Assessing the assumptions underlying mechanism design for the Internet

Steven J. Bauer
Massachusetts Institute of Technology
77 Mass Ave
Cambridge, MA
bauer@mit.edu

Peyman Faratin
Massachusetts Institute of Technology
77 Mass Ave
Cambridge, MA
peyman@mit.edu

Robert Beverly
Massachusetts Institute of Technology
77 Mass Ave
Cambridge, MA
rbeverly@mit.edu

ABSTRACT

The networking research community increasingly seeks to leverage mechanism design to create incentive mechanisms that align the interests of selfish agents with the interests of a principal designer. To apply mechanism design, a principal designer must adopt a variety of assumptions about the structure of the induced game and the agents that will be participating. (We focus in this paper on assumptions regarding agent preferences and non-repeated vs. repeated games.) As we demonstrate, such assumptions are central to understanding the degree to which theoretical claims based upon mechanism design support architectural design decisions or are useful predictors of real-world system dynamics. This understanding is central to integrating the theoretical results from mechanism design into a larger architectural discussion and engineering analysis required in networking research. We present two case studies that examine how the valid theoretical claims of [7, 18] relate to a larger, architectural discussion. We conclude with a discussion of general criteria for designing and evaluating incentive mechanisms for complex real-world networks like the Internet.

Categories and Subject Descriptors

J.4 [**Social and Behavioral Sciences**]: Economics; C.2.1 [**Network Architecture and Design**]: Packet-switching networks

General Terms

Economics, Theory

Keywords

network architecture, mechanism design, agent preferences, repeated games

1. INTRODUCTION

The networking community has often designed architectures and protocols that rely on the cooperative behaviors of participants (e.g. TCP). The field of mechanism design, though, suggests that network architectures and protocols can be designed that align the interests of non-cooperative selfish agents with the interests of a mechanism designer. This then would seem to be a powerful theory in which strong claims could be made of agent and system behaviors. However, strong claims are contingent upon many assumptions about selfish agents that may not hold in practice. Claims are much weaker if a mechanism only aligns the incentives of the subset of selfish agents that happen to match a principal's underlying assumptions.

While adopting simplifying assumptions can enable a network architect or protocol designer to prove theoretical properties such as the incentive compatibility or efficiency of a mechanism, these results may not always be useful predictors of actual agent behaviors or system dynamics when a mechanism is deployed in practice. While the simplifying assumptions of any theory are often easy to criticize from a practical perspective, the point of this paper is to not to criticize. Rather we hope to further the use of mechanism design by the networking community by promoting a better understanding of the contexts in which mechanism design succeeds (or fails) in practice to improve network and protocol designs.

While mechanism design requires simplifying assumptions – rationality, common knowledge – we focus on what mechanism designers can know about agent preferences and what they assume about whether the induced game will be a single-shot or repeated game. We focus on these assumptions because they strongly influence how applicable the theoretical results of mechanism design are in practice for the networking community.

The first class of assumptions we examine is the structure and type of agents' preferences. We consider how realistic various assumptions about agents' utilities are in practice. While the majority of agents participating in a game induced by a mechanism may match a designer's assumptions, it is likely that at least some agents will fail to conform to a mechanism designer's expectations in networks as large, complex and diverse as the Internet.

The second class of assumptions we examine deals with whether the game induced by a mechanism is part of a larger repeated game. In mechanism design the induced game is typically analyzed as a single-shot game. However, in networking, agents will interact repeatedly with mechanisms experiencing, over time, multiple mechanism outcomes. We therefore consider mechanisms which induce an outcome in a stage-game of a larger repeated game. It is well known that the equilibria of repeated games can be different than the equilibria of single-shot games [11]. Indeed, previous research in the networking community has noted the effect of repeated play on various routing mechanisms [2]. This paper considers more broadly the implications of the folk theorem [11] for mechanism design in any repeated context – a context that is very common in the real-world networking environments.

The rest of the paper is organized as follows. In §2 we discuss mechanism design for the Internet. In §3 we examine the assumptions about agent's preferences and provide an example of a mechanism, Re-Feedback [7], in which assumptions about the agents play a critical role in understanding the incentive compatibility claims. In §4 we examine assumptions about the number of times an agent plays the game induced by a mechanism and examine the impli-

cations for an ad-hoc routing and forwarding protocol [18] that is designed to be incentive compatible. In §5 we discuss the impact of our arguments on mechanism design for the Internet. In §6 we conclude with a summary.

2. MECHANISM DESIGN

Since this paper is targeted primarily at the networking community we begin with a brief review of mechanism design. As described in Fudenberg [11], mechanism design can be viewed as a multi-step game of incomplete information where agent "types" are private information. In the first step of the game the mechanisms designer, or principal, designs a "mechanism", "contract" or "incentive scheme." The objective of this mechanism is to illicit "messages" or "behaviors" from agents such that the mechanism's designer, or principal's expected utility is maximized. In the case of a benevolent principal, the expected utility that is maximized is some notion of social welfare. As network architects we often optimistically view ourselves as such benevolent principals, designing mechanisms to improve some notion of overall social welfare for the network.

In next step of the game, each agent either accepts or rejects the mechanism designed by the principal. Agents that accept enter the third step and play the game induced by the mechanism. Playing the game entails sending messages that are selected based upon an agent's private "type." In a networking context, one can interpret sending a message to a mechanism as engaging in a behavior that is observable to the network providers or other network participants.

The outcome of the game induced by the mechanism is called an "allocation" or "decision" k which is computed by the mechanism from the agent messages. The allocation consists of an assignment of goods and transfers of numeraire [13]. The allocation, for instance, in a VCG-based lowest-cost routing mechanism [8] consists of a selection of routing path and numeraire transfers of monetary payments to each of the nodes on the lowest-cost path. More generally, numeraire can be in terms of anything the agent values; often these are monetary transfers, but they can also be tokens that are valuable within the context of the mechanism. For instance, in mechanisms designed for the Internet these tokens might represent the right to transmit in a wireless network or they might be tokens employed in a traffic-shaping token-bucket.

The problem facing the mechanism designer is how to construct the message space and allocation rules such that it is in the interest of agents to truthfully reveal the private information that the principal conditions it's allocation decisions upon. Said another way, the mechanism must be designed to align the interests, behaviors, and actions of the agents with the interests of the mechanism designer.

In designing a mechanism, the principal is assumed to have some leverage over the agents that influences the agents' choice of messages or behaviors. This leverage is rooted in the principal's control over how goods and transfers are allocated to the agents. When designing a mechanism for the Internet then, an important question is what can a principal assume with confidence about agent preferences? The answer to this question is critically important to both the design of the mechanism as well as the equilibria that will result in the induced game. These are the topics that we consider in the following section.

3. ASSUMPTIONS ABOUT PREFERENCES

In the game of mechanism design, each agent has a private type θ_i that determines the agent's preferences over different allocations.[1] Agent types are assumed to be drawn from a known set of types $\theta_i \in \Theta_i$. The vector of all agents types is denoted as $\theta = (\theta_1, ...\theta_I)$ drawn from a vector of possible types $\theta \in \Theta_1 \times \cdots \times \Theta_I$ with a probability density function $\phi(\cdot)$. Each agent is also assumed to be an expected utility maximizer where the agent's utility function for an allocation k from the mechanism is denoted $u_i(k, \theta_i)$.

Many mechanisms designed by the networking community have further assumed that the structure of agents' utility functions have a quasilinear form:

$$u_i(k, \theta_i) = v(k, \theta_i) + (m_i + t_i) \quad (1)$$

where m_i is the agents i's initial endowment of the numeraire, k is the allocation of the good, $v()$ is the agent's valuation function for the allocation, and t_i is transfer of numeraire to/from agents. Quasilinear utility functions are popularly adopted because utility can be transfered across agents through transfers of the numeraire.

A key underlying assumption is that the probability density function $\phi(\cdot)$, the complete sets of possible types for each agent $\Theta_1, ..., \Theta_I$, and the structure of the utility functions $u_i(\cdot, \theta_i)$ are all common knowledge. In other words, all of this information is known by all agents and the principal. The only private information is the actual type of each agent θ_i.

By assuming that the complete set of possible types for each agent and structure of all utility functions are known, the principal can theoretically anticipate the effect of incentive mechanisms upon agent behaviors. If all these assumptions are sound, i.e. the assumptions accurately represent agents' utilities and types in the real world, then the mechanism designer can with confidence predict and describe the behaviors and equilibria in the game induced by a mechanism.

3.1 Assessing utility assumptions

The question, from a network architect's perspective, is what should be assumed about agents' utilities in network environments? In this section we consider mechanisms that assume agent utility functions are composed of terms representing the value of monetary and/or network goods to an agent.

3.1.1 Monetary utility terms

A monetary term that captures an agent's increase in utility with increased monetary assets seems fairly safe to assume in a utility function. Most selfish agents would seemingly prefer a larger amount of monetary goods to a smaller amount.

However, even this relatively safe assumption may not hold when considering the time value of money. An agent may be willing to incur a smaller short-term loss for a larger long-term gain. If agents are willing to incur loses within the induced game for longer-term gains realized in a different larger or repeated game that includes the induced game, a mechanism designer has more limited leverage to shape the agents behaviors through monetary incentives. In effect, the agents will not be playing the game the mechanism designer intended.

This is interesting because it suggests that even monetary rewards or penalties may not create the incentives expected by a principal. While perhaps most likely to occur over monetary goods, a tolerance for short-term loses for longer-term benefits also potentially effects how agents value network characteristics such as the ones discussed in the next section.

3.1.2 Network-based utility terms

The next class of terms in an agent's utility function we consider are ones that represent the value of network goods. These are

[1] See (23.B) "The Mechanism Design Problem" [16], for a more complete introduction to the assumptions summarized in this section.

terms that represent the value of network performance characteristics such as throughput, latency, and loss as well as more general goods such as transmission or access privileges on a network. It is often assumed that agents' utilities are an increasing function of improvements in the network good. Assuming that agents have traffic they want to send or receive on the network, such assumptions seem at first plausible.

However, a lesson from years of quality of service research in the networking community is that simplistic models of agent utilities are inadequate in the real-world [5]. Assuming that agents always value improvements in any one metric of network performance, such as throughput, latency, or network access fails to describe any one individual agent let alone being a good model for all agents on the network [5].

Moreover, in most networks today, agents are actively seeking to send or receive traffic only a small fraction of the time. During these periods, incentives leveraging the fact that agents value improvements in network performance will be effective. But, this raises the question of what governs and motivates an agent's behavior during other times? One is tempted to answer that agents will just cooperate during periods when they themselves are not selfishly invested in how the network is performing. But such a response weakens the strength of claims that can be made regarding a mechanism, particularly in networked environments, such as we have today, where actively malicious behaviors are common.

Finally, we note that a mechanism deployed in the real world cannot selectively admit only those agents that are well modeled by the utility functions assumed by the principal. Agent utilities and types are inherently private information. Agents that do not conform to a principal's assumptions may participate in the induced game. Therefore, the claims that can be made for a practical mechanism must be seen as limited to the subset of selfish agents that match a principal's underlying assumptions.

3.2 Case study: Re-Feedback

In this section, we provide a concrete example of a mechanism, Re-Feedback [7], in which assumptions about the agents play a critical role in understanding the incentive compatibility claims. We selected Re-Feedback because it was presented at one of the premier networking conferences and leverages mechanism design in support of a proposed real-world network architecture. We first provide a brief overview of the mechanism.

3.2.1 Objectives and incentives

The Re-Feedback framework attempts to add accountability for causing congestion to transport protocols. It is a technical network architecture that enables a "receiver-aligned" view of the downstream congestion along a path. How this congestion information is employed is left to the discretion of the network operators. They could directly charge or shape traffic based upon the congestion information.

We present a slightly abstracted description of the Re-Feedback mechanism; for details readers should consult the literature [7]. The framework leverages the Explicit Congestion Notification (ECN) bits of the IP header and the congestion feedback in the transport header in a novel manner to expose to network providers along a path the sender's "true" expectation of downstream congestion.

In Re-Feedback, the sender of a flow of traffic declares the expected congestion along the path (i.e. the rate at which the congestion-experienced (CE) bit will be set by ECN-capable routers) by setting selected bits (the ECT(0) code point) in the IP header. From replies sent back from the receiver in the transport protocol, which indicate the actual congestion experienced rate, the sender is capable

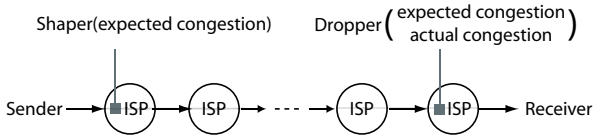

Figure 1: The incentives of Re-Feedback are designed to induce senders to truthfully report the expected congestion along a path. A shaper creates an incentive for senders to not *overstate* their view of expected congestion. A dropper creates an incentive for senders to not *understate* their view of downstream congestion.

of accurately adjusting it's expectation of downstream congestion. Routers in the framework are unmodified; they are simply assumed to set the CE bit in packets during periods of congestion.

Of course, without additional components, senders would have no incentive to accurately state their expectation of the downstream congestion. Re-Feedback tries to create this incentive through the addition of shapers and droppers in the network (see Figure 1). A shaper creates an incentive for senders to not *overstate* their expectation of downstream congestion by shaping the traffic flow to conform to a TCP-friendly rate given the stated expected congestion rate. A dropper creates an incentive for senders to not *understate* their expectation of downstream congestion by dropping enough traffic to make the rate of congestion marked packets (CE) equal to the rate of packets marked with expected congestion (ECT(0)) in a flow of traffic.

Any deviation from truthfully stating the expected downstream congestion leads to shaping or dropping the sender's traffic so that the overall throughput to the receiver is reduced. These incentives are designed to lead to a maximization of social welfare where the social welfare function is the global aggregate throughput in the network.

3.2.2 Claims and analysis

A series of claims are made regarding the behaviors that the Re-Feedback framework will induce [7].

1. "We have introduced an incentive framework which ensures that the dominant strategy of selfish parties around the feedback loop will be to declare Re-Feedback honestly." (p.11)

2. "The ingress edge network can rely on downstream congestion declared in the packet headers presented by the sender." (p.5)

3. "Inter-domain congestion charging ensures that any network that harbors compromised "zombie" hosts will have to pay for the congestion that their attacks cause in downstream networks." (p.9)

We now examine the implicit agent assumptions underlying these claims. The first assumption is that all agents' utilities are an increasing functions of throughput (throughput is the good being allocated by the mechanism). If the ingress network charges for a sender's declared expected congestion, the agent utility function $u()$ has a classic quasilinear form with a valuation function $v()$ increasing with increased throughput and the transfer charges t_i increasing as well with increases in the expected congestion rate. The question, then, is are these sound assumptions for agents on the Internet?

Perhaps, in some scenarios these assumptions would be sound. But if Re-Feedback is positioned as a *general* purpose incentive architecture for the Internet this may not always be the case. Consider, for instance, a prevalent problem on today's Internet of denial of service attacks. Agents that participate in such attacks not only do not value their own throughput, they also want to diminish others' throughput as well.

As indicated by the zombie claims above, the Re-Feedback framework seeks to address such malicious behavior. But consider the following behavior of an agent that wanted to launch a denial of service attack on the network infrastructure. All the agent would have to do would be send network traffic at any rate (up to full access line rate) declaring no expected downstream congestion (e.g the ECT(0) code point is not set on any packets).

But note that this strategy could reasonably be employed by non-malicious agents as well. An application might be designed that employed a loss resistance encoding (e.g. erasure coding) and streamed a large data set across the network. Declaring no expected congestion would result in the Re-Feedback dropper discarding many packets if the network were congested, but the application would be able to make use of any available non-congested network periods.

Note the result of this strategy in the Re-Feedback framework. Even if the ingress network is charging for declaring downstream congestion, the strategy is cost free as no congestion is ever declared. As no packets are marked with an ECT(0) code-point, the shapers also have no effect. The flow of traffic will be discarded by a dropper in the Re-Feedback framework, but, since droppers necessarily maintain flow state (and thus will always likely be closer to one of the network edges), this may not be until the egress edge network. (Egress droppers are depicted in the Re-Feedback paper [7].) This means that no network element before the first dropper can ever rely on the expected congestion declared for a flow if such agents are indeed present on the network.

Now consider that to combat this strategy, a dropper is added to the ingress edge of the network. An agent only has to declare an expected level of congestion equal to the congestion on the path from the sender to the ingress dropper. But still no element on the path from the ingress dropper to the egress dropper can rely on the congestion information declared by the sender. Adding additional droppers along the path raises the costs to the sender, but no guarantee can ever be made that all elements along a path can rely upon the declared expected congestion rate being representative of the actual downstream congestion.

The fundamental issue is that only the agents that seek to maximize throughput to a receiver will exhibit the social welfare maximizing behaviors. Given that many selfish agents on the Internet may not conform to these assumptions, the mechanism claims for Re-Feedback should perhaps be interpreted more narrowly.

Note, though, that we do not consider it a failure that Re-Feedback does not accommodate all the types of selfish agents on the Internet. Creating an incentive mechanism that aligns the interests of a subset of agents may be a worthwhile improvement over the current Internet that largely assumes full cooperation from all agents. It is, however, important to understand the limitations.

4. MECHANISM DESIGN FOR REPEATED GAMES

Classically, mechanism design is viewed as inducing a single-shot game. However, when mechanism design is applied to the construction of protocols and architectures for networking problems, it is actually more likely the same agents will be repeatedly playing the game induced by the mechanism.

From this perspective, the outcome of a game induced by a mechanism must be seen as the outcome of a stage-game i.e. one iteration of a single-shot game, in a larger repeated game. However, the effect of incentives in a single-shot game can be different then in a repeated game. The classic example of this is the prisoners' dilemma game. The only equilibrium in the single-shot game is for each prisoner to defect and take a plea bargain. However, in the repeated game, staying silent can also be an equilibrium strategy [11].

In general, any mechanisms designed by the networking community that are repeatedly played must be analyzed as repeated games. This entails that important theoretical results in repeated games must be considered – namely the "folk theorems" for repeated games (see Fudenberg [11] for formal statements of the folk theorems in repeated games).

The folk theorems assert that, if players are sufficiently patient, any individually rational, feasible outcome can be enforced by an equilibrium. To be individually rational, players select actions in each stage game that minimizes the maximum possible loss that they will face in the overall repeated game. A feasible outcome is one in which the rationality condition is satisfied for all agents. Thus, in a repeated game, almost any outcome can be an equilibrium outcome [11].

But since any feasible outcome can be supported for the repeated game, this raises the question of how much influence the incentives of the mechanism designer inducing each stage game has over the overall equilibrium of the repeated game. Consider again the classic prisoners' dilemma cast as a mechanism design problem. The principal representing the justice system wants to allocate prison sentences in such a way that induces guilty suspects to defect from their partners and tell the truth about their crimes i.e. the principal wants to design an incentive compatible mechanism.

However, in the context of a repeated game, prisoners will always maximize their utility by continuously remaining silent in each stage-game. Such an equilibrium can be enforced, for instance, if agents adopt tit-for-tat or grim trigger strategies that punish any agent that ever defects. The principal representing the justice system in this repeated game cannot design an incentive compatible mechanism for a single stage game if the penalty allocated to two prisoners that both remain silent must always be lower than the penalties if they both defect.

In different contexts, though, a principal can, to a degree, influence the equilibrium of the repeated game through the design of the messages that each agent can send to the mechanism. The work of Afergan [2], for instance, considers the effect of protocol periods and field granularity on the equilibrium price computed by a routing protocol. We are unaware of general results in this area; it appears that each mechanism must be analyzed individually in the context of a repeated game to understand what effect the control of incentives in each stage game will have over the equilibria in the repeated game.

4.1 Case study: ad-hoc networking

To illustrate the importance of considering repeated games in the engineering of network protocols we consider the incentives created in ad-hoc wireless networks by the protocols described by a 2005 paper in one of the premier conferences on wireless and mobile networking [18]. We focus in particular on the protocols for the routing stage. Space limitations preclude us from presenting a longer overview of the protocol. For specific details, consult the paper [18].

4.1.1 Ad-hoc routing and forwarding protocols

The goal of the routing stage is to compute the true costs, i.e. the power levels required for transmission, for each link along a path in the ad-hoc network. Based upon these costs, the price paid to each node on the lowest cost path is computed similar to the VCG-like mechanism presented in [4].

The challenge in wireless ad-hoc networks, the authors note, is that the cost of a link cannot be determined by the sender alone. Receivers are an integral part in reporting what power levels the sender must employ. In certain circumstances, which the authors describe, receivers have an incentive to misreport the power levels that a sender requires for transmission. To address this challenge the protocol designers employ a cryptographic solution that involves sending multiple messages, encrypted under a key shared with a third party, at increasing power levels. The receiver transmits all received messages to the third party that decrypts the messages and computes the true cost of each transmitting node.

4.1.2 Claims and analysis

A series of claims are made regarding the behaviors that these routing and forwarding protocols will induce [18].

- "We ... design the first incentive-compatible, integrated routing and forwarding protocol in wireless ad-hoc networks." (p.13)
- "We show that following the protocols is a dominant action for [the routing stage.]" (p.13)

These claims, however, are based upon an analysis of a single-shot game induced by the protocols. However, routing and forwarding in an ad-hoc network will unquestionably be a repeated game. As Afergan [2] notes, agents can advantageously deviate from the behaviors they would exhibit in a single-shot version of a VCG-based routing game. Namely, agents that collude (either implicitly or explicitly) have an incentive to reveal costs that are higher than their true costs so that they can enjoy larger payments from the mechanism. Thus, the principal's goal of truthful revelation of costs is potentially thwarted in a repeated game. Indeed, collusion between agents can only be supported in the context of a repeated game.

This is interesting because the cryptographic approach taken in this purposed protocol represents considerable engineering effort to align the incentives of a single-shot game. If, in fact, the game is repeated, admitting other agent behaviors, the considerable effort at aligning the incentives in the single-shot game appears potentially less worthwhile in the context of an engineering cost analysis.

5. DISCUSSION

This paper can be seen as an examination of applying theory to practice. While simplifying assumptions are crucial to employing theory and models, this necessarily entails that any model will not capture all details of the real-world. What is crucial is to understand when theory and models provide support and understanding of a system design versus when they are no longer applicable.

The theory of mechanism design can "raise the bar" of networking and protocol designs even if it does not accommodate all the types of selfish agents on the Internet or perform exactly as expected in a repeated game. We do not consider a mechanism designed for the Internet to be a failure simply because one can construct agents that do not meet the principal's assumptions. Creating an incentive mechanism that aligns the interests of a subset of agents is an improvement over a design that assumes full cooperation.

But understanding the real-world limitations of theoretical claims is important from an architectural and system engineering perspective. It is this understanding of the real-world limitations that enables the theoretical claims to be integrated into a larger architectural discussion and engineering cost analysis. While there is not a rigorous framework in which to conduct this discussion and analysis, we offer the following criteria for designing and evaluating incentive mechanisms for complex real-world networks like the Internet.

1. *Explicitly state assumptions:* Understanding the implications and applicability of mechanism design requires the underlying assumptions to be explicitly stated so that they can be analyzed and tested for soundness.
2. *Design defensively:* Network architectures and protocols should not rely upon incentives derived from mechanism design alone to ensure that desirable system dynamics are achieved. At least some agents in any network environment will not conform to a mechanism designer's assumptions.
3. *Understand the limitations of simple models of utility:* Assuming that agents always value improvements in any one metric of network performance, such as throughput, latency, or network access is not a realistic model of real-world agents.
4. *Analyze the repeated game:* Many mechanisms designed for networks will, in fact, be repeatedly played. The incentives created by this repeated play must be analyzed as a repeated game.

6. RELATED WORK

Our work was motivated by considering the implications of deploying, in practical networks, some of many mechanisms that have been proposed for network environments in recent years. This includes the work of [4, 7, 8, 10, 18].

While our work focuses on the underlying assumptions about agents, the work on Distributed Algorithmic Mechanism Design (DAMD) [9, 10] emphasizes the importance of the algorithmic properties of mechanisms designed for the Internet. They introduce the notion of protocol-compatibility which focuses on two aspects of the practical feasibility of a mechanism: the computational tractability and deployability of a mechanism.

The work of Afergan et al. [1, 2, 3] emphasizes the importance analyzing networking problems as repeated games. One of the focuses of this work is that the mechanism designer can influence the equilibria that occur in an incentive-based routing mechanism by controlling some of the protocol parameters such as the period lengths and granularity of protocol fields [2]. These results are dependent on other agent assumptions such as the adoption of "trigger price strategies."

Practical experiences applying mechanism design and game theory to networking problems are reported in Mahajan et al. and Huang et al. [12, 15]. Both note that theory does not necessarily apply as completely or easily as one might initially have hoped.

Potentially more realistic models of agents utilities are considered in [6]. The work of [17] considers how to prove, under certain assumptions, that an implementation of a mechanism in real-world system will match a designer's specification.

Finally, if a mechanism designer can re-implement a mechanism repeatedly then any outcome the designer cares about can be implemented in dominant-strategies [14]. This becomes possible because the principal can learn agents' preferences by observing their past behaviors.

7. CONCLUSION

This work focuses attention on the underlying assumptions about agents and how they will interact with mechanisms in complex networks like the Internet. We have emphasized that strong claims are contingent upon assumptions about the selfish agents and how they will interact with a mechanism. We have suggested that claims should perhaps be interpreted more narrowly if mechanisms only aligns the incentives of a smaller subset of selfish agents that match a principal's underlying assumptions. But we emphasize that creating an incentive mechanism that aligns the interests of a subset of agents can still be seen as an improvement over a design that assumes full cooperation from all agents. Finally, we emphasize that mechanism design cannot be a substitute for a systems engineering perspective.

In summary, the contributions of this paper are the following:

- A study of two classes of assumptions (agent preferences and repeated vs. single-shot induced games) and their impact on mechanisms designed for complex, real-world networks
- A consideration of what the folk theorem entails for any mechanisms that induce an outcome in a stage-game of larger repeated game
- Example case studies that illustrate understanding and evaluating theoretical claims based upon mechanism design in the context of larger architectural and engineering discussions
- A list of architectural and design criteria to consider when evaluating or applying mechanism design for networking problems

8. REFERENCES

[1] M. Afergan. *Applying the Repeated Game Framework to Multiparty Networked Applications*. PhD thesis, Massachusetts Institute of Technology, August 2005.

[2] M. Afergan. Using repeated games to design incentive-based routing systems. In *Proceedings of the 2006 IEEE Infocomm Conference*. IEEE, 2006.

[3] M. Afergan and R. Sami. Repeated-game modeling of multicast overlays. In *Proceedings of the 2006 IEEE Infocomm Conference*. IEEE, 2006.

[4] L. Anderegg and S. Eidenbenz. Ad hoc-VCG: a truthful and cost-efficient routing protocol for mobile ad hoc networks with selfish agents. In *MobiCom '03: Proceedings of the 9th annual international conference on Mobile computing and networking*, pages 245–259, New York, NY, USA, 2003. ACM Press.

[5] G. J. Armitage. Revisiting IP QoS: why do we care, what have we learned? ACM SIGCOMM 2003 RIPQOS workshop report. *SIGCOMM Comput. Commun. Rev.*, 33(5):81–88, 2003.

[6] F. Brandt, T. Sandholm, and Y. Shoham. Spiteful bidding in sealed-bid auctions. In D. Lehmann, R. M"uller, and T. Sandholm, editors, *Computing and Markets*, number 05011 in Dagstuhl Seminar Proceedings. Internationales Begegnungs- und Forschungszentrum (IBFI), Schloss Dagstuhl, Germany, 2005.

[7] B. Briscoe, A. Jacquet, C. D. Cairano-Gilfedder, A. Salvatori, A. Soppera, and M. Koyabe. Policing congestion response in an Internetwork using Re-Feedback. In *SIGCOMM '05: Proceedings of the 2005 conference on Applications, technologies, architectures, and protocols for computer communications*, pages 277–288, New York, NY, USA, 2005. ACM Press.

[8] J. Feigenbaum, C. Papadimitriou, R. Sami, and S. Shenker. A BGP-based mechanism for lowest-cost routing. In *Proceedings of the 2002 ACM Symposium on Principles of Distributed Computing.*, 2002.

[9] J. Feigenbaum, C. H. Papadimitriou, and S. Shenker. Sharing the cost of multicast transmissions. *Journal of Computer and System Sciences*, 63(1):21–41, 2001.

[10] J. Feigenbaum and S. Shenker. Distributed algorithmic mechanism design: Recent results and future directions. In *Proceedings of the 6th International Workshop on Discrete Algorithms and Methods for Mobile Computing and Communications*.

[11] D. Fudenberg and J. Tirole. *Game Theory*. MIT Press, Cambridge, MA.

[12] E. Huang, J. Crowcroft, and I. Wassell. Rethinking incentives for mobile ad hoc networks. In *PINS '04: Proceedings of the ACM SIGCOMM workshop on Practice and theory of incentives in networked systems*, pages 191–196, New York, NY, USA, 2004. ACM Press.

[13] M. Jackson. *Optimization an Operations Research*, chapter Mechanism Theory. EOLSS, Oxford, UK, 2003.

[14] E. Kalai and J. O. Ledyard. Repeated implementation. *Journal of Economic Theory*, 83(2):308, Apr. 1998.

[15] R. Mahajan, M. Rodrig, D. Wetherall, and J. Zahorjan. Experiences applying game theory to system design. In *PINS '04: Proceedings of the ACM SIGCOMM workshop on Practice and theory of incentives in networked systems*, pages 183–190, New York, NY, USA, 2004. ACM Press.

[16] A. Mas-Colell, M. Whinston, and J. Green. *Microeconomic Theory*. Oxford University Press, New York, NY.

[17] J. Shneidman and D. C. Parkes. Specification faithfulness in networks with rational nodes. In *PODC*, pages 88–97, 2004.

[18] S. Zhong, L. E. Li, Y. G. Liu, and Y. R. Yang. On designing incentive-compatible routing and forwarding protocols in wireless ad-hoc networks: an integrated approach using game theoretical and cryptographic techniques. In *MobiCom '05: Proceedings of the 11th annual international conference on Mobile computing and networking*, pages 117–131, New York, NY, USA, 2005. ACM Press.