To boost or not to boost: a stochastic game in wireless access networks

László Toka*, Márk Szalay*, Dávid Haja*, Géza Szabó†, Sándor Rácz†, Miklós Telek*
*HSNLab, Budapest University of Technology and Economics †TrafficLab, Ericsson Research

***Abstract*—Resource allocation in wireless access networks has been an intensively researched topic recently: many proposed solutions tackle radio channel access and dynamic spectrum allocation, but traditional issues of queuing, bandwidth sharing and packet processing at wireless access points have been targeted as well. In most of the related work the competition for high quality of service is usually solved by central coordination among users via optimizing a specific target aspect of the overall communication. In this paper we take a turn and provide users with the possibility of resource allocation suggestions. We propose a wireless access sharing framework in which users have a say in optimizing their quality of service on the long term, and we tackle its analysis with the tool set of stochastic game theory. Our findings show that greedy users become polite against their counterparts when the load is relatively low with the goal of preparing for situations with high load.**

***Index Terms*—wireless network, quality of service, access point, resource allocation, stochastic game**

I. Introduction

The current state in the evolution of Internet is the "wireless Internet" in which Internet access has become available for anybody, anywhere at any time via mobile devices. As a direct consequence, if the network operators can win the trust of costumers by running the Internet without any degradation or interruption in their services, the era of the Internet as "critical infrastructure for society" will begin. In order to reach this desired situation, the once so popular research topic of quality of service (QoS) has to be revived.

Indeed, the quality of user experience in wireless access networks has been and, with the ever-increasing competition for costumers, will be an important factor in the telecommunications industry. The proliferation of mobile device usage and the boom of the Networked Society (a.k.a the Internet of Things) forces the network operators to put all their efforts on improving the service quality continuously. With this in mind, we believe that the renaissance of QoS research has arrived.

In this paper we propose a service quality assurance framework with which the existing tools in the hand of network operators are extended with the capability of *user-driven* quality control. In our system the users (and their smart mobile devices) get an opportunity to signal their online demand for scarce resources towards the network, which in turn can improve its decisions on resource allocation with the ultimate goal of raising the satisfaction of its users. As a specific example, we make the case of user signals for urgent bandwidth demands, and of the scheduling decisions made at the access point based on those.

Our contribution in this paper is two-fold. First, we derive the model of the proposed framework with the elements of stochastic game theory. Our model describes the state of the access network via the users' bandwidth demand, their backlogged jobs, and the strategies they can choose with the aim of reaching a desirable bandwidth allocation. Second, hindered by the analytical complexity, we present a numerical evaluation of our model, and we show simulation results for different heuristic policies. Our main assumption is that urgent bandwidth demands, if not served with the required resource allocation, lose their valuation with time: jobs are depreciated in the backlog. Our goal is to show that despite this pushing time constraint even a user-driven resource allocation framework may alleviate congestion situations.

The paper is organized as follows. In Sec. II we introduce the setting in which our proposed framework can be implemented. In Sec. III we discuss both seminal and fairly recent related works. In Sec. IV we provide our stochastic game theoretical model and in Sec. V we show the related numerical analysis with some insightful results. Finally, in Sec. VI we summarize our findings.

II. Background

The average size and complexity of web pages has been growing. An average web page has over 100 objects and is 1200K in size [1]. The rendering of such page is far from trivial. It is a resource consuming task causing heavy load on the user terminal. The page load times can be even in the range of 5 secs with a 10Mbps connection on a modern PC [2].

The web page load time is a key performance metric for user Quality of Experience (QoE) thus many techniques aim to reduce this Key Performance Indicator (KPI). The two main factors that web page load time depends on are the terminal CPU and the network capacity. The potential of the enhancement of either of the two factors was analyzed in [2]. The authors estimated the following gains: a) when computational time is zeroed but the network time is unchanged, the page load time is reduced by 20% b) if the network time is reduced to one fourth, but the computational time is unchanged, 45% of the page load time is reduced. In this paper we introduce a third aspect besides the CPU of the terminal and the network capacity. We demonstrate that cooperation of the network and the terminal can also result in QoE enhancement.

Our focus of interest for QoE enhancement is the start render time KPI (T1). The start render time is the moment something first displays on the user's screen and gets interactive. Human-computer interaction (HCI) guidelines [3] recommend a 1-2 second start render time. Giving the user visual feedback that something is happening shows the user that the terminal is responsive. Also the user can already start the browsing as most of the information of the web page i.e., the textual content is already available, while large pictures are still being downloaded in the background. Start render optimization takes place before the first content appears to the user. The start render is composed of time to first byte connect time, server response time, processing objects in the head of the page, initial page parsing and rendering. Optimizing the start render time is a matter of optimizing each of these delay components. There are several approaches to address this problem: a) new protocols to improve Round Trip Time (RTT) and introduce prioritization of objects e.g., SPDY and QUIC [4], b) server side optimization of the content by modules like pagespeed mod [5], or data compression proxy like [6] and c) client side methods like caching, prefetching and preloading. We focus on a mobile broadband environment in which users share a common radio resource in a cell. In this environment the main issue is the competition between users on the radio resource that none of the above methods address.

Today some operators apply Deep Packet Inspection (DPI) middle boxes e.g., [7] to introduce traffic differentiation in the network for the internet traffic. However, end-to-end encryption e.g., HTTPS, SPDY, QUIC make it impossible to support any kind of QoE enhancement in a middle box for such encrypted traffic. In this paper we show how priority information extracted at the terminal and fed back to the network can reduce the time needed to start rendering the web page. The issue of encrypted traffic is avoided by gathering information at the client side which is fully aware of the browser status. Only traffic priority information is gathered and communicated to the network node, with the consent of the user, avoiding privacy issues.

III. Related work

Dynamic assignment of network resources has been heavily studied since the appearance of integrated communication systems [8]. Contradicting goals like service differentiation, fairness, low delay, low delay-variation, starvation avoidance have to be integrated in a properly operated network. The class of potential scheduling solutions include priority schemes with various levels of aggregations, different implementations of weighted fair queuing (WFQ) [9] like resource sharing, again with various levels of aggregations. The common root of these service class based service differentiation mechanisms is the system's (or service provider's) centric optimization of resource sharing. However, in recent communication systems more short term dynamic effects are considered. The start render time KPI, mentioned above, is a good example for the need of user oriented dynamic resource assignment.

Many works have targeted the dynamic nature of wireless access sharing. The considered optimization methods include several models where the parameters are optimized by numerical investigations and a set of problems that can be attacked with general stochastic optimization tools, e.g., Markov decision processes (MDP) [10]. Those related works that turn to distributed allocation schemes mostly apply the tool set of game theory [11], [12], but some employ other modeling techniques, e.g., portfolio theory [13]. In the current work we adopt a specific game theoretic framework for dynamic user-driven resource sharing that is able to account for the stochastic and sequential nature of demand for access: stochastic games.

The analytical tools developed for system oriented resource sharing are not applicable for the quantitative assessment and optimization of user oriented resource sharing. The analytic potential of the stochastic game theory approach for user-oriented dynamic behavior has been recently discovered by many researchers. A wide range of dynamic resource sharing mechanisms of wireless networks have been defined through stochastic games, here we mention only those few that we think are the most closely related to our work. The authors of [14] apply a linear program formulation to find the stationary policy for maximizing throughput given power and delay constraints. In [15] a multi-level game theoretic model is given which accounts for an evolutionary game within the set of secondary spectrum users, and for a potential game played among the providers competing for larger slices of spectrum. In contrast to these works we build an abstract model for wireless access, and apply an allocation mechanism of discrete resource units.

IV. Model and analysis

In this section we first present our framework, engineered to solve the issues described in Sec. II, then we build its simplified model, and give an analytical formulation and an example for solving the problem of finding the optimal policy.

A. The Boosting Framework

In Fig. 1 we depict the components of our framework, i.e., the radio access point, two terminals and a content provider server, with a toy example that introduces the notion of "boosting" in a case of web page rendering. The prioritizing (or boosting) logic in the wireless access point is stylized with different queues and a Weighted Fairness Queuing (WFQ) scheduler. The "Normal user" with Terminal A is not using the boosting service, while the "Boosted user" with Terminal B indeed does. The plots drawn at both users show time on their x axes, and the cumulated downloaded data and the received bandwidth on their y axes with dotted and dashed lines respectively. Solid sections show the web objects that are downloaded in the respective bandwidth-time products.

As the result of boosting, the toy example shows that although the total time to download all 3 web objects is the same for the 2 users, the "Boosted user" gets hold of the first 2 objects earlier than the "Normal user", which is proved to be critical in terms of QoE, hence the benefits of using our proposed Boosting Framework. While the download bandwidth for the "Normal user" remains at the same level

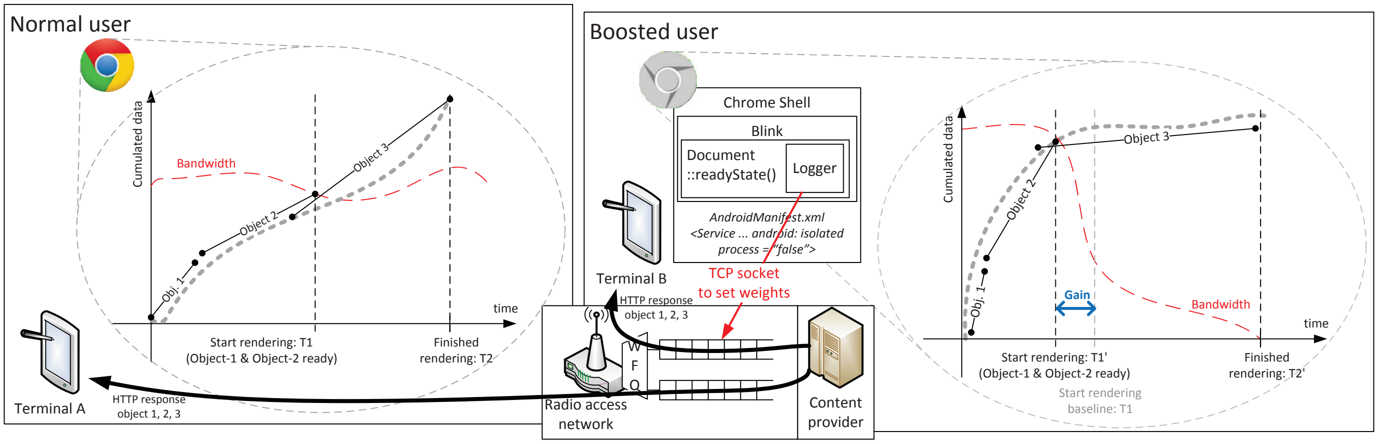


Fig. 1. The Boosting Framework in a toy example for 2 users

throughout the download session, the "Boosted user" receives higher bandwidth allocation at the beginning, and lower at the end. The boosting service in this case is implemented in the wireless access point, and is triggered by the Chrome web browser's custom plugin which sets the weights higher in the beginning and lower at the end compared to the default value for the WFQ scheduler in the wireless access point. In the following we model this as submitting bids to an auction where boosting bandwidth is allocated to users for time slots.

B. Discretized model

We split the time horizon into uniform slots and assume that in a given time slot only one user can get a unit amount of boosting bandwidth, resulting in the fact that one unit of the user's jobs gets boosted. This way we discretize our model and make sure that the auction in which the wireless access point selects the user whose job will be boosted is a single-unit auction. We apply a second-price auction: highest bid wins and pays the second highest bid for the single item. The usual notions and notations related to our auction are listed below:

- Users are denoted as $\mathcal{I} = \{1, 2, \ldots, i, \ldots, n\}$.
- The amount of boosted jobs of user i in a given time slot t is denoted by $x_i(t) \in \{0, 1\}$ $\forall i$ such that $\sum_i x_i(t) = 1$ $\forall t$. The amount of boosted jobs is defined in terms of traffic volume (given by the product of time slot length and the amount of bandwidth allocated for boosting). In each time slot only one user wins the opportunity to get its jobs boosted.
- Stochastic demand: $d_i(t)$, job arrival events are linked to time slots and job sizes are defined in the aforementioned boosted traffic volume units.
- Jobs that are not boosted cumulate in the backlog of the respective user, denoted as J_i for user i. We denote a job with j_i in the backlog $J_i = \{j_i\}$ of user i, its size as $|j_i|$ and its age, i.e., the time it spent in backlog, with $\tilde{j}_i$.
- The value of getting a boosting opportunity is $u_i(t) = u_i(x_i(t), J_i(t))$, $\forall i$, i.e., all the backlogged jobs influence the received utility because by not getting the opportunity all those jobs get delayed, hence negative effect on QoE.
- Users bid for boosting bandwidth with $b_i(t)$ in the discrete time slots. In a system implementation the bids are best produced by a browser plugin, just as it is suggested in the toy example of Sec. IV-A.
- The second highest bid, i.e., the price to be paid, is denoted by $c_i(t)$. The budget of user i in time slot t is $m_i(t)$. Accounting user budgets and subtracting costs to be paid at auctions are best handled by the access point in a possible system implementation.

C. Analysis

The system behaves as follows. In each time slot the system is in one of the states $\mathcal{S}$ that describe *traffic backlog to boost* and *remaining budget* for each user. If there are less jobs to be boosted than what the system can handle in a time slot, then naturally all of them get boosted. On the other hand, when the resource demand exceeds the offer, i.e., there are more jobs waiting for being boosted in the system than what can be served, a job ends up either being boosted or staying in backlog by the end of the time slot.

The aim of the analysis is to find the stationary policy that optimizes the service allocation according to various KPIs. The possible state transitions from one time slot to the subsequent one are due to job arrivals and serving jobs, which latter is driven by the job backlogs via boosting attempts. A Markov decision process based policy optimization approach would require the definition of selected actions $\pi_i(b_i|s) = \mathbb{P}\left[A_i^t = b_i | S_t = s\right]$ $\forall t$, where A_i^t denotes the random variable depicting user i's action in time slot t when the system is in state s. Due to the high memory dependence of the system behavior the MDP based analysis is infeasible.

Instead of state space based optimization, we treat the problem as searching for a policy in a bid-based stochastic game. In this setting the jobs are generated at users randomly, then they submit bids, finally one unit of the winner's backlogged jobs gets served. The auction winning user's job gets boosted, others' jobs to be boosted remain in their backlogs.

The payoff function users optimize is defined as follows.

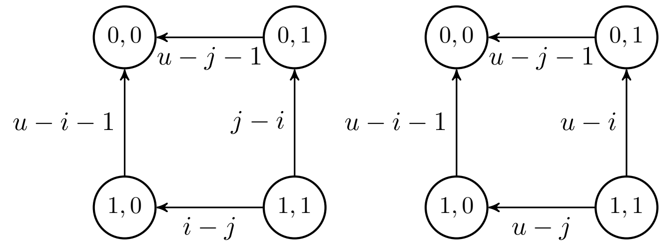


Fig. 2. Toy example for utility-based bidding policy (left-hand side) and zero bidding policy (right-hand side)

Definition 1. *The user payoff is equal to the sum of utilities of jobs served minus the cost paid for the service, i.e., for user i and time T: $p_i(T) = \sum_{t=1}^{T} u_i(t) - c_i(t)$.*

Although the value of boosted jobs is what directly affects the user QoE, we also integrate the cost that a user has to cover from its centrally allocated budget into the payoff. Doing so we intend to make the payoff resonate with the repeated and stochastic character of the game: this creates an incentive not to use up all the allocated funds at one, but keep savings for periods when multiple jobs arrive.

We assume that the rational users strive to maximize their payoffs, and therefore seek the optimal strategy: $b_i^* = \mathrm{argmax}_{b_i} \lim_{T\to\infty} \frac{p_i(T)}{T}$ with budget constraint $\sum_{t=1}^{T} m_i(t) - c_i(t) \geq 0,\ \forall T$. The strategy (bidding policy) of the user allows for describing a wide range of dynamic user and system behavior. Therefore optimization of various KPIs can be implemented through these user strategies.

D. Toy example

In order to demonstrate the complexity and the flexibility of the model, here we show a specific example. We assume a game of 2 players, unit-size jobs, service capability of 1 job per time slot in total and for both players a utility decreasing with the age of the head job of the player's backlog: $u_i(t) = u - a_i(t)$. For the sake of simplicity we assume that a player can choose between 2 actions in each time slot: either bids with the actual utility of its head job, or bids with zero.

In Fig. 2 we demonstrate two cases in which both players have one job, the first player's head job is i time slots old, the other player's head job is j time slots old. The left-hand side shows the case in which the players adopt the policy in which they bid with the head job utility, the right-hand side graph shows the case for zero value bids. The circles represent states and only those are shown that will be reached from the one-one job state until ending up in an empty system (no new job arrivals are supposed). In the circles we depict the number of jobs for the two players. On the arrows, we show the payoff of the player that wins the auction between the two states.

In the first case, whoever has the more recent job (e.g., $i < j << u$) will win the auction, but has to pay the other player's utility as cost (e.g., $p_i = u - i - (u - j) = j - i$). Then the other player can get its job boosted without any cost (e.g., $p_j = u - j - 1$), but with less utility, as the head job got older. In the second case the first time slot gets auctioned to a randomly selected player with no cost, hence e.g., $p_i = u - i$, and the other player gets the second time slot for no cost, as in the first case. Based on these payoffs, as long as $j - i > 0.5(u - i) + 0.5(u - i - 1)$ stands, i.e., $u > j > u - 0.5$, the first player is better off with the utility-based bidding policy. Otherwise, and this is the more probable case of the variables, zero value bidding results in higher payoffs. Furthermore, if $j - i < u - i - 1$, i.e., $j < u - 1$, it is actually more profitable for the first player to let the second player, with the older head job, win. The lack of costs compensates the player for the loss of utility in the latter time slot.

Note that this toy example with artificial payoff assumes that no new jobs arrive before the second allocation, and we assume that players either bid with their actual head job utility or with zero. These are restrictive assumptions, but keep the example tractable and show that even without the stochastic element of job arrivals how the sequential (or repeated) nature of the game rules out the utility-based strategy from the set of dominant strategies in these second-price auctions.

V. Numerical analysis

Based on the model presented in Sec. IV, we wrote and ran a discrete-time simulation. In this section we present its parameter settings, the analysis we made the cases for and finally the results we obtained.

A. Bidding policies and simulation parameters

We are interested in the interplay of different bidding policies. In our simulations we assume that users switch among our heuristic bidding strategies following an evolutionary process, i.e., moving towards policies that provide higher payoff. First we refine the valuation of boosting (of Sec. IV-B) for which the users bid in each round.

Definition 2. *The value of boosting a job of user i is $\max(u - \epsilon t, 0)$, i.e., the initial utility is linearly diminishing with the rounds spent in backlog.*

Before introducing the policies we investigated, let us define the term *opportunity cost*.

Definition 3. *The opportunity cost is the future loss of valuation of backlogged jobs: if a user does not get the boosting bandwidth in a given time slot, the valuation of its backlogged jobs decreases by the next time slot. Therefore the opportunity cost for user i with jobs $J_i = \{j_i\}$ is $\sum_{j_i \in J_i} |j_i|\epsilon$ where $|j_i|$ is the size of the job j_i still in backlog and ϵ is the depreciation of backlogged jobs from Def. 2.*

Now, given the time-sensitive job utility and the opportunity cost, we define three heuristic bidding policies.

Definition 4. *With greedy policy the player bids its whole budget; with rational policy one bids the actual utility of the jobs to be boosted; and with generous policy one bids the opportunity cost (Def. 3). In all policies the player's budget is the upper limit of the bid.*

We run simulations with the following parameters in order to demonstrate the pros and cons of the proposed heuristic policies, and to compare the distributed auction-based allocation with traditional bandwidth sharing.

- Number of users: $|\mathcal{I}| = 30$.
- Jobs are generated at users with independent and identically distributed exponential random inter-arrival times having mean $\beta = \frac{1}{\lambda} = |\mathcal{I}|$ (Poisson process), and all the jobs are unit-sized.
- The age of a job j_i for user i is given by the number of rounds the job has spent in the user's backlog, i.e., a job's age is 0 in the round it arrived, 1 in the following round, and so on.
- Jobs in the backlog are continuously served with the bandwidth not allocated for boosting, therefore in each time slot their size decreases by $\delta = 0.2$. This however does not induce any valuation for the user.
- Users bid for a bandwidth-timeslot unit in each round. For the auction winner i $x_i = 1$ in the given round and $x_j = 0$ $\forall j \in \mathcal{I} \setminus i$. This traffic opportunity is used to boost jobs in the winner's backlog under FIFO policy.
- Each user gets the same budget increment in each round that they use for bidding. Users strive to increase their payoffs which is the value of boosted jobs minus the cost of winning the auction.
- We assume the same initial utility and depreciation for all jobs and all players: $u = 1$ and $\epsilon = 0.1u$.

B. Fitness of policies with various job arrival rates

It is well-known that the *rational* policy, i.e., truthful bidding, would be the optimal strategy in case the game was a one-shot game because of the desirable characteristics of second-price auctions. We make the case, however, for a stochastic game, a repeated game with many different states represented by the job backlogs.

In Fig. 3 we depict the number of players implementing each policy in systems with increasing load levels. We let users change their applied bidding strategy mimicking the dynamics of evolutionary game theoretical models: we assume that when a given user wants to bid for boosting jobs it randomly selects one of the policies with probabilities proportional to the average cumulated payoffs of users grouped by their actual strategies. In the beginning users are evenly split among the policies to start with. The job arrival rate is 0.003 in Fig. 3(a), 0.03 in Fig. 3(b) and 0.3 in Fig. 3(c) with a number of players of 30 in all cases. The solid, dashed and dotted lines show the evolution of player counts with the 3 presented policies on the y-axis in the function of simulation rounds on the x-axis. When the load is low (Fig. 3(a)) the *generous* policy prevails, when the system is saturated (Fig. 3(b)) the *rational* policy seems to provide the highest payoff, while in an overloaded system (Fig. 3(c)) no policy is better than the others.

The *generous* bidding strategy pays off on the long run when the average system load is low because in most cases the relatively low bid is enough to get jobs boosted, and the budget is saved for bursty times. Intuitively, when the system is close to its saturation it is worth bidding with the actual value of the jobs to-be-boosted (*rational policy*) in order to beat users applying other bidding strategies and to get jobs boosted as soon as possible. In any case it is not beneficial to burn up the whole budget in single bids applying the *greedy* policy, unless the system is overloaded for a long time (Fig. 3(c)) but then no policy beats the others at providing the user better chance to get boosting opportunity.

C. The raise of social welfare compared to central allocation

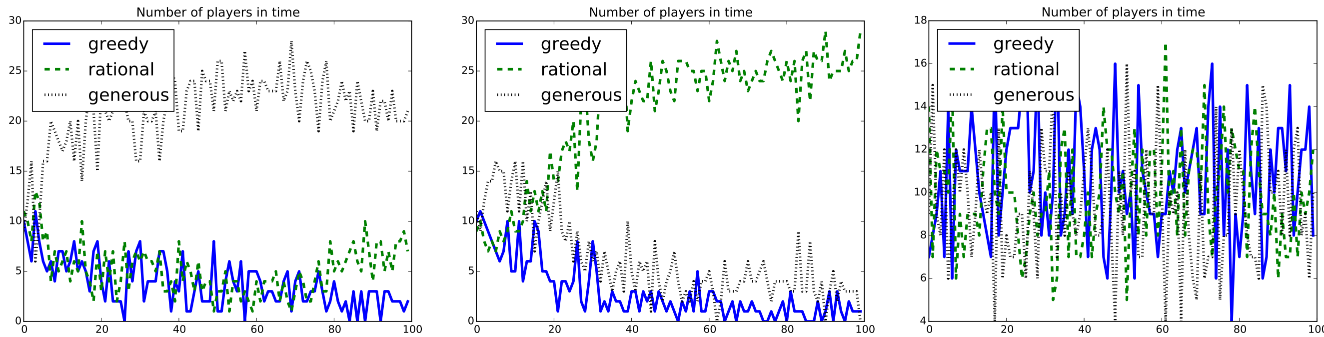
In the second set of simulations we compare the utilities of boosted jobs in a system where users all bid following the *rational* policy with a system in which users get to boost their jobs in a traditional round robin fashion. The job arrival rates are the same as in the previous batch of simulations. Both system simulations are launched with the same job arrival patterns, randomly generated beforehand.

The results are presented in Fig. 4: the subplots represent the three cases of job arrival rates, i.e., in Fig. 4(a) $\lambda = 0.003$, in Fig. 4(b) $\lambda = 0.03$ and in Fig. 4(c) $\lambda = 0.3$ with 30 users and fixed unit-sized jobs. On the y-axis the accumulated utility, on the x-axis the number of rounds are depicted. Note that as in the central allocation scheme no budget is distributed to users and they do not pay for boosting jobs, the accumulated utilities are compared, not the payoffs. When the load is low (Fig. 4(a)) the two schemes perform the same way since competitive situation is rare: at most a few users want to boost their jobs throughout the rounds. However in a saturated system (Fig. 4(b)) the *rational* policy outperforms the central allocation because round robin processing does not distinguish among jobs based on their values as bidding users do. The central allocation allows users to boost their jobs if its their round even is other users have more recent, thus more valuable, jobs to be boosted. In an overloaded system (Fig. 4(c)) this phenomenon is even more conspicuous.

VI. Conclusion

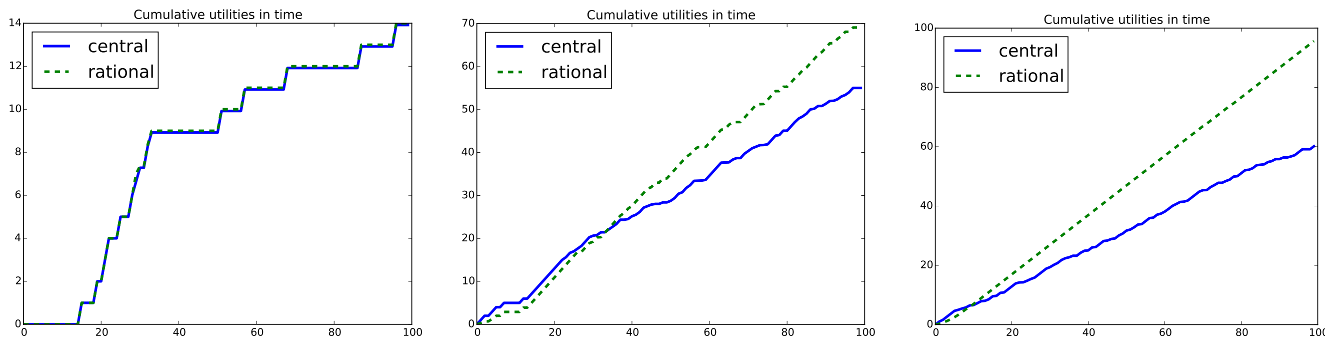
Users' quality of experience in wireless access networks has been and, with the proliferation of mobile device usage and the boom of the Networked Society (a.k.a the Internet of Things), will be an important factor in telecommunications. With our framework the existing tools in the hand of network operators are extended with the capability of user-driven quality assurance: users (and their smart mobile devices) get an opportunity to signal their online demand for scarce resources towards the network, which in turn can improve its decisions on resource allocation with the ultimate goal of raising the satisfaction of its users. We have shown an example for dynamically weighted queuing in which the actual bandwidth demands are pondered in the access point with user-provided weights. We call the "fast lane" service as boosting: boosted jobs receive a predefined fraction of the total bandwidth, but in turn the number of simultaneously boosted jobs is limited.

First, we derived the model of the proposed framework with the elements of stochastic game theory, accounting for users' bandwidth demand and their backlogged jobs, and the



(a) $\frac{0.1}{30}$ job arrival rate (b) $\frac{1}{30}$ job arrival rate (c) $\frac{10}{30}$ job arrival rate

Fig. 3. Evolution of number of players of various strategies



(a) $\frac{0.1}{30}$ job arrival rate (b) $\frac{1}{30}$ job arrival rate (c) $\frac{10}{30}$ job arrival rate

Fig. 4. Evolution of cumulative utilities with central round-robin allocation and distributed rational policy

users' strategies they can choose in order to reach the desirable allocation. Second, hindered by the analytical complexity, we chose the numerical evaluation of our model, and we have shown simulation results for different heuristic policies. Our main assumption is that urgent bandwidth demands, if not served, lose their valuation with time. Based on the results we draw the following conclusions: although users can always be viewed as greedy players when they compete for boosting opportunity, as a surprising result we have shown situations when users start saving on their budget and allow their counterparts (*generous* policy) to access the boosting opportunity when the system load are moderate. The seemingly polite users save their budgets for forthcoming load peaks. When the system is saturated a more aggressive (*rational* policy), but not over-greedy (*greedy* policy) behavior proves to be the best on the long run. More importantly, in these latter cases the distributed scheme beats the traditional central allocation.

Finally, we argue that our model is flexible enough to cover various optimization targets a network operator can be interested in. With the handouts of budgets, the operator has the power of implementing its preferences in the service policy, should it be minimizing the average delay, maximizing the fairness, or avoiding starvation. Given the constraints of budgets, users will choose the policy that best fits their payoff.

References

[1] HTTP Archive, http://www.httparchive.org.

[2] X. S. Wang et al. Demystifying Page Load Performance with WProf. In *USENIX NSDI*, 2013.

[3] N. Bevan. Human-computer interaction standards. In *International Conference on Human Computer Interaction*, 1995.

[4] QUIC. src.chromium.org /chrome/trunk/src/net/quic/.

[5] PageSpeed Module. developers.google.com/speed/pagespeed/module.

[6] OperaMax. opera.com/mobile/max.

[7] Ericsson Service Aware Support Node. ericsson.com/ourportfolio/products/service-aware-support-node.

[8] J. Roberts et al. *Broadband Network Teletraffic*. Springer, 1996.

[9] A.K. Parekh et al. A generalized processor sharing approach to flow control in integrated services networks: the single-node case. *IEEE/ACM Transactions on Networking*, 1(3):344–357, 1993.

[10] S. M. Ross. *Applied Probability Models with Optimization Applications*. Holden-Day, San Francisco, 1970.

[11] D. Niyato et al. Competitive pricing for spectrum sharing in cognitive radio networks: Dynamic game, inefficiency of nash equilibrium, and collusion. *IEEE JSAC*, 26(1):192–202, 2008.

[12] K. Akkarajitsakul et al. Distributed resource allocation in wireless networks under uncertainty and application of bayesian game. *IEEE ComMag*, 49(8):120–127, 2011.

[13] T. Wysocki et al. Spectrum management in cognitive radio: Applications of portfolio theory in wireless communications. *IEEE Wireless Communications*, 18(4):52–60, 2011.

[14] E. Altman et al. Constrained stochastic games in wireless networks. In *IEEE GLOBECOM*, 2007.

[15] K. Zhu et al. Dynamic bandwidth allocation under uncertainty in cognitive radio networks. In *IEEE GLOBECOM*, 2011.