Layering As Optimization Decomposition: A Mathematical Theory of Network Architectures

Mung Chiang
Electrical Engineering Department, Princeton

Connections II, Caltech
August 16, 2006

Nature of the Talk

- Give an overview of the topic. Details in various papers
- Not exhaustive survey. Highlight the key ideas and challenges
- Biased presentation. Focus on work by us

M. Chiang, S. H. Low, A. R. Calderbank, and J. C. Doyle, "Layering as optimization decomposition: A mathematical theory of network architectures" *Proceedings of IEEE*, January 2007.

Preprint and (3 hour) tutorial slides available online

Outline

- Overview: Holistic view on layered architecture

NUM and G.NUM

- Clarifications and Examples

- Summary of Recent Activities

Horizontal Decompositions

Vertical Decompositions

Alternative Decompositions

- Key Messages and Methodologies

- Future Research and Open Issues

Collaborators On This Topic

Grateful to all the joint work and discussions

Basic framework: Rob Calderbank, Lijun Chen, John Doyle, Koushik Kar, Jang-Won Lee, Ying Li, Steven Low, Daniel Palomar, Xin Wang

Nonconvex optimization: Maryam Fazel, Prashanth Hande, David Gao, Pablo Parrilo, Asuman Ozdaglar, Chee Wei Tan

Stochastic NUM: Jianwei Huang, Jiaping Liu, Devavrat Shah, Ao Tang, Junshan Zhang

Automating decomposition: Stephen Boyd, Ma'ayan Bresler, Neil Gershenfeld

Network X-ities: Jiayue He, Jennifer Rexford, Dahai Xu

General discussion: Dah Ming Chiu, Bob Fry, Andrea Goldsmith, Frank Kelly, P.R. Kumar, Ruby Lee, Xiaojun Lin, Sanjay Shakkottai, Ness Shroff, R. Srikant, Lin Xiao

Overview

Layered Network Architecture

Application
Presentation
Session
Transport
Network
Link
Physical

Important foundation for data networking

Ad hoc design historically (within and across layers)

Rethinking Layering

How to, and how not to, layer? A question on architecture

Functionality allocation: who does what and how to connect them?

More fuzzy (and important) question than just resource allocation

But want answers to be rigorous, quantitative, simple, and relevant

- How to modularize (and connect)? How to distribute (and connect)?
- How to search in the design space of alternative architectures?
- How to quantify the benefits of better codes/modulation/schedule/routes... for network applications?

A common language to rethink these issues?

The Goal

A Mathematical Theory of Network Architectures

Layering As Optimization Decomposition

The first unifying view and systematic approach

Network: Generalized NUM

Layering architecture: Decomposition scheme

Layers: Decomposed subproblems

Interfaces: Functions of primal or dual variables

Horizontal and vertical decompositions through

- implicit message passing (e.g., queuing delay, SIR)
- explicit message passing (local or global)

3 Steps: G.NUM $\Rightarrow$ A solution architecture $\Rightarrow$ Alternative architectures

Network Utility Maximization

Basic NUM (KellyMaulloTan98):

$$\begin{array}{ll} \text{maximize} & \sum_s U_s(x_s) \\ \text{subject to} & \mathbf{R}\mathbf{x} \preceq \mathbf{c} \\ & \mathbf{x} \succeq 0 \end{array}$$

Generalized NUM (one possibility shown here) (Chiang05a):

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

10 Questions About the Framework

Let's be very skeptical and critical first:

- Isn't this just cross-layer? Does it really solve architectural issues?
- How to pick utility functions?
- What about delay, jitter, energy? Can it guarantee QoS?
- Isn't it just all about dual decomposition?
- How do you know which decomposition to pick?
- But many problem formulations are nonconvex optimization?
- Infinite backlog/buffer and fluid model don't make sense?
- Is anyone actually going to use this (too much message passing)?
- Who cares about convergence at time infinity with a weird stepsize?
- Why should network operator optimize performance in the first place?

Answers to some of the above questions in this talk

Two Cornerstones for Conceptual Simplicity

Networks as optimizers

Reverse engineering mentality: give me the solution (an existing protocol), I'll find the underlying problem implicitly being solved

- Why care about the problem if there's already a solution?
- It leads to simple, rigorous understanding for systematic design

Layering as decomposition

1. Analytic foundation for network architecture
2. Common language for thinking and comparing
3. Methodologies, analytic tools

Layering as Optimization Decomposition

What's so unique about this particular framework for cross-layer design?

- Network as optimizer
- End-user application utilities as the driver
- Performance benchmark without any layering
- Unified approach to cross-layer design (it simplifies our understanding about network architecture)
- Separation theorem among modules
- Systematic exploration of architectural alternatives

Not every cross-layer paper is 'layering as optimization decomposition'

Can Architecture Be Mathematically Understood?

- Particular focus on the architectures of modularization (layering) and distributed control
- There are also limitations of the use of mathematical approach to the economics, psychology, and engineering of network architectures
- But the right angle certainly provides rigorous approaches on why protocols work, when it will not work, and how to make it work better
- Also provides conceptually clear understanding on the opportunities and risks of cross layer design

Clarifying the Framework

GNUM Formulation

- Objective function: What the end-users and network provider care about

Can be a function of throughput, delay, jitter, energy, congestion...

Can be coupled, eg, network lifetime

- Constraint set: Physical and economic limitations. Hard QoS constraints (what the users and operator must have)

- Variables: What're under the control of this design

- Constants: What're beyond the control of this design

Utility

Which utility? Five ways of defining utility functions:

1. Reverse engineering: TCP maximizes utilities

2a. Behavioral model: user satisfaction

2b. Traffic model: traffic elasticity

3a. Economics: resource allocation efficiency

3b. Economics: different utility functions lead to different fairness

Three choices: Weighted sum, Pareto optimality, Uncooperative game

- Goal: Distributed and modularized algorithm converging to globally and jointly optimum resource allocation

- Limitations to be discussed at the end

Layers

Insights on both:

- What each layer can do (Optimization variables)
- What each layer can see (Constants, Other subproblems' variables)

Restriction: we focus on resource allocation functionalities rather than semantics functionalities

- TCP: congestion control (but not session establishment)

Each word has Different meanings:

- Routing: RIP/OSPF, BGP, wireless routing, optical routing, dynamic/static, single-path/multi-path, multicommodity flow routing...
- MAC: scheduling or contention-based
- PHY: power control, coding, modulation, antenna signal processing...

Connections With Mathematics

- Convex and nonconvex optimization
- Decomposition and distributed algorithm
- Feedback control theory
- Game theory, General market equilibrium theory

- Algebraic geometry (nonconvex formulations)
- Differential topology (heterogeneous protocols)

Adoption By Industry

Industry adoption of Layering As Optimization Decomposition:

• Internet resource allocation: TCP FAST (Caltech)

• Wired broadband access: FAST Copper (Princeton, Stanford, Fraser)

• Wireless broadband access: Cellular networks (Flarion Qualcomm, Princeton)

This talk is mainly about the underlying common language and methodologies

Illustrating Basic Decomposition Methods

Network Utility Maximization

BNUM (KellyMaulloTan98):

$$\begin{array}{ll} \text{maximize} & \sum_s U_s(x_s) \\ \text{subject to} & \mathbf{R}\mathbf{x} \preceq \mathbf{c} \\ & \mathbf{x} \succeq 0 \end{array}$$

GNUM (one possibility shown here) (Chiang05a):

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

Dual-based Distributed Algorithm

BNUM with concave smooth utility functions:

Convex optimization (Monotropic Programming) with zero duality gap

Lagrangian decomposition:

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

Dual problem:

$$
\begin{array}{ll}
\text{minimize} & g(\boldsymbol{\lambda}) = L(\mathbf{x}^*(\boldsymbol{\lambda}), \boldsymbol{\lambda}) \\
\text{subject to} & \boldsymbol{\lambda} \succeq 0
\end{array}
$$

Dual-based Distributed Algorithm

Source algorithm:

$$x_s^*(\lambda^s) = \text{argmax}\left[U_s(x_s) - \lambda^s x_s\right], \quad \forall s$$

- Selfish net utility maximization locally at source s

Link algorithm (gradient or subgradient based):

$$\lambda_l(t+1) = \left[\lambda_l(t) - \alpha(t)\left(c_l - \sum_{s:l\in L(s)} x_s(\lambda^s(t))\right)\right]^+, \quad \forall l$$

- Balancing supply and demand through pricing

Certain choices of step sizes $\alpha(t)$ of distributed algorithm guarantee convergence to globally optimal $(\mathbf{x}^*, \boldsymbol{\lambda}^*)$

Primal-Dual

Different meanings:

- Primal-dual interior-point algorithm
- Primal-dual solution
- Primal or dual driven control
- Primal penalty function approach
- Primal or dual decomposition

Coupling in constraints (easy: flow constraint, hard: SIR feasibility)

Coupling in objective (easy: additive form, hard: min max operations)

Primal Decomposition

Simple example:

$$x + y + z + w \le c$$

Decomposed into:

$$\begin{aligned} x + y &\le \alpha \\ z + w &\le c - \alpha \end{aligned}$$

New variable α updated by various methods

Interpretation: Direct resource allocation (not pricing-based control)

Engineering implications: Adaptive slicing (GENI)

Pricing feedback: dual decomposition

Adaptive slicing: primal decomposition

Where Do We Stand Now

Horizontal Decompositions

Reverse engineering:

- Layer 4 TCP congestion control: Basic NUM (LowLapsley99, RobertsMassoulie99, MoWalrand00, YaicheMazumdarRosenberg00, KunniyurSrikant02, LaAnatharam02, LowPaganiniDoyle02, Low03, Srikant04...)

- Layer 4 TCP heterogeneous protocol: Nonconvex equilibrium problem (TangWangLowChiang05)

- Layer 3 IP inter-AS routing: Stable Paths Problem (GriffinSheperdWilfong02)

- Layer 2 MAC backoff contention resolution: Non-cooperative Game (LeeChiangCalderbank06a)

Forward engineering for horizontal decompositions also carried out recently

Vertical Decompositions

A partial list of work along this line:

- Jointly optimal congestion control and adaptive coding or power control (Chiang05a, LeeChiangCalderbank06b)
- Jointly optimal congestion and contention control (KarSarkarTassiulas04, ChenLowDoyle05, WangKar05, YuenMarbach05, ZhengZhang06, LeeChiangCalderbank06c)
- Jointly optimal congestion control and scheduling (ErilymazSrikant05, Stolyar05)
- Jointly optimal routing and scheduling (KodialamNandagopal03)
- Jointly optimal routing and power control (XiaoJohanssonBoyd04, NeelyModianoRohrs05)
- Jointly optimal congestion control, routing, and scheduling (LinShroff05, ChenLowChiangDoyle06)

Vertical Decompositions

- Jointly optimal routing, scheduling, and power control (CruzSanthanam03, XiYeh06)
- Jointly optimal routing, resource allocation, and source coding (YuYuan05)
- TCP/IP interactions (WangLiLowDoyle05, HeChiangRexford06) and jointly optimal congestion control and routing (KellyVoice05, Hanetal05)
- Network lifetime maximization (NamaChiangMandayam06)
- Application adaptation and congestion control/resource allocation (ChangLiu04, HuangLiChiangKatsaggelos06)

Apology, Apology, Apology for any missing reference

Fewer Publications, Not More

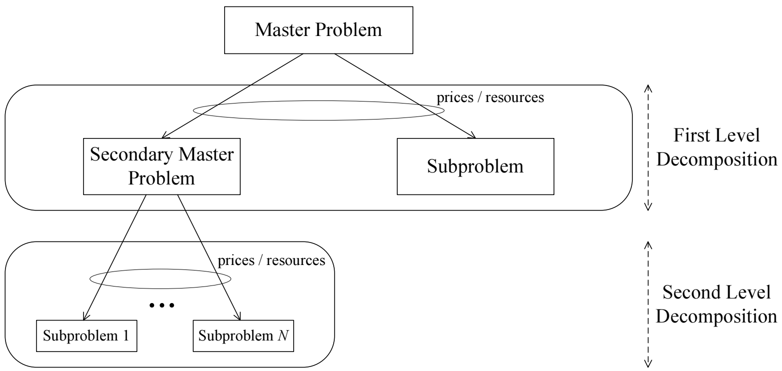
- Specific designs not important
- Common language and key messages methodologies important

Goal: Shrink, not grow knowledge tree on cross-layer design

Alternative Decompositions

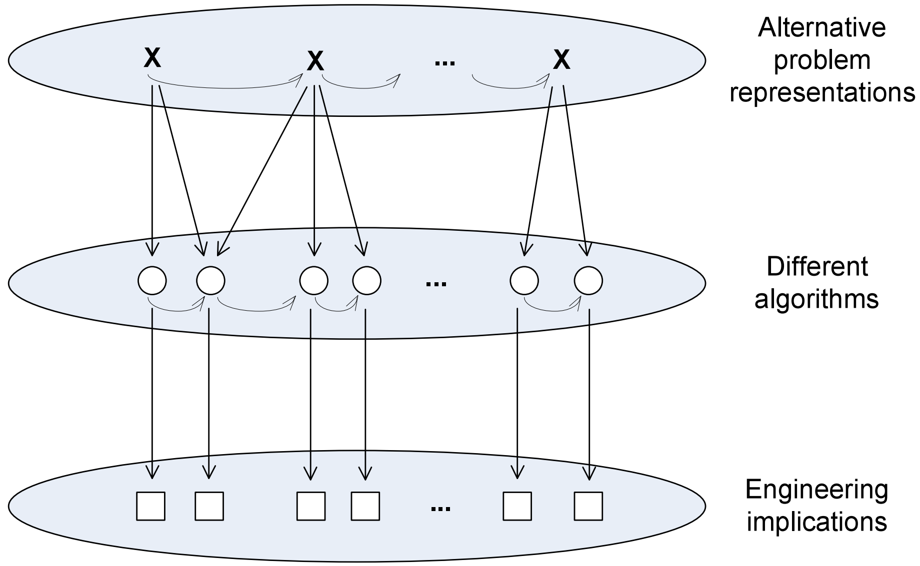
Many ways to decompose:

- Primal and dual decomposition
- Partial decomposition
- Multi-level decomposition
- Different combinations



Lead to alternative architectures (PalomarChiang06) with different engineering implications

Alternative Decompositions



Systematically explore the space of alternative decompositions

Recent Successes

Number of alternative decompositions 2005-2006:

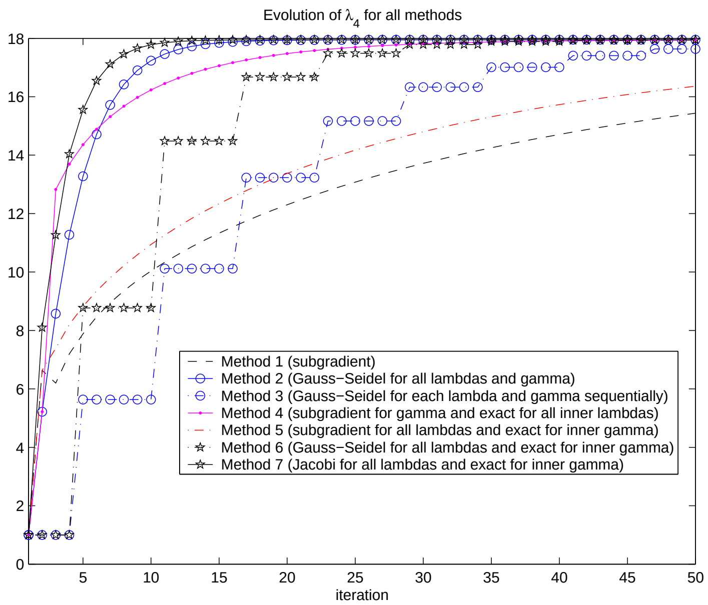
- Joint congestion control, routing, and scheduling: 5
- Joint congestion control and contention control: 3
- Joint congestion control and routing: 5
- Rate control over network coding based multicast: 4
- Horizontal decompositions: 3-7

An Example

A family of NUM formulations (PaolmarChiang06)

$$\begin{array}{ll} \text{minimize} & \sum_i U_i(x_i) \\ \text{subject to} & f_i(x_i, y_i) = 0, \ \ \forall i, \\ & \sum_i g_i(x_i, y_i) \leq 1 \end{array}$$

One of the metrics: Different speed of convergence



CAD Tool for Network Architecture

Automate the enumeration of alternative decompositions:

Challenge: Be careful about new possibilities from problem transformations that change the decomposability structure

Automate the comparison of alternative decompositions:

Challenge: Some of the following metrics are not well defined, fully quantified, or accurately characterized

- Speed of convergence
- Robustness (errors, failures, network dynamics)
- Message passing (amount, locality, symmetry)
- Local computation (amount, symmetry)
- Ease of relaxing to simpler heuristics
- Ease of modification as new applications arise

Key Messages and Methodologies

10 Key Messages

- Existing protocols in layers 2,3,4 have been reverse engineered
- Reverse engineering leads to better design
- There is one unifying approach to cross-layer design
- Loose coupling through layering price
- Queue length often a right layering price, but not always
- Many alternatives in decompositions and layering architectures
- Convexity is key to proving global optimality
- Decomposability is key to designing distributed solution
- Still many open issues in modeling, stochastic dynamics, and nonconvex formulations
- Architecture, rather than optimality, is the key

A Sample of 20 Methodologies

- Reverse engineering cooperative protocol as an optimization algorithm
- Lyapunov function construction to show stability
- Proving convergence of dual descent algorithm
- Proving stability by singular perturbation theory
- Proving stability by passivity argument
- Proving equilibrium properties through vector field representation
- Reverse engineering non-cooperative protocol as a game
- Verifying contraction mapping by bounding the Jacobian's norm
- Analyzing cross-layer interaction systematically through G.NUM
- Change of variable for decoupling, and computing minimum curvature needed

A Sample of 20 Methodologies

- Dual decomposition for jointly optimal cross layer design
- Computing conditions under which a general constraint set is convex
- Introducing an extra "layer" to decouple the problem
- End user generated pricing
- Different timescales of protocol stack interactions through different decomposition methods
- Maximum differential congestion pricing for node-based back-pressure scheduling
- Absorbing routing functionality into congestion control and scheduling
- Primal and dual decomposition for coupling constraints
- Consistency pricing for decoupling coupled objective
- Partial and hierarchical decompositions for architectural alternatives

Where Are We Going Next?

Asymptotic properties of deterministic, convex formulations of throughput utility maximization very well understood

But so much more remain under-explored

Future Research Issues

- Technical: Global stability under delay...

- Modeling: routing in ad hoc network, ARQ, MIMO...

- Time issues

- Why deterministic fluid model?

Shannon 1948: remove finite blocklength, Law of Large Numbers kicks in (later finite codewords come back...)

Kelly 1998: remove coupled queuing dynamics, optimization and decomposition view kicks in (later stochastics come back...)

- What if it's not convex optimization?

Rockafellar 1993: Convexity is the watershed between easy and hard (what if it's hard?)

- Is performance the only optimization objective?

Future Research: Time Issues

- Rate of convergence
- Timescale separation
- Transient behavior bounding
- Utility as a function of latency
- Utility as a function of transient rate allocations

Future Research: Stochastic Issues

Fill the table with 3 stars in all entries:

Union of Stochastic Network and Network Optimization

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

Table 1: State-of-the-art in Stochastic Network Utility Maximization.

With a good layering architecture:

- Stochastic doesn't hurt
- Stochastic may help

Future Research: Stochastic Issues

- Channel level stochastic

Stability and optimality (Stolyar05, ChenLowChiangDoyle06)

- Session level stochastic

Earlier result: Markov model (BonaldMaussoulie01, deVecianaLeeKonstantopoulos01, Ye03, LinShroff04, Srikant04, KellyWilliams05, KeyMassoulie06, BonaldMassoulieProutiereVirtamo06...)

Recent result: general model (Bramson06, Massoulie06, ChiangShahTang06)

- Packet level stochastic (ChangLiu04, DebShakkottaiSrikant05)

- Topology level stochastic

Future Research: Nonconvexity Issues

- Nonconcave utility (eg, real-time applications)
- Nonconvex constraints (eg, power control in low SIR)
- Integer constraints (eg, single-path routing)
- Exponentially long description length (eg, scheduling)

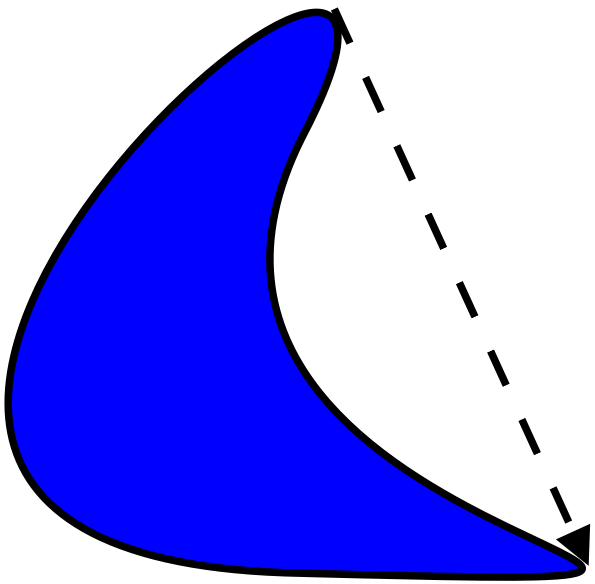
Convexity not invariant, so we can have, e.g.,

- Sum-of-squares method (Stengle73, Parrilo03, others)
- Geometric programming (DuffinPetersonZener67, Chiang05b, others)

From optimal/complicated to suboptimal/simple modules (LinShroff05)

Future Research: Nonconvexity Issues

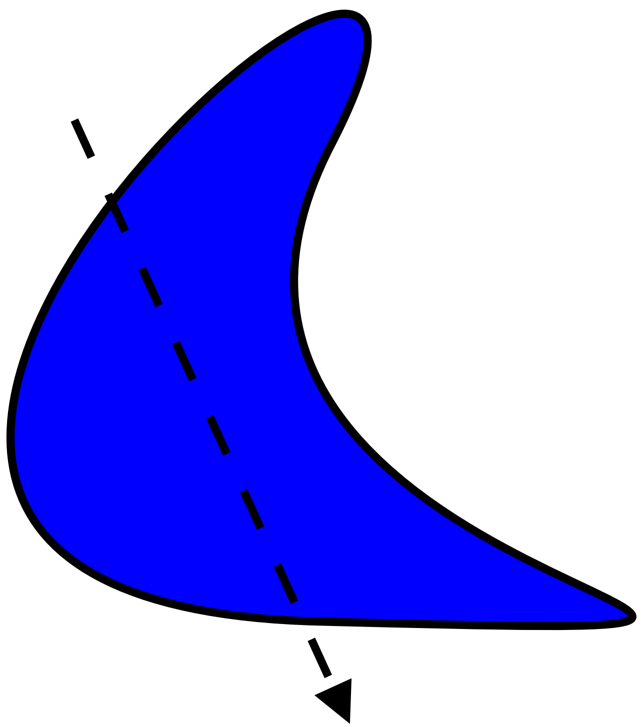
Option 1: Go around nonconvexity



- Geometric Programming, change of variable
- Sufficient condition under which the problem is convex
- Sufficient conditions for uniqueness of KKT points

Future Research: Nonconvexity Issues

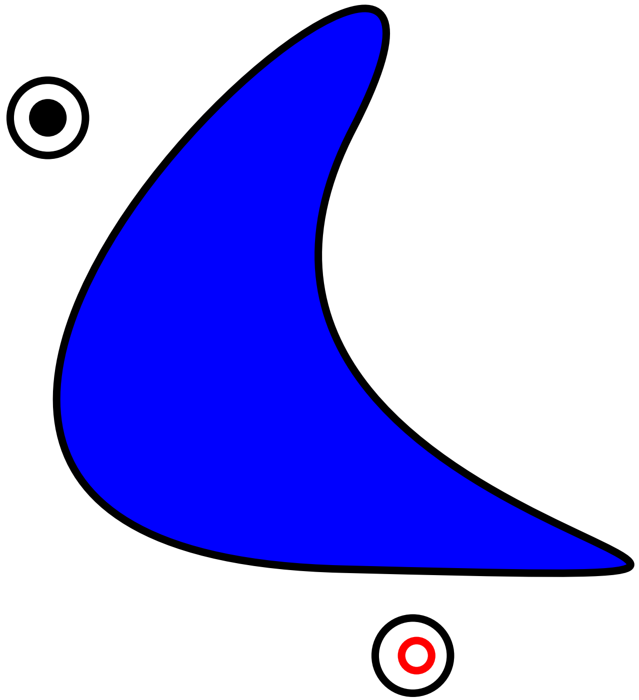
Option 2: Go through nonconvexity



- SOS, Signomial programming, successive convex approximation
- Special structure (e.g., DC, generalized quasiconcavity)
- Smart branch and bound

Future Research: Nonconvexity Issues

Option 3: Go above nonconvexity: Design for Optimizability



Change the problem, rather than solve it (HeRexfordChiang06)

- Redraw architecture or protocol to make the problem easy to solve
- Recent successes in Internet intra-domain routing

Future Research: Network X-ities Issues

From Bit to Utility to Control and Management

Over-optimized? Optimizing for what?

- Availability
- Diagnosability
- Scalability
- Evolvability

Pareto-optimal tradeoff between Performance and Network X-ities

From Forward Engineering to Reverse Engineering to

- Design for Optimizability

Research Challenges

A sample of 30 bullets in three categories

Open Problems

- Stochastic stability for general filesize distribution, general utility functions and convex constraint set, without timescale separation?
- Performance (utility, delay...) under session, channel, and packet level stochastic?
- Impacts of stochastic feedback for multi-timescale decompositions?
- Validation of fluid model from packet level dynamics?
- Global convergence of successive convex approximations for signomial programming?
- Distributed Sum-of-Squares for nonconcave NUM
- Duality gap: estimation, bounding, and implications
- Tight bound on the rate of convergence of various distributed algorithms?
- Practical stepsize rules in asynchronous networks?
- Low spatial-temporal complexity scheduling algorithm?
- Global stability under feedback delay?

Open Issues

- Constraint set of G.NUM from information theory?
- How to systematically search alternative G.NUM representations and alternative decompositions?
- Adaptive slicing by primal decompositions?
- Modeling of routing (ad hoc network and BGP)?
- Dealing with utility as functions of delay and transient resource allocations for real-time flows?
- Degree of heterogeneity and price of heterogeneity?
- Topology level stochastic?
- New notions of fairness in S.NUM?
- Quantify suboptimality's impact on fairness?
- Characterize and bound instability?
- Hardware and application modeling?

New Mentalities

- Robustness-optimality tradeoff?

- Move away from optimality?

Suboptimal (with bounded loss of optimality) and simple algorithm for each module

Good architecture contains the "damage" to the overall system

- Stochastic network dynamics is good? "Washes away" the corner cases?

- From focus on equilibrium to investigations of the transients (eg, how close to optimum within a given time? Will resource allocation during transient drop below certain thresholds?)

- How to enumerate and compare alternative architectures?

New Mentalities

- Redesign architectures (especially the division between control protocols and network management systems) for optimizability?

The Need for new architecture as a function of degree of difficulty of the problems induced by the existing architecture?

- Quantify other Network X-ities?

- Managing complexity in other types of networks through layering?

Beyond Recent Successes

Layering As Optimization Decomposition

But move away from:

- One architecture fits all (there're architecture choices)
- Deterministic fluids (good architecture works fine under stochastics)
- Asymptotic convergence (when is it good enough under a given architecture)
- Optimality (good architecture contains suboptimality damage)
- Optimization (good architecture from "design for optimizability")

So what's left?

Think about "right" decomposition in the "right" way

Contacts

chiangm@princeton.edu
www.princeton.edu/∼chiangm