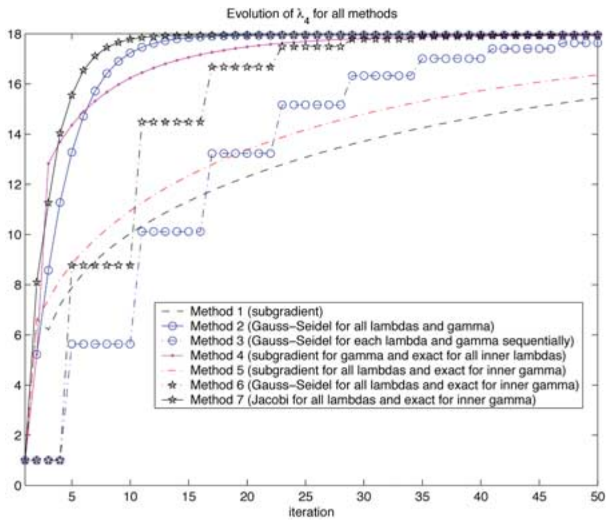


Fig. 17. ***Different speeds of convergence for seven different alternatives of horizontal decomposition in a numerical example of (114). Each curve corresponds to a different decomposition structure.***

Table 9 Summary of Main Notation for Section V

Symbol	*Meaning*
N_r	Number of active flows of type r
λ_r	The Poission arrival rate of flow of type r
μ_r	The mean of exponential file size in flow of type r
ρ_r	The load of flow of type r
$\mathbf{h}$	Channel state

Implicit in all decompositions is a choice of particular *representation* of the constraints. There are several ways to obtain different representation of the same NUM problem. One way is through substitution or change of variables [47]. The second way is to use a different set of variables, e.g., node-centric versus link-centric formulations in III-D. A third way is to group the variables in different orders. For example, in [16], it is shown that the joint TCP and MAC design problem may be formulated as maximizing network utility subject to the constraint that $\mathbf{FRx} \preceq \mathbf{c}$, where $\mathbf{F}$ is a contention matrix and $\mathbf{R}$ is the routing matrix. Then depending whether we first group $\mathbf{Rx}$ or $\mathbf{FR}$ in the constraint, the Lagrange dual variables we introduce are different, corresponding to either drawing a division between TCP and MAC or not.

In general, since every individual constraint (e.g., capacity of a link) gives rise to a corresponding Lagrange multiplier, we have the somewhat surprising consequence that the specific dual problems will be different depending on how the primal constraints are written, even redundant constraints that may change the dual problem properties.

As summarized in Fig. 18, each representation of a particular NUM may lead to different decomposability structures, different decomposition approaches, and the associated different distributed algorithms. Each of these algorithms may represent a different way to modularize and distribute the control of networks, and differs from the others in terms of engineering implications (more discussions in Section V-C).

V. FUTURE RESEARCH DIRECTIONS

Despite the wide range of progress made by many researchers over the last several years, there are still a variety of open issues in the area of "Layering as Optimization Decomposition." Some of these have formed a set of

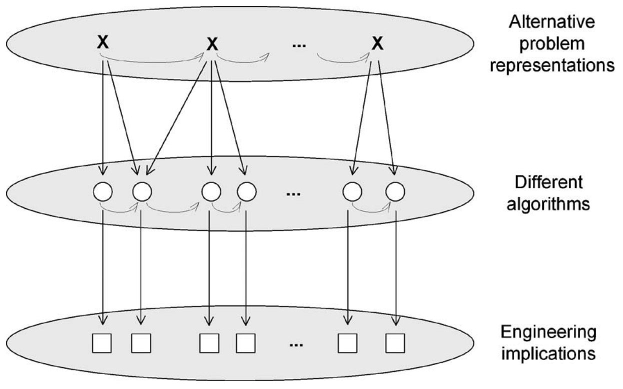


Fig. 18. ***Each alternative problem representation may lead to a choice of distributed algorithms with different engineering implications.***

widely recognized and well-defined open problems, which is in fact a sign indicating the maturity of a research area.

This section highlights some of the main issues for future research and their recent developments. Aside from technical challenges in proving desirable properties, such as global asymptotic convergence for different distributed algorithms under arbitrary delay [106], [107], we will classify the array of open issues in six groups.

A. Modeling and Complexity Challenges

First of all, there are semantic functionalities, such as session initiation and packet reordering, that we do not explicitly model. BGP in IP protocol and a variety of wireless *ad hoc* network routing protocols are yet to be fully incorporated in modeling language of NUM. Much further work also remains to be done to model utility functions in specific applications, especially inelastic, real-time applications such as VoIP [79] and streaming media where the notion of fairness may also need to be reconsidered [136]. In a more refined physical/link layer model, the option of forwarding rather than re-encoding at intermediate nodes must be considered, as well as per-hop retransmission schemes through ARQ. More NUM models are also needed to explicitly incorporate robustness with respect to algorithmic errors, network failures [160], multiple sets of constant parameters, and uncontrolled stochastic perturbances.

Several important modules commonly encountered in many cases of "Layering as Optimization Decomposition" still do not have simple, distributed solutions. An important topic is on distributed and suboptimal scheduling algorithms that have low spatial complexity (the amount and reach of explicit message passing do not grow rapidly with network size) and temporal complexity (the amount of backoff needed does not grow rapidly with network size) and can still guarantee certain throughput-delay performance. There have been many recent results on this topic, e.g., [12], [14], [80], [121], [149], [150], [165], and many more in print, based on different physical models, e.g., node-exclusive, SIR-based, and capture models.

Most of the designs have focused on the optimal message passing across layers and theoretically motivated choices of parameters such as stepsize. A systematic study on suboptimal message passing heuristics and practical guidelines in choosing algorithmic parameters would help characterize the tradeoff between complexity and suboptimality gap.

B. Research Issues Involving "Time"

Utility functions are often modeled as functions of equilibrium rates. For applications involving real-time control or multimedia communication, utility should instead be a function of latency or even the entire vector of rate allocation through the transients. How to maximize such utility functions remains an under-explored topic.

Different functions in each layer operate on time scales that may differ by several orders-of-magnitude different. For example, the application layer time scale is determined by the user behavior, the transport layer time scale by the round-trip-time in traversing the network, and the physical layer time scale by the physics of the transmission medium. Iterative algorithms themselves also have a time scale of operation determined by their rate of convergence, which is often difficult to bound tightly.

Furthermore, characterizing transient behaviors of iterative algorithms remains a challenging and under-explored topics in this area. For certain applications, if the resource allocation (e.g., window size, SIR) for a user drops below a threshold during the transient, the user may be disconnected. In such cases, the whole idea of equilibrium becomes meaningless. *Invariance* during transients [41], instead of *convergence* in the asymptote, becomes a more useful concept: how fast can the algorithm get close enough to the optimum and stay in that region? Usually the overall system performance derived out of a modularized design determines "how close is close enough" for each module's transients.

C. Alternative Decompositions

Even a different representation of the same NUM problem may change the duality and decomposability structures even though it does not change the optimal solution. It remains an open issue how to systematically explore the space of alternative vertical and horizontal decompositions, and thus the space of alternative network architectures, for a given set of requirements on, e.g., rate of convergence, symmetry of computational load distribution, and amount of explicit message passing.

An intellectually bold direction for future research is to explore if both the *enumeration* and *comparison* of alternative decompositions, horizontally and vertically, can be carried out systematically or even be automated.

To enumerate the set of possible decompositions and the associated sets of distributed algorithms, we have to take into account that transformations of the problem (e.g., change of variable) may lead to new decomposability structure, or turn a seemingly nondecomposable problem into a decomposable one [47]. This would open the door to even more choices of modularized and distributed network architectures with different properties.

To compare a variety of distributed algorithms, the following metrics all need to be considered: speed of convergence, the amount and symmetry of message passing for global communication, the distribution of local computational load, robustness to errors, failures, or network dynamics, the impact to performance metrics not directly incorporated into the objective function (e.g., user-perceived delay in throughput-based utility maximization formulations), the possibility of efficient relaxations and simple heuristics, and the ability to remain evolvable as the application needs change over time.

Some of the above metrics have no quantitative units of measurement, such as evolvability. Some do not have a universally agreed definition, such as the measure of *how* distributed an algorithm is.[11] Some are difficult to analyze accurately, such as the rate of convergence. Application contexts lead to a prioritization of these possibly conflicting metrics, based on which, the "best" decomposition can be chosen from the range of alternatives.

Summarizing, there are three stages of conceptual understanding of a decomposition-theoretic view of network architectures:

- First, modularized and distributed network architectures can be rigorously understood as decompositions of an underlying optimization problem.
- Second, there are in fact many alternatives of decompositions and therefore alternatives of network architectures. Furthermore, we can systematically explore and compare such alternatives.
- Third, there may be a methodology to exhaustively enumerate all alternatives, to quantify various comparison metrics, and even to determine *a priori* which alternative is the best according to any given combination of comparison metrics.

Many issues in the third stage of the above list remain open for future research.

D. Stochastic NUM

Stochastic theory of communication networks has a long and rich history. However, many key problems in this area remain open despite decades of effort. In his seminal paper [64], Kelly *et al.* used a deterministic fluid model to remove packet level details and microscopic queuing dynamics, followed by an optimization/game/control-theoretic approach. By assuming deterministic fluids with infinite backlog, we can avoid the difficulty of coupling across links in a general queuing network, and are often able to obtain insights on the structures of network resource allocation and functionality allocation.

An analogy can be drawn with Shannon's seminal work in 1948 [120]. By turning the focus from design of finite-blocklength codes to the regime of infinite-blocklength codes, thus enabling the law of large numbers to take effect, Shannon provided architectural principles (e.g., source-channel-separation) and established fundamental limits (e.g., channel capacity) in his mathematical theory of communication. Since then, the complicating issues associated with the design of practical codes have been brought back into the framework.

In the framework of "Layering as Optimization Decomposition," it is time to incorporate stochastic network dynamics, at session, packet, channel, and topology levels, back to the generalized NUM formulations. This leads to challenging models of queuing networks. For example, service rates of queues are determined by distributed solution to NUM, while parameters of NUM formulations are in turn stochastically varying according to states of the queues.

A combination of stochastic network control and optimization-based resource allocation raises challenging new questions, including stochastic stability, average case performance, outage performance, and, eventually, the distribution of attained utility (or other QoS parameters such as delay) as induced by the distributions of the stochastic models. *Stochastic stability* is the most basic and well-explored question in this area: under what conditions will a certain distributed algorithm of an NUM problem remain stochastically stable, in the sense that the number of flows and the total queue length in the network remain finite? Sometimes, deterministic optimization formulations can be derived from the limiting regime of stochastic network optimization, as in the case of social welfare maximization for loss networks in [110] and [111].

Stochastic models arise at four different levels due to many reasons:

- Session level (also referred to as flow level, connection level, or end-user level): flows arrive with finite workload and depart after finishing the workload, rather than holding infinite backlog and staying in the network forever. For certain combinations of the models of arrival process, utility functions, constraint sets, and time-scale separation,[12] researchers have established that the (stochastic) stability region of the basic NUM is the largest possible, which is the capacity region formed by the fixed link capacities in the deterministic NUM formulation. This means that satisfying the constraints in a deterministic formulation is both necessary and sufficient for stochastic stability.

[11]A common unit is the amount and reach of explicit message passing needed, and how they grow as the network grows.

[12]Here time-scale separation means that the resource allocation algorithm converges before the number of sessions changes.

Table 10 State-of-the-Art in Stochastic NUM

	Stability or Validation	Average Performance	Outage Performance	Fairness
Session Level	⋆⋆	⋆		⋆
Packet Level	⋆	⋆		
Channel Level	⋆⋆	⋆		
Topology Level				

- Packet level: packets of each flow arrive in bursts, and at a microscopic level go through probabilistic marking, and interact with uncontrolled flows such as UDP-based multimedia flows and web mice traffic. There have been at least three sets of results that appeared over the last several years. The first set shows many-flow asymptotical validation of fluid model (justifying the transition from microscopic to macroscopic model) [2], [30], [119], [139], and analyzes the interaction between congestion-controlled flows and uncontrolled flows under various queue regimes and different time-scale assumptions on flows' random variation [31], [113], [137], [164], [166]. The second set translates on–off HTTP session utility into transport layer TCP utility (mapping from microscopic to macroscopic model) [13]. The third set demonstrates convergence behavior for stochastic noisy feedback [170] (characterizing the impact of microscopic dynamics to macroscopic properties).
- Channel level: network transmission conditions are time-varying rather than fixed. Channel variations offer both the challenge to prove stability/optimality for existing algorithms and the ability to do opportunistic transmission and scheduling. We focus on the first set of issues in this survey.[13] For example, in [17], *stability* and *optimality* are established for dual-decomposition-based algorithms under channel-level stochastic for any convex optimization where the constraint set has the following structure: a subset of the variables lies in a polytope and other variables lie in a convex set that varies according to an irreducible, finite-state Markov chain.
- Topology level: Topology of wireless networks can change due to mobility of nodes, sleep mode, and battery power depletion. Solving generalized NUM problems over networks with randomly varying topology remains an under-explored area with little known results on models or methodologies. The problem is particularly challenging when the topology level stochastic dynamics is determined by battery usage, which is in turn determined by the solution of the NUM problem itself.

[13]Readers are referred to [85] for a survey of the second set of issues.

As shown in Table 10, where we use a system of zero to three stars to roughly represent the state of our understanding of the subject (from almost no results to complete characterization), much remains to be explored in this long over-due union between distributed optimization of networks and stochastic network theory. In the rest of this section, we briefly summarize some of the recent results in the first column of the table.

1) Session Level Stochastic: Consider session level dynamics characterized by the random arrivals and departures of sessions. For each type r, suppose for now that flows arrive and depart according to a Poisson process with intensity λ_r, and the size of the flows to be transmitted is exponentially distributed with mean $1/\mu_r$. The traffic load is $\rho_r = \lambda_r/\mu_r$. Let N_r be the number of ongoing flows, i.e., the number of type r flows in the network. It is a Markov process with the following transition rates:

- $N_r(t) \to N_r(t) + 1$, with rate λ_r;
- $N_r(t) \to N_r(t) - 1$, with rate $\mu_r x_r(t) N_r(t)$.

The stochastic version of the basic NUM formulation naturally becomes the following problem over $\mathbf{x}$:

$$\begin{aligned} \text{maximize} \quad & \sum_r N_r U_r(x_r) \\ \text{subject to} \quad & \sum_r R_{lr} x_r N_r \le c_l, \quad \forall l. \end{aligned} \tag{121}$$

Obviously, the definition of problem (121) depends on $\{N_r\}$, whose stochastic variations are in turn determined by the solution to problem (121).

For problem (121), it is shown [7], [29], [94], [162] that the stochastic stability region is the interior of feasibility rate region formed by the fixed link capacities, i.e., the following condition is sufficient to guarantee stochastic stability of the basic NUM for for α-fair utility functions with $\alpha > 0$:

$$\mathbf{R}\boldsymbol{\rho} < \mathbf{c}. \tag{122}$$

These results assumed time-scale separation. However, in many practical networks, session level stochastic operates on a fast time scale, with the arrive-and-depart process of flows varying constantly. Hence, instantaneous convergence of the rate allocation algorithm may not hold. The above result is extended in [81] and [126] to the case without time-scale separation assumption: [81] studies α-fair utilities using a discrete-time model and shows that there is an upper bound on the step size that would guarantee stochastic stability, and [126] shows similar results for α-fair utilities using the fluid limit model.

Other recent extensions include the following. Using the technique in [28] and [163] relaxes the assumption of Poisson arrivals, by studying a general stationary and a bursty network model. Recently, [68] studies a model that includes flows of two types, file transfers and streaming traffic, and generalizes the congestion control problem with convex constraints. And [8] correlates the utility maximization to classical queueing theory, and studies several typical utility functions and the stability condition. Then stochastic stability for any strictly concave maximization over general convex constraints without time-scale separation is recently reported.

An important and often invalid assumption in all these stability results is that file size (or workload) follows exponential distribution. In [66], a proper fluid model is formulated for exponentially distributed workload to study the "invariant states" as an intermediate step for obtaining diffusion approximation for all $\alpha \in (0, \infty)$. In [45], the fluid model is established for α-fair rate allocation, $\alpha \in (0, \infty)$, under general distributional condition on arrival process and service distribution. Using this fluid model, they have obtained characterization of "invariant states," which led to stability of network under α-fair allocation, $\alpha \in (0, \infty)$, when the network topology is a tree. For general network topologies, three recent preprints have tackled this difficult problem of stochastic stability under general file-size distribution, for special cases of utility functions: first, [11] establishes stability for max-min fair (corresponding to $\alpha = \infty$) rate allocation, then [93] establishes stability for proportional fair (corresponding to $\alpha = 1$) rate allocation for Poisson arrival and phase-type service distribution. Finally, using the fluid model in [45] but under a different scaling, [22] establishes the stability of α-fair rate allocation for general file-size distribution for a continuum of α: α sufficiently close to (but strictly larger than) zero, and a partial stability results for any $\alpha > 0$ fair allocation policy.

2) Packet Level Stochastic: Randomness at packet level may be a result of probabilistic marking of certain AQM schemes. It can also model "mice" traffic which is not captured in the standard NUM model. In [2], a detailed stochastic model is presented for N TCP Reno sources sharing a single bottleneck link with capacity Nc implementing RED. They show that as the number of sources and the link capacity both increase linearly, the queue process converges (in appropriate sense) to a deterministic process described by differential equations as usually assumed in the congestion control literature. Even though these results are proved only for a single bottleneck node, they provide a justification for the popular deterministic fluid model by suggesting that the deterministic process is the limit of a scaled stochastic process as the number of flows and link capacities scale to infinity [139].

Other convergence results are shown in [30] and [119]: the deterministic delay differential equation model with noise replaced by its mean value is accurate asymptotically in time and the number of flows. Because such convergence is shown asymptotically in time (except in the special case of log utility [119] where it is shown for each time), the trajectory of the stochastic system does not converge to that of the deterministic system in the many-flow regime [30]. However, [30] shows that the global stability criterion for a single flow is also that for the stochastic system with many flows, thus validating parameter design in the deterministic model even when networks have packet level stochastic dynamics.

Stochastic stability of greedy primal-dual-driven algorithm, a combination of utility maximization and maximum weight matching, is shown in [129] for dynamic networks where traffic sources and routers are randomly time-varying, interdependent, and limited by instantaneously available transmission and service rates.

Besides packet-level stochastic dynamics, there is also burstiness at the application level. Reference [13] considers its effect on TCP. It shows that the utility maximization at the transport layer induces a utility maximization at the application layer, i.e., an objective at the application layer is implemented in the transport layer. Specifically, consider a single link with capacity Nc (bits) shared by N HTTP-like flows. Each flow alternates between *think times* and *transfer times*. During the period of a think time, a flow does not require any bandwidth from the link. Immediately after a period of think time, the source starts to transmit a random amount of data by a TCP connection. The transfer time depends on the amount of transfer and the bandwidth allocation to this flow by TCP. The number of active flows is random, but at any time, the active flows share the link capacity according to TCP, i.e., their throughputs maximize aggregate utility subject to capacity constraints. Assume there are a fixed number of flow types. Then it is shown in [13] that the average throughput, i.e., the throughput aggregated over active flows of each type normalized by the total number of flows of that type, also solves a utility maximization problem with different utility functions as the TCP utility functions.

Yet another line of recent work [170] studies the impact of stochastic noisy feedback on dual-decomposition-based distributed algorithms, where the noise can be due to probabilistic marking, dropping, and contention-induced loss of packets. When the gradient estimator is unbiased, it is established, via a combination of the stochastic Lyapunov Stability Theorem and local analysis, that the iterates generated by distributed NUM algorithms converge with probability one to the optimal point, under standard technical conditions. In contrast, when the gradient estimator is biased, the iterates converge to a contraction region around the optimal point, provided that the biased terms are asymptotically bounded by a scaled version of the true gradients. These results confirm those derived based on deterministic models of feedback with errors [18], [95]. In the investigation of the rate of convergence for the unbiased case, it is found that, in general, the limit process of the interpolated process based on the normalized iterate sequence is a stationary reflected linear diffusion process, not necessarily a Gaussian diffusion process.

3) Channel Level Stochastic: Models in [17], [35], [83], [99], and [128] consider random channel fluctuations. For example, stability of primal-based algorithms under channel variations is established in [128]. In [17], the channel is assumed to be fixed within a discrete time slot but changes randomly and independently across slots. Let $\mathbf{h}(t)$ denote the channel state in time slot t. Corresponding to the channel state $\mathbf{h}$, the capacity of link l is $c_l(\mathbf{h})$ when active and the feasible rate region at the link layer is $\mathbf{\Pi}(\mathbf{h})$. We further assume that the channel state is a finite state process with identical distribution $q(\mathbf{h})$ in each time slot. Define the mean feasible rate region as

$$\bar{\mathbf{\Pi}} = \left\{ \bar{\mathbf{r}} : \bar{\mathbf{r}} = \sum_{\mathbf{h}} q(\mathbf{h})\mathbf{r}(\mathbf{h}), \mathbf{r}(\mathbf{h}) \in \mathbf{\Pi}(\mathbf{h}) \right\}. \tag{123}$$

The joint congestion control, routing, and scheduling algorithm discussed in Section III-D can be directly applied with the schedulable region $\mathbf{\Pi}$ in Step 2 replaced by the current feasible rate region $\mathbf{\Pi}(\mathbf{h}(t))$. It is proved in [17] that the prices $\boldsymbol{\lambda}(t)$ form a stable Markov process, by appealing to the generalized NUM (85) with the rate region $\mathbf{\Pi}$ replaced by the mean rate region $\bar{\mathbf{\Pi}}$

$$\begin{aligned} \text{maximize} \quad & \sum_s U_s(x_s) \\ \text{subject to} \quad & x_i^k \le \sum_{j:(i,j)\in L} f_{ij}^k - \sum_{j:(j,i)\in L} f_{ji}^k, \quad \forall i,j,k \\ & \mathbf{f} \in \bar{\mathbf{\Pi}}. \end{aligned} \tag{124}$$

Moreover, the primal and dual values along the trajectory converge arbitrarily close to their optimal values, with respect to (124), as the stepsize in the algorithm tends to zero.

For generalized NUM problems, [17] establishes the *stability* and *optimality* of dual-decomposition-based algorithms under channel-level stochastic for any convex optimization where the constraint set has the following structure: a subset of the variables lie in a polytope and other variables lie in a convex set that varies according to an irreducible, finite-state Markov chain. Algorithms developed from the deterministic NUM formulation and requiring only the knowledge of current network state remain stochastically stable and optimal (in the expectation, with respect to an optimization problem whose constraint is replaced by the average constraint set under the given channel variations).

E. Nonconvex NUM

It is widely recognized that the watershed between easy and hard optimization problems is convexity rather than linearity. Nonconvex optimization formulations of generalized NUM often appear, in at least four different forms. First is nonconcave utility, such as the sigmoidal utility that are more realistic models in applications including voice. Second is nonconvex constraint set, such as lower bounds on SIR as a function of transmit power vector, in the low-SIR regime of interference-limited networks. Third is integer constraint, such as those in single-path routing protocols. Fourth is convex constraint set requiring a description length that grows exponentially with the number of variables, such as certain schedulability constraints in multihop interference models. In general, such nonconvex optimization is difficult in theory and in practice, even through centralized computation.

In particular, nonconvex optimization problems often have nonzero duality gaps. A nonzero duality gap means that the standard dual-decomposition-based distributed subgradient algorithm may lead to suboptimal and even infeasible primal solutions and instability in cross layer interactions. Bounding, estimating, and reducing the resulting duality gap remains a challenging task. Sometimes, these very difficult problems can be tackled through a combination of well-established and more recent optimization techniques, e.g., sum-of-squares programming [108], [109] and geometric-signomial programming [19], [32], although some aspects of these techniques are not well understood (e.g., convergence to global optimality in signomial programming).

As an example, consider nonconvexity in the objective function. There have been three recent approaches to solve nonconcave utility maximization over linear constraints:

- Reference [77] proposes a distributed admission control method (for sigmoidal utilities) called the "self-regulating" heuristic, which is shown to avoid link congestion caused by sigmoidal utilities.
- Reference [48] determines optimality conditions for the dual-decomposition-based distributed algorithm to converge globally (for all nonlinear utilities). The engineering implication is that appropriate provisioning of link capacities will ensure global convergence of the dual-decomposition-based distributed algorithm even when user utility functions are nonconcave.

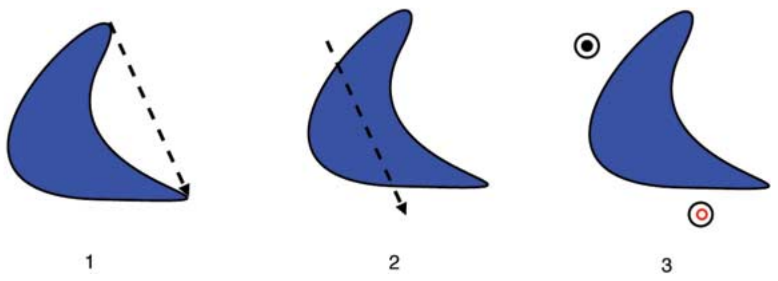


Fig. 19. *Three major types of approaches when tackling nonconvex NUM: Go 1) around, 2) through, or 3) above nonconvexity.*

- Reference [40] develops an efficient but centralized method to compute the global optimum (for a wide class of utilities that can be transformed into polynomial utilities), using the sum-of-squares method. However, no distributed versions of this method are available.

As illustrated in Fig. 19, there are at least three very different approaches to tackle the difficult issue of nonconvexity in either the objective function or the constraint set:

- *Go around nonconvexity*: discover a change of variables that turns the seemingly nonconvex problem into a convex one, determine conditions under which the problem is convex or the KKT point is unique [75], [124], or make approximations to make the problem convex. A popular example of tackling nonconvexity is the application of geometric programming to communication systems [19]. In some problems, an appropriate change of variables turns an apparently nonconvex problem into a convex one [131].
- *Go through nonconvexity*: use successive convex relaxations (e.g., sum-of-squares, signomial programming), utilize special structures in the problem (e.g., difference of convex functions, generalized quasi-concavity), or leverage smarter branch and bound methods.
- *Go above nonconvexity*: observe that optimization problem formulations are induced by some underlying assumptions on what the architectures and protocols should look like. By changing these assumptions, a different, much easier-to-solve or easier-to-approximate NUM formulations may result. We refer to this approach as *design for optimizability* [51], which concerns with selectively perturbing some underlying assumption to make the resulting NUM problem easier to solve. This approach of changing a hard problem into an easier one is in contrast to *optimization*, which tries to solve a given, possibly difficult NUM problem. A recent successful example of "design for optimizability" is on intra-domain routing in the Internet [159].

F. Quantifying Network X-ities

As we draw this paper towards the end, it is important to ask why should network operators optimize performance in the first place? Indeed, optimality is *not* the key point. Optimization is used here primarily as a modeling language and a starting point to develop and compare architectural choices, rather than just defining a particular point of operation at global optimum. Suboptimal, but simple (low spatial-temporal complexity) algorithms can be used in various modules (e.g., the scheduling module). As long as the suboptimality gap is bounded and the network architecture is "good," then the "damage" from the suboptimal design in one layer can be contained at the systems level. Similarly, stochastic dynamics may also wash away the worst cases and even be beneficial to the average system performance [83], again provided that the network architecture is appropriately designed. In such cases, it is also necessary to study the meaning of utility-suboptimality in terms of degradation to fairness, since $x\%$ of optimality loss may not imply $x\%$ degradation of fairness. In fact, even quantified metrics of unfairness are not well-established.

Protocols and layered architectures are not just for maximizing the efficiency of performance metrics, such as throughput, latency, and distortion, but also for ensuring security and for maximizing the important yet fuzzy metrics of robustness in operation, such as evolvability, scalability, availability, diagnosability, and manageability. Interactions among layers introduce the risks of losing robustness against unforseen demands arising over time or significant growth over space. Indeed, under the metrics of deployment cost and operations cost, the success of packet networks comes down to the scalability and evolvability of TCP/IP and the way control is modularized and distributed.

Despite their importance in practical network operations, these *network X-ities* remain as fuzzy or even ill-defined notions, and a quantified foundation for them is long overdue [24]. Intuitively, "design by decomposition" enhances scalability and evolvability, but may present risks to manageability such as diagnosability and optimizability. The benefits and risks arise together, in part because layering means that each layer is limited in what it can do (optimization variables in a decomposed subproblem) and what it can observe (a subset of constant parameters and variables in other decomposed subproblems). Throughout Sections II and III, we have illustrated how the framework of "Layering as Optimization Decomposition" helps make the decision of what each module should control and observe. Still, quantifying network X-ities, and trading-off network X-ities with performance metrics, in layered protocol stack designs remain a challenging long-term research direction.

We may also carry the intellectual thread from "forward-engineering" (solving a given problem) to "reverse-engineering" (finding the problem being solved by a given protocol) one step further to "design for optimizability." The difficulty of solving a particular set of subproblems may illustrate that the given decomposition was conducted possibly in a wrong way and suggests that better alternatives exist.

In summary, in order to fulfill the long-term goal of providing a simple, relevant abstraction of what makes a network architecture "good," the framework of "Layering as Optimization Decomposition" needs to move away from the restriction of one decomposition fits all, away from the focus on deterministic fluid models and asymptotic convergence, and even away from the emphasis on optimality, and instead towards "optimizability."

VI. CONCLUSION

We provide a survey of the recent efforts to establish "Layering as Optimization Decomposition" as a common "language" for systematic network design. "Layering as Optimization Decomposition" is a unifying framework for understanding and designing distributed control and cross-layer resource allocation in wired and wireless networks. It has been developed by various research groups since the late 1990s, and is now emerging to provide a mathematically rigorous and practically relevant approach to quantify the risks and opportunities in modifying the existing layered network architectures or in drafting clean-slate architectures. It shows that many existing network protocols can be reverse-engineered as implicitly solving some optimization-theoretic or game-theoretic problems. By distributively solving generalized NUM formulations through decomposed subproblems, we can also systematically generate layered protocol stacks. There are many alternatives for both horizontal decomposition into disparate network elements and vertical decomposition into functional modules (i.e., layers). A variety of techniques to tackle coupling and nonconvexity issues have become available, which enable developments of distributed algorithms and proofs of global optimality, respectively. Many such techniques are becoming standard methods to be readily invoked by researchers.

"Layering as Optimization Decomposition" provides a top-down approach to design layered protocol stacks from first principles. The resulting conceptual simplicity stands in contrast to the ever-increasing complexity of communication networks. The two cornerstones for the rigor and relevance of this framework are "networks as optimizers" and "layering as decomposition." Together, they provide a promising angle to understand not just *what* "works" in the current layered protocol stacks, but also *why* it works, what may *not* work, and what *alternatives* network designers have. Realizing the importance of functionality allocation and motivated by the view of "architecture first," we hope that there will be continuous progress towards a mathematical theory of network architectures. ■

Acknowledgment

The authors would like to gratefully acknowledge their coauthors in the research area surveyed here: L. Andrew, M. Bresler, L. Chen, H. Choe, J.-Y. Choi, M. Fazel, D. Gayme, P. Hande, J. He, S. Hegde, J. Huang, C. Jin, J.-W. Lee, J. Lee, L. Li, Y. Li, J. Liu, F. Paganini, D. Palomar, J. Pongsajapan, J. Rexford, D. Shah, R. Srikant, C. W. Tan, A. Tang, J. Wang, D. Wei, B. Wydrowski, and J. Zhang.

They also thank many colleagues for illuminating discussions, especially on network architectures and layering, with D. Acemoglu, V. Chan, D.-M. Chiu, A. Ephremides, N. Gershenfeld, A. Goldsmith, B. Fry, M. Johansson, K. Kar, F. P. Kelly, P. R. Kumar, X. Lin, Z. Liu, H. Newman, D. O'Neill, A. Proutiere, C. Ramming, S. Shakkottai, N. Shroff, J. Wieseltheir, X. Wang, Z.-K. Wang, L. Xiao, E. Yeh, Y. Yi, AT&T Labs, Bell Labs, and the Cisco ARTI group, on optimization theory, with S. Boyd, D. Gao, A. Ozdaglar, and P. Parrilo, and on network X-ities, with J. Kurose, R. Lee, D. Towsley, and D. Xu.

REFERENCES

[1] M. Andrews, "Joint optimization of scheduling and congestion control in communication networks," in *Proc. CISS*, Mar. 2006.

[2] F. Baccelli, D. R. McDonald, and J. Reynier, "A mean-field model for multiple TCP connections through a buffer implementing RED," INRIA, Tech. Rep. RR-4449, Apr. 2002.

[3] D. P. Bertsekas, "Dynamic behavior of shortest path routing algorithms for communication networks," *IEEE Trans. Autom. Control*, vol. 27, no. 2, pp. 60–74, Feb. 1982.

[4] ——, *Nonlinear Programming*, 2nd ed., Belmont, MA: Athena Scientific, 1999.

[5] D. P. Bertsekas and J. N. Tsitsiklis, *Parallel and Distributed Computation*. Englewood Cliffs, NJ: Prentice-Hall, 1989.

[6] S. Bhadra, S. Shakkottai, and P. Gupta, "Min-cost selfish multicast with network coding," in *Proc. IEEE NetCod Workshop*, Apr. 3–6, 2006.

[7] T. Bonald and L. Massoulie, "Impact of fairness on Internet performance," in *Proc. ACM SIGMETRICS*, 2001.

[8] T. Bonald, L. Masoulie, A. Proutiere, and J. Virtamo, "A queueing analysis of max-min fairness, proportional fairness and balanced

fairness," *Queueing Systems: Theory and Applications*, pp. 65–84, 2006.

[9] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge, U.K.: Cambridge Univ. Press, 2004.

[10] L. S. Brakmo and L. L. Peterson, "TCP Vegas: End to end congestion avoidance on a global Internet," *IEEE J. Sel. Areas Commun.*, vol. 13, no. 8, pp. 1465–1480, Oct. 1995.

[11] M. Bramson, "Stability of networks for max-min fair routing," presented at the INFORMS Applied Probability Conf., Ottawa, ON, Canada, 2005.

[12] L. Bui, A. Eryilmaz, R. Srikant, and X. Wu, "Joint congestion control and distributed scheduling in multihop wireless networks with a node exclusive interference model," in *Proc. IEEE INFOCOM*, Apr. 2006.

[13] C. S. Chang and Z. Liu, "A bandwidth sharing theory for a large number of HTTP-like connections," *IEEE/ACM Trans. Netw.*, vol. 12, no. 5, pp. 952–962, Oct. 2004.

[14] P. Chaporkar and S. Sarkar, "Stable scheduling policies for maximizing throughput in generalized constrained queuing," in *Proc. IEEE INFOCOM*, Apr. 2006.

[15] L. Chen, T. Ho, S. H. Low, M. Chiang, and J. C. Doyle, "Rate control for multicasting with network coding," in *Proc. IEEE INFOCOM*, May 2007.

[16] L. Chen, S. H. Low, and J. C. Doyle, "Joint TCP congestion control and medium access control," in *Proc. IEEE INFOCOM*, Mar. 2005.

[17] L. Chen, S. H. Low, M. Chiang, and J. C. Doyle, "Joint optimal congestion control, routing, and scheduling in wireless ad hoc networks," in *Proc. IEEE INFOCOM*, Apr. 2006.

[18] M. Chiang, "Balancing transport and physical layer in wireless multihop networks: Jointly optimal congestion control and power control," *IEEE J. Sel. Areas Commun.*, vol. 23, no. 1, pp. 104–116, Jan. 2005.

[19] ——, "Geometric programming for communication systems," *Foundations and Trends in Communications and Information Theory*, vol. 2, no. 1, pp. 1–156, Aug. 2005.

[20] ——, "Nonconvex optimization of communication systems," in *Special Volume in Nonconvex and Nonsmooth Optimization*, D. Y. Gao and H. D. Sherali, Eds. New York: Springer, 2007.

[21] M. Chiang and J. Bell, "Balancing supply and demand of bandwidth in wireless cellular networks: Utility maximization over powers and rates," in *Proc. IEEE INFOCOM*, Mar. 7–11, 2004, vol. 4, pp. 2800–2811.

[22] M. Chiang, D. Shah, and A. Tang, "Stochastic stability of network utility maximization: General filesize distribution," in *Proc. Allerton Conf.*, Sep. 2006.

[23] M. Chiang, C. W. Tan, D. Palomar, D. O'Neill, and D. Julian, "Power control by geometric programming," *IEEE Trans. Wireless Comm.*, 2007, to be published.

[24] M. Chiang and M. Yang, "Towards X-ities from a topological point of view: Evolvability and scalability," in *Proc. Allerton Conf.*, Oct. 2004.

[25] D. M. Chiu and R. Jain, "Analysis of the increase and decrease algorithms for congestion avoidance in computer networks," *Computer Networks and ISDN Systems*, vol. 17, no. 1, pp. 1–14, 1989.

[26] J. Y. Choi, K. Koo, D. X. Wei, J. S. Lee, and S. H. Low, "Global stability of FAST TCP," *Preprint*, 2006.

[27] R. L. Cruz and A. Santhanam, "Optimal routing, link scheduling, and power control in multihop wireless networks," in *Proc. IEEE INFOCOM*, Mar. 30–Apr. 3, 2003, vol. 1, pp. 702–711.

[28] J. G. Dai, "On positive harris recurrence of multiclass queueing networks: A unified approach via fluid limit models," *Annals Appl. Probability*, vol. 5, pp. 49–77, 1995.

[29] G. de Veciana, T. J. Lee, and T. Konstantopoulos, "Stability and performance analysis of network supporting elastic services," *IEEE/ACM Trans. Netw.*, vol. 9, no. 1, pp. 2–14, Feb. 2001.

[30] S. Deb, S. Shakkottai, and R. Srikant, "Asymptotic behavior of Internet congestion controllers in a many-flow regime," *Math. Operations Res.*, vol. 30, no. 2, pp. 420–440, May 2005.

[31] S. Deb and S. Srikant, "Rate-based versus queue-based models of congestion control," in *Proc. ACM SIGMETRICS*, 2004.

[32] R. J. Duffin, E. L. Peterson, and C. Zener, *Geometric Programming: Theory and Applications*. New York: Wiley, 1967.

[33] N. Dukkipati, M. Kobayashi, R. Zhang-Shen, and N. McKeown, "Processor sharing flows in the Internet," in *Proc. IEEE IWQoS*, Jun. 2005.

[34] A. Elwalid, C. Jin, S. H. Low, and I. Widjaja, "MATE: MPLS adaptive traffic engineering," in *Proc. IEEE INFOCOM*, Apr. 22–26, 2001, vol. 3, pp. 1300–1309.

[35] A. Eryilmaz and R. Srikant, "Fair resource allocation in wireless networks using queue-length-based scheduling and congestion control," in *Proc. IEEE INFOCOM*, Mar. 13–17, 2005, vol. 3, pp. 1794–1803.

[36] ——, "Joint congestion control, routing and MAC for stability and fairness in wireless networks," *IEEE J. Sel. Areas Commun.*, vol. 24, no. 8, pp. 1514–1524, Aug. 2006.

[37] FAST TCP Project. [Online]. Available: netlab.caltech.edu

[38] FAST Copper Project. [Online]. Available: www.princeton.edu/fastcopper

[39] Z. Fang and B. Bensaou, "Fair bandwidth sharing algorithms based on game theory frameworks for wireless ad-hoc networks," in *Proc. IEEE INFOCOM*, Mar. 7–11, 2004, vol. 2, pp. 1284–1295.

[40] M. Fazel and M. Chiang, "Nonconcave utility maximization through sum-of-squares method," in *Proc. IEEE CDC*, Dec. 2005.

[41] M. Fazel, D. Gayme, and M. Chiang, "Transient analysis of power control," in *Proc. IEEE GLOBECOM*, Nov. 2006.

[42] S. Floyd, "Highspeed TCP for large congestion windows," Internet Draft, draftloyd-cpighspeedtxt. [Online]. Available: http://www.icir.org/floyd/hstcp.html

[43] S. Floyd and V. Jacobson, "Random early detection gateways for congestion avoidance," *IEEE/ACM Trans. Netw.*, vol. 1, no. 4, pp. 397–413, Aug. 1993.

[44] T. G. Griffin, F. B. Shepherd, and G. Wilfong, "The stable path problem and interdomain routing," *IEEE/ACM Trans. Netw.*, vol. 10, no. 2, pp. 232–243, Apr. 2002.

[45] H. C. Gromoll and R. Williams, "Fluid limit of a network with fair bandwidth sharing and general document size distribution," *Preprint*, 2006.

[46] H. Han, S. Shakkottai, C. V. Hollot, R. Srikant, and D. Towsley, "Overlay TCP for multi-path routing and congestion control," in *Proc. IMA Workshop Measurement and Modeling of the Internet*, Jan. 2004.

[47] P. Hande, S. Rangan, and M. Chiang, "Distributed algorithms for optimal SIR assignment in cellular data networks," in *Proc. IEEE INFOCOM*, Apr. 2006.

[48] P. Hande, S. Zhang, and M. Chiang, "Distributed rate allocation for inelastic flows," *IEEE/ACM Trans. Netw.*, Feb. 2008, to be published.

[49] J. He, M. Bresler, M. Chiang, and J. Rexford, "Towards multi-layer traffic engineering: Optimization of congestion control and routing," *IEEE J. Sel. Areas Comm.*, 2007, to be published.

[50] J. He, M. Chiang, and J. Rexford, "TCP/IP interaction based on congestion prices: Stability and optimality," in *Proc. IEEE ICC*, Jun. 2006.

[51] J. He, J. Rexford, and M. Chiang, "Design principles of manageable networks," Princeton University Computer Science, Technical Report TR-770-06, Oct. 2006.

[52] R. Cendrillon, J. Huang, M. Chiang, and M. Moonen, "Autonomous spectrum balancing for digital subscriber lines," in *IEEE Trans. Sign. Proc.*, 2007, to be published.

[53] J. Huang, Z. Li, M. Chiang, and A. K. Katsaggelos, "Pricing-based rate control and joint packet scheduling for multi-user wireless uplink video streaming," in *Proc. IEEE Packet Video Workshop*, Apr. 2006.

[54] Internet 0 Meeting Materials. [Online]. Available: cba.mit.edu/events/04.09.I0/

[55] V. Jacobson, "Congestion avoidance and control," in *Proc. ACM SIGCOMM*, Aug. 1988.

[56] C. Jin, D. X. Wei, and S. H. Low, "FAST TCP: Motivation, architecture, algorithms, and performance," in *Proc. IEEE INFOCOM*, Mar. 2004.

[57] C. Jin, D. X. Wei, S. H. Low, G. Buhrmaster, J. Bunn, D. H. Choe, R. L. A. Cottrell, J. C. Doyle, W. Feng, O. Martin, H. Newman, F. Paganini, S. Ravot, and S. Singh, "FAST TCP: From theory to experiments," *IEEE Netw.*, vol. 19, no. 1, pp. 4–11, Jan./Feb. 2005.

[58] B. Johansson, P. Soldata, and M. Johansson, "Mathematical decomposition techniques for distributed cross-layer optimization of data networks," *IEEE J. Sel. Areas Commun.*, vol. 24, no. 8, pp. 1535–1547, Aug. 2006.

[59] S. Kandula and D. Katabi, "Walking the tightrope: Responsive yet stable traffic engineering," in *Proc. ACM SIGCOMM*, Aug. 2005.

[60] K. Kar, S. Sarkar, and L. Tassiulas, "Optimization based rate control for multipath sessions," in *Proc. Int. Teletraffic Congress*, Dec. 2001.

[61] ——, "Achieving proportional fairness using local information in Aloha networks," *IEEE Trans. Autom. Control*, vol. 49, no. 10, pp. 1858–1862, Oct. 2004.

[62] D. Katabi, M. Handley, and C. Rohrs, "Congestion control for high-bandwidth delay product networks," in *Proc. ACM SIGCOMM*, Aug. 2002.

[63] F. P. Kelly, "Fairness and stability of end-to-end congestion control," *European Journal of Control*, vol. 9, pp. 159–176, 2003.

[64] F. P. Kelly, A. Maulloo, and D. Tan, "Rate control for communication networks: Shadow prices, proportional fairness and stability," *J. Operations Res. Soc.*, vol. 49, no. 3, pp. 237–252, Mar. 1998.

[65] F. P. Kelly and T. Voice, "Stability of end-to-end algorithms for joint routing and rate control," *Comput. Commun. Rev.*, vol. 35, no. 2, pp. 5–12, Jan. 2005.

[66] F. P. Kelly and R. J. Williams, "Fluid model for a network operating under a fair bandwidth-sharing policy," *Ann. Appl. Probability,* vol. 14, pp. 1055–1083, 2004.

[67] T. Kelly, "Scalable TCP: Improving performance in highspeed wide area networks," *Comput. Commun. Rev.*, vol. 32, no. 2, Apr. 2003.

[68] P. Key and L. Massoulie, "Fluid models of integrated traffic and multipath routing," *Queuing Syst.*, vol. 53, no. 1, pp. 85–98, Jun. 2006.

[69] P. Key, L. Massoulie, and D. Towsley, "Combining multipath routing and congestion control for robustness," in *Proc. CISS*, Mar. 2006.

[70] R. R. Kompella, A. Greenberg, J. Rexford, A. C. Snoeren, and J. Yates, "Cross-layer visibility as a service," in *Proc. ACM HotNets Workshop*, Nov. 2005.

[71] S. Kunniyur and R. Srikant, "A time-scale decomposition approach to adaptive explicit congestion notification (ECN) marking," *IEEE Trans. Autom. Control,* vol. 47, no. 6, pp. 882–894, Jun. 2002.

[72] ——, "End-to-end congestion control: Utility functions, random losses and ECN marks," *IEEE/ACM Trans. Netw.*, vol. 11, no. 5, pp. 689–702, Oct. 2003.

[73] R. J. La and V. Anantharam, "Utility-based rate control in the Internet for elastic traffic," *IEEE/ACM Trans. Netw.*, vol. 10, no. 2, pp. 272–286, Apr. 2002.

[74] J. W. Lee, M. Chiang, and R. A. Calderbank, "Jointly optimal congestion and contention control in wireless ad hoc networks," *IEEE Commun. Lett.*, vol. 10, no. 3, pp. 216–218, Mar. 2006.

[75] ——, "Price-based distributed algorithm for optimal rate-reliability tradeoff in network utility maximization," *IEEE J. Sel. Areas Commun.*, vol. 24, no. 5, pp. 962–976, May 2006.

[76] ——, "Utility-optimal medium access control," *IEEE Trans. Wireless Commun.*, 2007, to be published.

[77] J. W. Lee, R. R. Mazumdar, and N. Shroff, "Non-convex optimization and rate control for multi-class services in the Internet," *IEEE/ACM Trans. Netw.*, vol. 13, no. 4, pp. 827–840, Aug. 2005.

[78] J. W. Lee, A. Tang, J. Huang, M. Chiang, and A. R. Calderbank, "Reverse engineering MAC: A game theoretic model," *IEEE J. Sel. Areas Comm.*, 2007, to be published.

[79] Y. Li, M. Chiang, A. R. Calderbank, and S. Diggavi, "Optimal delay-rate-reliability tradeoff in networks with composite links," in *Proc. IEEE INFOCOM*, May 2007.

[80] X. Lin and S. Rasool, "Constant time distributed scheduling policies for ad hoc wireless networks," in *Proc. IEEE CDC*, Dec. 2006.

[81] X. Lin and N. B. Shroff, "On the stability region of congestion control," in *Proc. Allerton Conf.*, Oct. 2004.

[82] ——, "Joint rate control and scheduling in multihop wireless networks," in *Proc. IEEE CDC*, Dec. 14–17, 2004, vol. 2, pp. 1484–1489.

[83] ——, "The impact of imperfect scheduling on cross-layer rate control in wireless networks," *IEEE/ACM Trans. Netw.*, vol. 14, no. 2, pp. 302–315, Apr. 2006.

[84] ——, "Utility maximization for communication networks with multipath routing," *IEEE Trans. Autom. Control,* vol. 51, no. 5, pp. 766–781, May 2006.

[85] X. Lin, N. B. Shroff, and R. Srikant, "A tutorial on cross-layer design in wireless networks," *IEEE J. Sel. Areas Commun.*, vol. 24, no. 8, pp. 1452–1463, Aug. 2006.

[86] D. S. Lun, N. Ratnakar, M. Mdard, R. Koetter, D. R. Karger, T. Ho, and E. Ahmed, "Minimum-cost multicast over coded packet networks," *IEEE Trans. Inform. Theory,* vol. 52, no. 6, pp. 2608–2623, Jun. 2006.

[87] S. H. Low, "A duality model of TCP and queue management algorithms," *IEEE/ACM Trans. Netw.*, vol. 11, no. 4, pp. 525–536, Aug. 2003.

[88] S. H. Low, J. Doyle, and F. Paganini, "Internet congestion control," *IEEE Control Syst. Mag.*, vol. 21, no. 1, pp. 28–43, Feb. 2002.

[89] S. H. Low, L. L. Perterson, and L. Wang, "Understanding Vegas: A duality model," *J. ACM,* vol. 49, no. 2, pp. 207–235, Mar. 2002.

[90] S. H. Low and D. E. Lapsley, "Optimization flow control, I: Basic algorithm and convergence," *IEEE/ACM Trans. Netw.*, vol. 7, no. 6, pp. 861–874, Dec. 1999.

[91] S. H. Low and R. Srikant, "A mathematical framework for designing a low-loss, low-delay Internet," *Special Issue on Crossovers Between Transportation Planning and Telecommunications, Netw. Spatial Econ.*, vol. 4, pp. 75–101, Mar. 2004.

[92] P. Marbach and Y. Lu, "Active queue management and scheduling for wireless networks: The single cell case," in *Proc. CISS*, Mar. 2006.

[93] L. Massoulie, "Structural properties of proportional fairness: Stability and insensitivity," *Preprint*, 2006.

[94] L. Massoulie and J. W. Roberts, "Bandwidth sharing and admission control for elastic traffic," *Telecommun. Syst.*, vol. 15, pp. 185–201, Mar. 2000.

[95] M. Mehyar, D. Spanos, and S. H. Low, "Optimization flow control with estimation error," in *Proc. IEEE INFOCOM*, Mar. 7–11, 2004, vol. 2, pp. 984–992.

[96] J. Mo and J. Walrand, "Fair end-to-end window-based congestion control," *IEEE/ACM Trans. Netw.*, vol. 8, no. 5, pp. 556–567, Oct. 2000.

[97] H. Nama, M. Chiang, and N. Mandayam, "Utility lifetime tradeoff in self regulating wireless sensor networks: A cross-layer design approach," in *Proc. IEEE ICC*, Jun. 2006.

[98] T. Nandagopal, T. Kim, X. Gao, and V. Bharghavan, "Achieving MAC layer fairness in wireless packet networks," in *Proc. ACM MOBICOM*, Aug. 2000.

[99] M. J. Neely, E. Modiano, and C. P. Li, "Fairness and optimal stochastic control for heterogeneous networks," in *Proc. IEEE INFOCOM*, Mar. 13–17, 2005, vol. 3, pp. 1723–1734.

[100] M. J. Neely, E. Modiano, and C. E. Rohrs, "Dynamic power allocation and routing time varying wireless networks," *IEEE J. Sel. Areas Commun.*, vol. 23, no. 1, pp. 89–103, Jan. 2005.

[101] F. Paganini, "Congestion control with adaptive multipath routing based on optimization," in *Proc. CISS*, Mar. 2006.

[102] F. Paganini, J. C. Doyle, and S. H. Low, "Scalable laws for stable network congestion control," in *Proc. IEEE CDC*, Dec. 4–7, 2001, vol. 1, pp. 185–190.

[103] F. Paganini, Z. Wang, J. C. Doyle, and S. H. Low, "Congestion control for high performance, stability and fairness in general networks," *IEEE/ACM Trans. Netw.*, vol. 13, no. 1, pp. 43–56, Feb. 2005.

[104] D. Palomar and M. Chiang, "Alternative decompositions for distributed maximization of network utility: Framework and applications," *IEEE Trans. Autom. Control*, 2007, to be published.

[105] ——, "A tutorial to decompositon methods for network utility maximization," *IEEE J. Sel. Areas Commun.*, vol. 24, no. 8, pp. 1439–1450, Aug. 2006.

[106] A. Papachristodoulou, "Global stability analysis of a TCP/AQM protocol for arbitrary networks with delay," in *Proc. 43rd IEEE CDC*, Dec. 14–17, 2004, pp. 1029–1034.

[107] A. Papachristodoulou, L. Li, and J. C. Doyle, "Methodological frameworks for large-scale network analysis and design," *Comput. Commun. Rev.*, pp. 7–20, Jul. 2004.

[108] P. A. Parrilo, "Semidefinite programming relaxations for semialgebraic problems," *Math. Programming Ser. B,* vol. 96, no. 2, pp. 293–320, 2003.

[109] S. Prajna, A. Papachristodoulou, P. Seiler, and P. A. Parrilo, *SOSTOOLS: Sum of Squares Optimization Toolbox for MATLAB User's Guide*, Jun. 2004.

[110] I. C. Paschalidis and J. N. Tsitsiklis, "Congestion-dependent pricing of network services," *IEEE/ACM Trans. Netw.*, vol. 8, no. 2, pp. 171–184, Apr. 2000.

[111] I. C. Paschalidis and Y. Liu, "Pricing in multi-service loss networks: Static pricing, asymptotic optimality, and demand substitution effects," *IEEE/ACM Trans. Netw.*, vol. 10, no. 3, pp. 425–438, Jun. 2002.

[112] J. Ponsajapan and S. H. Low, "Reverse engineering TCP/IP-like networks using delay-sensitive utility functions," in *Proc. IEEE INFOCOM*, Anchorage, Alaska, May 2007.

[113] G. Raina and D. Wischik, "Buffer sizes for large multiplexers: TCP queuing theory and stability analysis," in *Proc. Next Generation Internet Networks*, 2005.

[114] S. Ramanathan, "A unified framework and algorithms for channel assignment in wireless networks," *Wireless Networks,* vol. 5, no. 2, pp. 81–94, 1999.

[115] P. Ranjan, R. J. La, and E. H. Abed, "Characterization of global stability conditions with an arbitrary communication delay," *IEEE/ACM Trans. Netw.*, vol. 14, no. 2, pp. 94–107, Apr. 2006.

[116] J. Roberts and L. Massoulie, "Bandwidth sharing and admission control for elastic traffic," *Telecommun. Syst.*, vol. 15, pp. 185–201, 2000.

[117] R. T. Rockafellar, *Network Flows and Monotropic Programming*. Belmont, MA: Athena Scientific, 1998.

[118] ——, "Saddle-points and convex analysis," in *Differential Games and Related Topics*, H. W. Kuhn and G. P. Szego, Eds. Amsterdam, The Netherlands: North-Holland, 1971.

[119] S. Shakkottai and R. Srikant, "Mean FDE models for Internet congestion control under a many-flows regime," *IEEE Trans. Inf. Theory*, vol. 50, no. 6, pp. 1050–1072, Jun. 2004.

[120] C. E. Shannon, "A mathematical theory of communication," *Bell Syst.Tech. J.*, vol. 27, pp. 379–423/623–656, 1948.

[121] G. Sharma, R. R. Mazumdar, and N. B. Shroff, "On the complexity of scheduling in wireless networks," in *Proc. ACM MOBICOM*, 2006.

[122] S. Shenker, "Fundamental design issues for the future Internet," *IEEE J. Sel. Areas Commun.*, vol. 13, no. 7, pp. 1176–1188, Sep. 1995.

[123] R. N. Shorten and D. J. Leith, "H-TCP: TCP for high-speed and long-distance networks," in *Proc. PFLDnet*, 2004.

[124] A. Simsek, A. Ozdaglar, and D. Acemoglu, "Generalized Poincare–Hopf theorem for compact non-smooth regions," *Preprint*, 2006.

[125] R. Srikant, *The Mathematics of Internet Congestion Control*. Cambridge, MA: Birkhauser, 2004.

[126] ——, "On the positive recurrence of a Markov chain describing file arrivals and departures in a congestion-controlled network," in *IEEE Comput. Commun. Workshop*, Oct. 2004.

[127] W. Stevens, *TCP/IP Illustrated: The Protocols*. Reading, MA: Addison-Wesley, 1999, vol. 1.

[128] A. L. Stolyar, "On the asymptotic optimality of the gradient scheduling algorithm for multi-user throughput allocation," *Operations Res.*, vol. 53, no. 1, pp. 12–25, Jan. 2005.

[129] ——, "Maximizing queueing network utility subject to statbility: Greedy primal-dual algorithm," *Queueing Syst.*, vol. 50, no. 4, pp. 401–457, 2005.

[130] C. W. Tan, D. Palomar, and M. Chiang, "Distributed optimization of coupled systems with applications to network utility maximization," in *Proc. IEEE ICASSP*, May 14–19, 2006, vol. 5, pp. V-981–V-984.

[131] C. W. Tan, D. Palomar, and M. Chiang, "Exploiting hidden convexity for flexible and robust resource allocation in cellular networks," in *Proc. IEEE INFOCOM*, May 2007.

[132] A. Tang, J. Wang, and S. H. Low, "Counter-intuitive throughput behavior in networks under end-to-end control," *IEEE/ACM Trans. Netw.*, vol. 14, no. 2, pp. 355–468, Apr. 2006.

[133] A. Tang, J. Wang, S. Hegde, and S. H. Low, "Equilibrium and fairness of networks shared by TCP Reno and FAST," *Telecommun. Syst.* vol. 30, no. 4, pp. 417–439, 2005.

[134] A. Tang, J. Wang, S. H. Low, and M. Chiang, "Network equilibrium of heterogeneous congestion control protocols," *IEEE/ACM Trans. Netw.*, Oct. 2007.

[135] A. Tang, D. Wei, S. H. Low, and M. Chiang, "Heterogeneous congestion control: Efficiency, fairness, and control," in *Proc. IEEE ICNP*, Nov. 2006.

[136] S. W. Tam, D. M. Chiu, J. C. S. Lui, and Y. C. Tay, "A case for TCP-friendly admission control," in *Proc. IEEE IWQoS*, 2006.

[137] P. Tinnakornsrisuphap and R. J. La, "Characterization of queue fluctuations in probabilistic AQM mechanisms," in *Proc. ACM SIGMETRICS*, Jun. 2004, pp. 229–238.

[138] L. Tassiulas and A. Ephremides, "Stability properties of constrained queueing systems and scheduling policies for maximum throughput in multihop radio networks," *IEEE Trans. Autom. Control*, vol. 36, no. 12, pp. 1936–1948, Dec. 1992.

[139] P. Tinnakornsrisuphap and A. M. Makowski, "Limit behavior of ECN/RED gateways under a large number of TCP flows," in *Proc. IEEE INFOCOM*, Mar. 2003.

[140] G. Vinnicombe, "On the stability of end-to-end congestion control for the internet," Cambridge Univ., Tech. Rep. CUED/F-INFENG/TR.398, Dec. 2000.

[141] ——, "On the stability of networks operating TCP-like congestion control," in *Proc. IFAC World Congress*, 2002.

[142] T. Voice, "A global stability result for primal-dual congestion control algorithms with routing," *Comput. Commun. Rev.*, vol. 34, no. 3, pp. 35–41, 2004.

[143] X. Wang and K. Kar, "Cross-layer rate optimization for proportional fairness in multihop wireless networks with random access," *IEEE J. Sel. Areas Commun.*, vol. 24, no. 8, pp. 1548–1559, Aug. 2006.

[144] J. Wang, L. Li, S. H. Low, and J. C. Doyle, "Cross-layer optimization in TCP/IP networks," *IEEE/ACM Trans. Netw.*, vol. 13, no. 3, pp. 582–268, Jun. 2005.

[145] J. Wang, D. X. Wei, J.-Y. Choi, and S. H. Low, "Modeling and stability of FAST TCP," in *IMA Volumes in Mathematics and Its Applications*, Wireless Communications, P. Agrawal, M. Andrews, P. J. Fleming, G. Yin, and L. Zhang, Eds. New York: Springer Science, vol. 143, 2006.

[146] J. Wang, D. X. Wei, and S. H. Low, "Modeling and stability of FAST TCP," in *Proc. IEEE INFOCOM*, Mar. 13–17, 2005, vol. 2, pp. 938–948.

[147] D. X. Wei, C. Jin, S. H. Low, and S. Hegde, "FAST TCP: Motivation, architecture, algorithms, and performance," *IEEE/ACM Trans. Netw.*, vol. 14, no. 6, pp. 1246–1259, Dec. 2006.

[148] J. T. Wen and M. Arcak, "A unifying passivity framework for network flow control," *IEEE Trans. Autom. Control*, vol. 49, no. 2, pp. 162–174, Feb. 2004.

[149] X. Wu and R. Srikant, "Regulated maximal matching: A distributed scheduling algorithm for multi-hop wireless networks with node exclusive spectrum sharing," in *Proc. IEEE CDC*, Dec. 12–15, 2005, pp. 5342–5347.

[150] ——, "Bounds on the capacity region of multihop wireless newtorks under distributed greedy scheduling," in *Proc. IEEE INFOCOM*, Apr. 2006.

[151] Y. Wu and S. Y. Kung, "Distributed utility maximization for network coding based multicasting: A shortest path approach," *IEEE J. Sel. Areas Commun.*, vol. 24, no. 8, pp. 1475–1488, Aug. 2006.

[152] Y. Wu, M. Chiang, and S. Y. Kung, "Distributed utility maximization for network coding based multicasting: A critical cut approach," in *Proc. IEEE NetCod Workshop*, Apr. 3–6, 2006.

[153] B. Wydrowski and M. Zukerman, "MaxNet: A congestion control architecture for maxmin fairness," *IEEE Commun. Lett.* vol. 6, no. 11, pp. 512–514, Nov. 2002.

[154] B. Wydrowski, L. L. H. Andrew, and M. Zukerman, "MaxNet: A congestion control architecture for scalable networks," *IEEE Commun. Lett.*, vol. 7, no. 10, pp. 511–513, Oct. 2003.

[155] L. Xu, K. Harfoush, and I. Rhee, "Binary increase congestion control (BIC) for fast long distance networks," in *IEEE Proc. INFOCOM*, Mar. 7–11, 2004, vol. 4, pp. 2514–2524.

[156] Y. Xi and E. Yeh, "Node-based distributed optimal control of wireless networks," in *Proc. CISS*, Mar. 2006.

[157] ——, "Distributed algorithms for minimum cost multicast with network coding," in *Proc. IEEE NetCod Workshop*, Apr. 3–6, 2006.

[158] L. Xiao, M. Johansson, and S. Boyd, "Joint routing and resource allocation via dual decomposition," *IEEE Trans. Commun.*, vol. 52, no. 7, pp. 1136–1144, Jul. 2004.

[159] D. Xu, M. Chiang, and J. Rexford, "DEFT: Distributed exponentially-weighted flow splitting," in *Proc. IEEE INFOCOM*, May 2007.

[160] D. Xu, Y. Li, M. Chiang, and A. R. Calderbank, "Optimal provisioning of elastic service availability," in *Proc. IEEE INFOCOM*, May 2007.

[161] H. Yäiche, R. R. Mazumdar, and C. Rosenberg, "A game theoretic framework for bandwidth allocation and pricing of elastic connections in broadband networks: Theory and algorithms," *IEEE/ACM Trans. Netw.*, vol. 8, no. 5, pp. 667–678, Oct. 2000.

[162] H. Ye, "Stability of data networks under optimization-based bandwidth allocation," *IEEE Trans. Autom. Control*, vol. 48, no. 7, pp. 1238–1242, Jul. 2003.

[163] H. Ye, J. Qu, and X. Yuan, "Stability of data networks: Starionary and bursty models," *Oper. Res.*, vol. 53, pp. 107–125, 2005.

[164] Y. Yi, S. Deb, and S. Shakkottai, "Timescale decomposition and rate-based marking," *IEEE/ACM Trans. Netw.*, vol. 14, no. 5, pp. 938–950, 2006.

[165] Y. Yi, G. de Veciana, and S. Shakkottai, "Learning contention patterns and adapting to load topology changes in MAC scheduling algorithms," in *Proc. IEEE Workshop on Wireless Mesh Networks*, Sep. 2006.

[166] Y. Yi and S. Shakkottai, "On the elasticity of marking functions: Scheduling, stability, quality-of-service in the Internet," in *Proc. CISS*, Mar. 2005.

[167] W. Yu and J. Yuan, "Joint source coding, routing, and resource allocation for wireless sensor networks," in *Proc. IEEE ICC*, May 2005.

[168] C. Yuen and P. Marbach, "Price-based rate control in random access networks," *IEEE/ACM Trans. Netw.*, vol. 13, no. 5, pp. 1027–1040, Dec. 2005.

[169] J. Zhang and D. Zheng, "A stochastic primal-dual algorithm for joint flow control and MAC design in multihop wireless networks," in *Proc. CISS*, Mar. 2006.

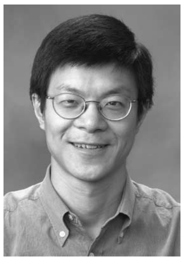
[170] J. Zhang, D. Zheng, and M. Chiang, "Impacts of stochastic noisy feedback in network utility maximization," *Proc. IEEE INFOCOM*, May 2007.

ABOUT THE AUTHORS

Mung Chiang (Member, IEEE) received the B.S. (Hon.) degree in electrical engineering and in mathematics, and the M.S. and Ph.D. degrees in electrical engineering from Stanford University, Stanford, CA, in 1999, 2000, and 2003, respectively.

He is an Assistant Professor of Electrical Engineering, and an affiliated faculty member of the Program in Applied and Computational Mathematics at Princeton University, Princeton, NJ. He conducts research in the areas of optimization of communication systems, theoretical foundations of network architectures, algorithms in broadband access networks, and stochastic models of communications.

Dr. Chiang has been awarded a Hertz Foundation Fellowship, and received the Stanford University School of Engineering Terman Award for Academic Excellence, the SBC Communications New Technology Introduction contribution award, the National Science Foundation CAREER Award, and the Princeton University Howard B. Wentz Junior Faculty Award. He is the Lead Guest Editor of the IEEE Journal of Selected Areas in Communications, Special Issue on Nonlinear Optimization of Communication Systems, a Guest Editor of the IEEE Transactions on Information Theory and IEEE/ACM Transactions on Networking, Joint Special Issue on Networking and Information Theory, an Editor of IEEE Transactions on Wireless Communications, the Program Co-Chair of the 38th Conference on Information Sciences and Systems, and a co-editor of Springer book series on Control and Optimization of Communication Systems. He is a co-author of IEEE GLOBECOM Best Student Paper Award, and one of his paper becomes the Fast Breaking Paper in Computer Science in 2006 by ISI citation data.



Steven H. Low (Senior Member, IEEE) received the B.S. degree from Cornell University, Ithaca, NY, and the Ph.D. degree from the University of California, Berkeley, both in electrical engineering.

He is a Professor of Computer Science and Electrical Engineering at California Institute of Technology (Caltech), Pasadena. He was with AT&T Bell Laboratories, Murray Hill, NJ, from 1992 to 1996, the University of Melbourne, Australia, from 1996 to 2000, and was a Senior Fellow of the University of Melbourne from 2000 to 2004. He is a member of the Networking and Information Technology Technical Advisory Group for the U.S. President's Council of Advisors on Science and Technology (PCAST). His interests are in the control and optimization of networks and protocols.

Dr. Low was a co-recipient of the IEEE William R. Bennett Prize Paper Award in 1997 and the 1996 R&D 100 Award. He was on the editorial board of IEEE/ACM Transactions on Networking from 1997 to 2006 and on that of *Computer Networks Journal* from 2003 to 2005. He is on the editorial boards of *ACM Computing Surveys*, *NOW Foundations*, and *Trends in Networking*, and is a Senior Editor of the IEEE Journal on Selected Areas in Communications.

A. Robert Calderbank (Fellow, IEEE) is a Professor of Electrical Engineering and Mathematics at Princeton University, Princeton, NJ, where he directs the Program in Applied and Computational Mathematics. He joined Bell Laboratories as a member of Technical Staff in 1980, and retired from AT&T in 2003 as Vice President of Research. He has research interests that range from algebraic coding theory and quantum computing to the design of wireless and radar systems.

Dr. Calderbank served as Editor in Chief of the IEEE Transactions on Information Theory from 1995 to 1998, and as Associate Editor for Coding Techniques from 1986 to 1989. He was a member of the Board of Governors of the IEEE Information Theory Society from 1991 to 1996. He was honored by the IEEE Information Theory Prize Paper Award in 1995 for his work on the Z4 linearity of Kerdock and Preparata Codes (joint with A. R. Hammons Jr., P. V. Kumar, N. J. A. Sloane, and P. Sole), and again in 1999 for the invention of space–time codes (joint with V. Tarokh and N. Seshadri). He is a recipient of the IEEE Millennium Medal, and was elected to the National Academy of Engineering in 2005.

John C. Doyle received the B.S. and M.S. degrees in electrical engineering from Massachusetts Institute of Technology in 1977 and the Ph.D. degree in mathematics from the University of California, Berkeley, in 1984.

He is the John G. Braun Professor of Control and Dynamical Systems, Electrical Engineering, and Bio Engineering at California Institute of Technology (Caltech), Pasadena. His early work was in the mathematics of robust control, LQG robustness, (structured) singular value analysis, H-infinity plus recent extensions. He coauthored books and software toolboxes currently used at over 1000 sites worldwide, the main control analysis tool for high performance commercial and military aerospace systems, as well as many other industrial systems. Early example industrial applications include X-29, F-16XL, F-15 SMTP, B-1, B-2, 757, Shuttle Orbiter, electric power generation, distillation, catalytic reactors, backhoe slope-finishing, active suspension, and CD players. Current research interests are in theoretical foundations for complex networks in engineering, biology, and multiscale physics.

Dr. Doyle's group led the development of the open source Systems Biology Markup Language (SBML) and the Systems Biology Workbench (SBW), the analysis toolbox SOSTOOLS, and contributed to the theory of the FAST protocol that shattered multiple world land speed records. Prize papers include the IEEE Baker, the IEEE Automatic Control Transactions Axelby (twice), and the AACC Schuck. Individual awards include the AACC Eckman and the IEEE Control Systems Field and Centennial Outstanding Young Engineer Awards. He has held national and world records and championships in various sports.