Simple Channel Predictors for Lookahead Scheduling

Thomas Werthmann*, Matthias Kaschub*, Magnus Proebster*, and Stefan Valentin†
*Institute of Communication Networks and Computer Engineering, Universität Stuttgart, Stuttgart, Germany.
†Bell Labs, Alcatel-Lucent, Stuttgart, Germany
thomas.werthmann@ikr.uni-stuttgart.de, stefan.valentin@alcatel-lucent.com

***Abstract*—To cope with highly loaded networks in diverse traffic situations, planning the scheduling for future time slots is an efficient approach [1]. However, such Lookahead scheduling relies on future Channel Quality Information (CQI), which has to be accurately predicted. In this paper, we introduce a generic model for channel prediction errors. With this model, we study how the performance of Lookahead scheduling depends on channel prediction's accuracy. Finally, we propose a simple channel predictor. This predictor provides high accuracy, is easy to implement and of low complexity. Consequently, it is a large step in making Lookahead scheduling feasible.**

I. Introduction

Today, users request high throughput and low delay even in cellular networks. This tremendous increase in traffic load results from:

- Users that generate more traffic and spend longer time with their device [2], [3].
- New applications and multitasking operation systems, leading to traffic patterns that are difficult to model and to predict [3], [4].
- Modern applications that heavily use IP, thus, forcing the base station's scheduler to treat their flows as best effort. This ignores the applications' individual delay and rate requirements which decreases Quality of Service (QoS) and wastes capacity.

To cope with these problems, we proposed Lookahead scheduling in [5]. This new type of scheduler exploits application layer knowledge to trade off delay and throughput for each application [1].

Instead of allocating the upcoming time slot to user packets, this scheduler plans the allocation of a set of packets (called transaction) to multiple time slots in advance. For these time slots, the Channel Quality Information (CQI) has to be known prior to allocation. Thus, the performance of Lookahead scheduling relies on CQI prediction. In this paper, we study how the errors of practical prediction affect the scheduler's performance.

A. Contributions

To understand how vulnerable Lookahead scheduling is to practical CQI prediction we:

1) Propose a generic predictor model: This model allows to study the estimation error of arbitrary unbiased predictors.
2) Use this model to study the performance of the Lookahead scheduling heuristic introduced in [1].
3) Employ average-based CQI prediction for Lookahead scheduling: Comparing it to ideal prediction shows a negligible throughput loss.

From these contributions, we conclude that simple average-based CQI prediction is sufficient for practical Lookahead schedulers.

B. Related work

The related work falls in two categories. First, filter algorithms for CQI prediction are extensively treated in standard literature [6] and used in commercial products. In [7], the authors studied the impact of linear predictors on the performance of a downlink OFDMA channel. High throughput gains where observed when this simple predictor was combined with an Orthogonal Frequency-Division Multiple Access (OFDMA) subcarrier allocation heuristic. Similar gains were found in [8] for optimal OFDMA subcarrier allocation. Unlike this work, we focus on Lookahead schedulers that do not perform a subcarrier allocation for the next time slot but schedule multiple slots in advance on the full frequency band.

The second category of related work is CQI predictor models for simulation. Such models are widely used to abstract the Physical layer (PHY) in higher layer simulators and are described in, e.g., [9]. By generalizing the models of [9] as in Section III-B, we can now study arbitrary unbiased CQI predictors. This increases modeling flexibility and can be useful beyond the scope of this paper.

C. Structure

In Section II we describe our scenario, traffic assumptions, channel model, the utility-based QoS model, and the Lookahead scheduler. We introduce our generic predictor model and the practical average-based predictor in Section III. Simulation results for utility and throughput are presented and discussed in Section IV. In Section V we draw our conclusions.

II. System Model

A. Scenario

We evaluate the predictors and schedulers with an event-based system level simulator for 3GPP Long Term Evolution (LTE) systems. For a summary of the parameters of our simulations see Table I.

978-1-4673-0990-5/12/$31.00 ©2012 IEEE

TABLE I
SYSTEM MODEL PARAMETERS

Property	Value
BS/UE height	32 m / 1.5 m
Frequency	10 MHz @ 2 GHz
BS TX power	46 dBm
Antenna model	Isotropic
Path loss	$128.1 + 37.6 \log_{10}(d)$, d distance in km [10]
Shadowing	8 dB log-normal, correlation distance 50 m
Multi-path prop.	Rayleigh fading with Jakes-like temporal correlation [11], frequency-selective fading with Vehicular A channel taps [12]
UE velocity	10 km/h for radio channel effects
Link adaptation	Shannon, SINR clipped at 20 dB

We model the effects of channel variations and degradations due to shadowing, multi-path fast fading, and interference from neighbor cells. To focus on scheduling, we neglect control loops such as Automatic Repeat-Request (ARQ), handover, uplink, and Transmission Control Protocol (TCP).

We evaluate 20 User Equipments (UEs) in the center cell of a hexagonal cell layout. We place the UEs randomly over this cell on the start of each simulation drop, following a uniform distribution. At an inter-cell distance of 1 km, a ring of 6 neighboring cells cause interference by transmitting with constant power. The interference received by a UE depends on its individual position, which does not change during one simulation drop. However, we account for the effects of the UE's velocity in the channel model. Shadowing causes the channel to vary at a timescale of seconds and the multi-path fast fading causes variations with a coherence time of $\approx 12\,\text{ms}$.

The scheduler operates per Transmission Time Interval (TTI) of 1 ms and allocates all subcarriers to a single UE. Rate adaptation follows Shannon's equation but is clipped to account for the highest possible modulation order.

B. *Traffic Model*

We model Hyper-Text Transfer Protocol (HTTP) and File Transfer Protocol (FTP) traffic. This accounts for web surfing and file downloads, which contribute to the majority of best-effort traffic on mobile devices [2].

We use the traffic models from [13]. One transaction is either the download of a single object using FTP or the download of a web page including its embedded objects using HTTP. These models use truncated log-normal distributions for the size of objects, with the parameters given in Table II. The number of embedded objects belonging to a web page is drawn from a truncated Pareto distribution (mean 5.64 objects, maximum 53 objects).

While 90% of the transactions are HTTP transactions, this accounts only for 20% of the data volume. Since FTP downloads have a larger data volume, they account for the remaining 80%. We model the Inter-Arrival Time (IAT) of the transactions to be a negative exponential process with rate $\lambda = 3/4\,\text{s}^{-1}$ per user which leads to an average offered load of 30.5 MBit/s. One user can have multiple unfinished transactions at a time, which models modern Smartphones with many applications that run in parallel.

TABLE II
PARAMETERS FOR TRAFFIC MODEL DISTRIBUTIONS [13]

Object type	μ	σ	minimum	maximum
FTP	14.45	0.35	0	5 MBytes
HTTP main obj.	8.37	1.37	100 Bytes	2 MBytes
HTTP embedded obj.	6.17	2.36	50 Bytes	2 MBytes

C. *Utility Functions*

We formalize the user's QoS demand by a time-utility function [14] which assigns a utility value to a transaction finish time. Parameters for this QoS model can be obtained from studies of the user's delay acceptance such as [15], [16].

We model this quality degradation by a logistic time-utility function which is monotonically decreasing and has an inverse S-shape:

$$U(t) = \frac{1}{1 + e^{(t - t_{\text{infl}})k}} \tag{1}$$

where t is the finish time of the transaction, t_{infl} is the time of the inflection point, and k scales the steepness of the curve. We obtain these parameters in (1) from values, depending on each transaction:

$$\begin{aligned} t_{\text{infl}} &= t_{\text{start}} + x \cdot (t_{\text{exp}} - t_{\text{start}}) && (2) \\ k &= \frac{d_{\text{exp}}}{(1-x)L} \ln\left(\frac{1}{U(t_{\text{exp}})} - 1\right) && (3) \\ t_{\text{exp}} &= t_{\text{start}} + \frac{L}{d_{\text{exp}}} && (4) \end{aligned}$$

The time t_{start} is the time, when this transaction arrives at the scheduler. The users expectation, when the transaction should be finished, is expressed by t_{exp} which depends on the data size L of this transaction and the data rate d_{exp} the user expects to obtain for good experience. We model such experience by a high utility value $U(t_{\text{exp}})$. We use the parameter x to configure the steepness and solve for k accordingly.

We configure the requirements of the transactions as follows: The expected rate d_{exp} is set to 6 MBit/s for foreground (HTTP) and 3 MBit/s for background (FTP) transactions. The expected utility $U(t_{\text{exp}})$ is set to 0.95. The parameter x is set to 5.4462 for foreground and to 5.7799 for background transactions. This configuration is reasoned in detail in [1]. We observed that our algorithms yield similar results for other choices of the utility functions.

In cases where the system is overloaded some transactions never finish. To be able to include these transactions in the evaluation, transactions are dropped when their utility value has fallen below a threshold of $U_{\text{drop}} = 0.01$. Dropped transactions are accounted for with a utility of $U = 0$.

D. *Lookahead Scheduling*

The aim of our scheduling heuristic is to increase the sum of the utility over all transactions. Therefore, we schedule transactions instead of packets. Doing so reduces interleaving of transactions and thereby reduces the average finish time.

Our algorithm consists of two components. First, it selects the transactions' serving order such that the utility increases.

The second component incorporates the short term channel variations. This second step can override the decision of the first component if users with high priority (first component) have poor CQI. In the following, we describe both components and their combination.

1) Sequence selection: Assume that $\mathcal{T}$ is the set of transactions known to the basestation at time t_0, with $|\mathcal{T}| = N_T$. We define an arbitrary sequence of all transactions $S : \{1, 2, \ldots, N_T\} \rightarrow \mathcal{T}$. The sequence S can be written as $S = (n_1, n_2, \ldots, n_{N_T})$ with $n_j \in \mathcal{T}$ being the transaction at index j. The sum utility of the sequence S is defined as

$$U_{\text{total}}(S) = \sum_{j=0}^{N_T} U_{n_j}(t_{\text{fin},j}) \tag{5}$$

where $U_{n_j}(t)$ is the utility function of transaction n_j evaluated at time t and $t_{\text{fin},j}$ is the predicted finish time of the transaction n_j in sequence S. The finish times of the transactions depend on the order of the sequence and on the predicted channel capacities for the respective mobiles. They are calculated such that they fulfill

$$\sum_{\tau=t_{\text{start},j}}^{t_{\text{fin},j}-1} r_{n_j}(t_0+\tau) < s_{n_j} \leq \sum_{\tau=t_{\text{start},j}}^{t_{\text{fin},j}} r_{n_j}(t_0+\tau) \tag{6}$$

where s_{n_j} is the size of transaction n_j in bytes,

$$t_{\text{start},j} = \begin{cases} t_0 & \text{for } j = 0 \\ t_{\text{fin},j-1}+1 & \text{otherwise} \end{cases} \tag{7}$$

is the starting time of transaction n_j, and $r_{n_j}(t)$ is the rate which can be transmitted at time t to the mobile which owns the transaction n_j in bytes per TTI.

We aim to choose the serving sequence such that the sum utility increases. This task is performed by an evolutionary algorithm as follows:

1) Start with a sequence S_{old}. This is random for the first TTI. For the following TTIs, use the sequence S_{best} determined in the previous TTI and append or remove transactions to adapt to the changes in $\mathcal{T}$.
2) Create a sequence S_{new} which is derived from S_{old}, but modified randomly.
3) If $U_{\text{total}}(S_{\text{new}}) > U_{\text{total}}(S_{\text{old}})$ then replace S_{old} with S_{new}, otherwise keep S_{old}.
4) Repeat steps (2) and (3) for N_I iterations.

The resulting sequence is called S_{best}.

2) Combination of sequence with proportional fair: The Proportional Fair (PF) scheduling weight $w_n(t)$ of transaction n at time t is determined as follows

$$w_n(t) = \frac{r_n(t)}{\overline{r_n}(t)} \tag{8}$$

where $\overline{r_n}(t)$ is the moving average of the data rate, which is updated in each TTI:

$$\overline{r_n}(t) = \alpha_{\text{pf}}\overline{r_n}(t-1) + (1-\alpha_{\text{pf}})r_{\text{update},n}(t-1) \tag{9}$$

The value $r_{\text{update},n}(t-1)$ is set to $r_n(t-1)$ if the user owning n was served at time $t-1$, otherwise to 0. The parameter α_{pf} determines the time scale over which the average is calculated. Based on that, the combined weight v_{n_j} of transaction n_j at position j in the sequence S_{best} is calculated as follows

$$\forall j : \; v_{n_j} = w_{n_j}(t) - p \cdot (j-1) \tag{10}$$

where $p \in [0, \infty)$ is called penalty-factor. The transaction with the maximum v_n is served in the current TTI. The scheduling process, including sequence selection, is repeated in each TTI.

The free parameter p trades off the influence of the sequence versus the PF weight. For $p = 0$, the transaction sequence has no influence on v_n, for $p \rightarrow \infty$ the first transaction in the sequence is served independently of the proportional fair weight w_n.

III. Channel Predictors

As described above, the scheduling heuristic uses the rate $r_n(t)$ of transmissions in future TTIs to choose a beneficial sequence of transactions. We introduce three predictor models to analyze the influence of imperfect channel prediction on that heuristic.

To measure the prediction accuracy, we present a generic simulation model for estimation errors. This model allows us to evaluate the performance of Lookahead scheduling for various CQI prediction errors. Then, we propose a simple prediction algorithm that is based on the latest CQI value, e.g., observed from CQI feedback. We use the following notation:

- $t_0 \in \mathbb{N}$: The time (in number of TTIs since start of the simulation) of performing the estimation.
- $\tau \in \mathbb{N}$: The time offset (in number of TTIs) for which the channel is predicted, $\tau >= 0$.
- $\gamma_{\text{real}}(t)$: The real Signal to Interference-plus-Noise Ratio (SINR) at time t in logarithmic scale.

A. Ideal predictor model

The baseline is given by full channel knowledge at the base station scheduler. We denote this case as

$$\gamma_{\text{ideal}}(\tau, t_0) = \gamma_{\text{real}}(t_0+\tau) \tag{11}$$

and call it *ideal* CQI.

B. Generic predictor model

We design the generic model for the channel predictor γ_{gen} to approximate the behavior of feasible prediction mechanisms. This allows us to study how the performance of a Lookahead scheduling algorithm depends on the prediction accuracy, without restricting to a special realization of a prediction mechanism. To draw valid conclusions, we have to avoid that the scheduling algorithm under investigation implicitly exploits properties of the prediction error model which would not occur with real predictors.

We model γ_{gen} as a weighted combination of the future channel quality $\gamma_{\text{real}}(t_0+\tau)$ and a random error $\gamma_{\text{err}}(t_0, \tau)$, where $\gamma_{\text{real}}(t_1)$ and $\gamma_{\text{err}}(t_2, \tau)$ are uncorrelated for all values of t_1, t_2, and τ. By setting the weight, we define the accuracy of the hypothetical prediction mechanism.

The error γ_{err} has to be random, because otherwise (e.g. if using the long-term average $\gamma_{\text{err}} = \overline{\gamma_{\text{real}}}$) the scheduler could improve the estimation by amplifying the difference of the estimation to the non-random value. It is required that $\gamma_{\text{err}}(t_0, \tau) = \gamma_{\text{err}}(t_0 + \Delta t, \tau - \Delta t) = \gamma_{\text{err}}(t_0 + \tau, 0)$, so that the scheduler cannot improve the estimation by averaging over multiple predictions for the same TTI $t_0 + \tau$. Thus, the value of γ_{err} for an absolute time instant remains the same for all observation times t_0. As a consequence, we omit the second parameter and write $\gamma_{\text{err}}(t_0 + \tau)$. In addition, the error γ_{err} has to have the same auto-correlation function as γ_{real} to assure that the scheduler does not improve the estimation by averaging over τ.

We assume that the prediction error $\gamma_{\text{gen}}(t) - \gamma_{\text{real}}(t)$ follows a normal distribution. As $\gamma_{\text{real}}(t)$ is normally distributed, $\gamma_{\text{gen}}(t)$ also has to follow a normal distribution. The predictor needs to be unbiased and needs to have the same variance as the predicted value:

$$\mathrm{E}\left(\gamma_{\text{real}}(t)\right) = \mathrm{E}\left(\gamma_{\text{gen}}(t)\right) \tag{12}$$

$$\mathrm{Var}\left(\gamma_{\text{real}}(t)\right) = \mathrm{Var}\left(\gamma_{\text{gen}}(t)\right) \tag{13}$$

As mentioned above, the predictor model is formed by combining γ_{real} and γ_{err}:

$$\gamma_{\text{gen}}(\tau, t_0) = a \cdot \gamma_{\text{real}}(t_0 + \tau) + b \cdot \gamma_{\text{err}}(t_0 + \tau) + c \tag{14}$$

The random process $\gamma_{\text{err}}(t)$, which is independent and identically distributed as $\gamma_{\text{real}}(t)$, produces the estimation error. With a weight $w(\tau)$ controlling the size of the estimation error, the parameters a, b, and c are determined such that the conditions (12) and (13) are fulfilled:

$$\begin{aligned} a &= \sqrt{1 - w^2(\tau)} \\ b &= w(\tau) \\ c &= (1 - a - b) \cdot \overline{\gamma_{\text{real}}} \end{aligned} \tag{15}$$

Ideal prediction is modeled with $w(\tau) = 0$, whereas $w(\tau) = w_{\max}$ models an estimation without knowledge. We determine $w_{\max}$ such that the maximum prediction error is not higher than if the predictor is predicting the long-term average channel quality $\overline{\gamma_{\text{real}}}$:

$$\mathrm{Var}\left(\gamma_{\text{real}}(t) - \gamma_{\text{gen}}(t)\right) \leq \mathrm{Var}\left(\gamma_{\text{real}}(t) - \overline{\gamma_{\text{real}}}\right) \tag{16}$$

With (12) to (15), this results in

$$w \leq \sqrt{3/4} \tag{17}$$

Therefore:

$$w_{\max} = \sqrt{3/4} \tag{18}$$

We model a hypothetical channel prediction algorithm which shows a prediction error that increases with τ:

$$w(\tau) = \min(w_{\max}, s \cdot \tau) \tag{19}$$

The value of s controls how fast the estimation error increases. For $s = 0$, we model the ideal channel prediction. With $s = w_{\max}$ the estimation error immediately rises to its maximum value for all $\tau > 0$. We are not interested in a non-ideal knowledge of the current channel situation, therefore $w(0) = 0$, which is independent of s. By adding a constant offset to w, the model could be extended to include inaccuracies from channel measurement and signaling. Figure 1 illustrates the behavior of Mean Squared Error (MSE) of the generic predictor model. The parameter s defines the gradient of the MSE curve.

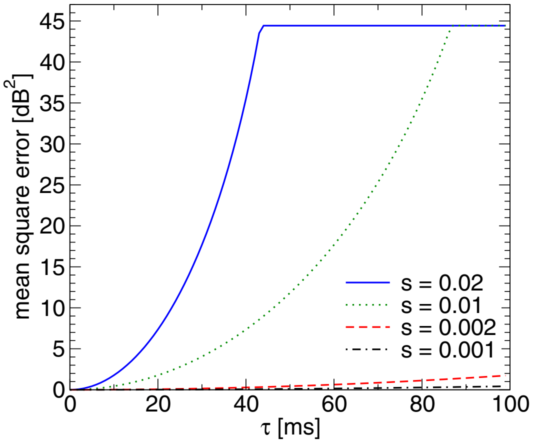

Fig. 1. MSE of the generic predictor model, for illustration. The model parameter s configures how the prediction performance degrades with increasing time offset τ.

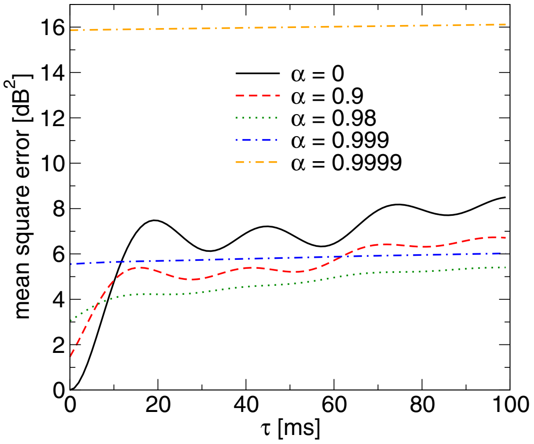

Fig. 2. MSE of the average predictor. The filter coefficient α configures how quickly the average value adapts to the current channel situation. For $\alpha = 0$, only the current CQI is used, for $\alpha \rightarrow 1$ the long term average is used.

C. Average predictor

The average predictor uses an auto-regressive filter on the current channel quality. Filtering the previous channel measurements averages-out short term variations (small scale fading). The coefficient α controls the filter's attenuation:

$$\begin{aligned} \gamma_{\text{avg}}(\tau, t_0) &= \gamma_{\text{ar}}(t_0) \qquad \text{with} \\ \gamma_{\text{ar}}(t_0) &= \alpha \gamma_{\text{ar}}(t_0 - 1) + (1 - \alpha) \gamma_{\text{real}}(t_0) \end{aligned} \tag{20}$$

The prediction is independent of τ, i.e. the filter predicts a constant value for all future TTIs.

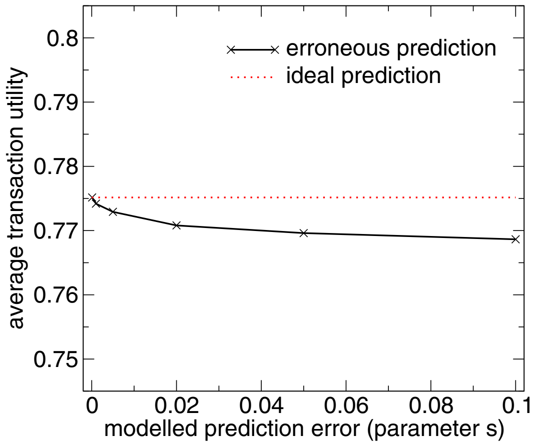

Fig. 3. Utility with generic predictor model, utility with ideal prediction shown as reference. With increasing the prediction error s, utility degrades only slightly. PF is not aware of utility functions and achieves an average utility of 0.35 (not shown).

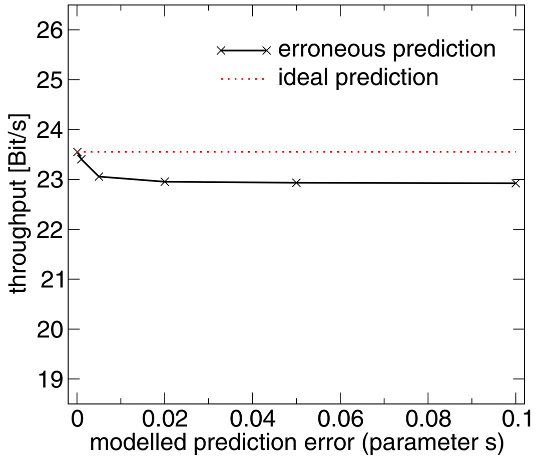

Fig. 4. Throughput with generic predictor model, throughput with ideal prediction shown as reference. When increasing the prediction error s, throughput reduces by 2.7 %.

Figure 2 shows the mean square error of the average predictor for different filter coefficients. For $\alpha = 0$, the current channel value is predicted, and no filtering is performed. Therefore, the predictor performs well for $\tau < 10\,\mathrm{ms}$ as the channel's auto-correlation is high within the coherence time of $\approx 12\,\mathrm{ms}$. Beyond the correlation run-length of fast fading, i.e., $\tau > 10\,\mathrm{ms}$, the prediction is not accurate.

With $\alpha \approx 0.98$, the inverse cut-off frequency of the filter lies between the correlation run-lengths of fast fading and shadowing. Here, the filter removes the fast-fading fluctuations but follows the shadowing. This improves prediction accuracy for $\tau > 10\,\mathrm{ms}$ but is less accurate for $\tau < 10\,\mathrm{ms}$. With larger values for α, the filter also attenuates shadowing fluctuations. This degrades the prediction performance for relevant ranges of τ.

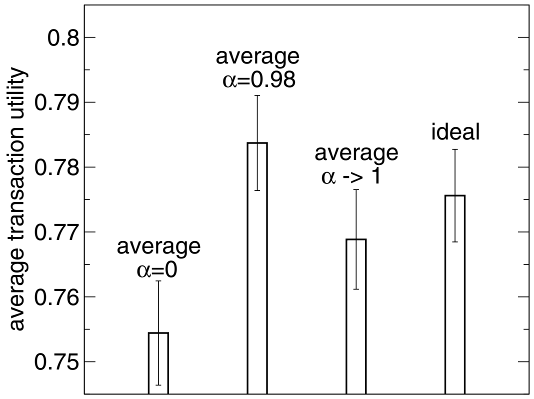

Fig. 5. Utility with average predictor, utility with ideal prediction shown as reference. Best utility is achieved when α is set such that the filter removes the fluctuations by fast-fading but follows the shadowing. PF is not aware of utility functions and achieves an average utility of 0.35 (not shown).

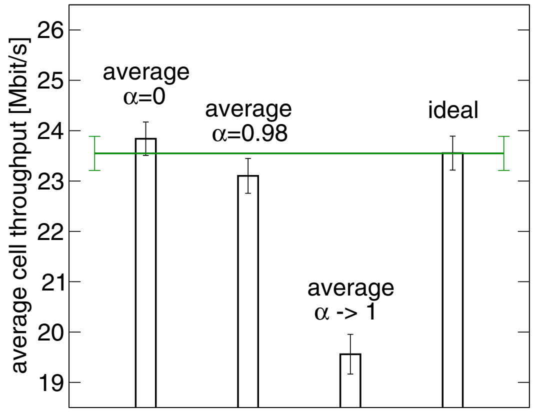

Fig. 6. Throughput with average predictor, throughput with ideal prediction (right bar) and with prop. fair (green line) shown as reference. With increasing α, the predictor disregards knowledge of the current channel quality, which hinders channel adaptation.

IV. Performance Evaluation

We first evaluate the influence of channel prediction on the performance of the Lookahead scheduler by simulation with the generic model. Then, we evaluate the performance of the same scheduler with the realistic channel predictors using the model from Section II. Based on the results in [1], the scheduler is configured as follows: N_I is set to 400, as with this value the performance has converged to the maximum for the simulated traffic load. The penalty-factor p is set to 1, which offers a beneficial trade-off between utility and cell throughput. The PF coefficient α_{pf} is set to 0.999.

We evaluate two metrics. Cell throughput, averaged over all TTIs and independent replications, and the transaction utility. The latter is defined as the arithmetic mean over the utility achieved by all transactions in the cell, which are either completely served or dropped within the simulation time.

Figure 3 shows the performance of the scheduling heuristic in average utility. We see that, when the prediction accuracy decreases (i.e., increasing s), the average utility only slightly degrades.

As for the utility, increasing s degrades the cell throughput by 2.7 % in comparison to ideal channel prediction (cp. Figure 4). While scheduling can still adapt to short-term CQI, the serving order of the transactions is not adapted to the real channel quality. Since now only the second (short-term) component of the heuristic is aware of the channel, the overall cell throughput degrades.

After investigating the performance with the generic predictor model, we now apply the realizable predictor. We focus on three values of α. For $\alpha = 0$, only the current channel value is used and no filtering is performed. For $\alpha = 0.98$, the filter removes the fluctuations by small-scale fading but follows shadowing. The MSE performance of these parametrizations is the best in the regions of $\tau < 10\,\text{ms}$ and $\tau > 10\,\text{ms}$, respectively. For comparison, we also show results for $\alpha \rightarrow 1$, which completely ignores instantaneous CQI.

Figure 5 shows the utility performance of the Lookahead scheduler for these configurations of the average predictor. For all cases, the Lookahead scheduler outperforms simple PF scheduling which is not aware of utility functions and achieves an average utility of 0.35. Setting $\alpha = 0$ performs worst in terms of utility as this prefers UEs with a currently good channel and impairs the stability of planning ahead. With $\alpha = 0.98$, the performance can be slightly increased beyond ideal CQI. This is because the AR-filter reduces the fluctuations in channel quality and, thus, simplifies the sequence selection. Further increasing α towards 1 reduces the utility performance again, because prediction accuracy degrades and the real channel variations are strongly absorbed. Thus, time-shifting of transactions based on this knowledge is not advantageous anymore.

Besides utility, we show the cell throughput results in Figure 6. The reference line gives the throughput result of PF which equals to the result with ideal channel prediction in this simulation. Except for $\alpha \rightarrow 1$, the throughput performance is similar. The poor performance with $\alpha \rightarrow 1$ results from reducing the instantaneous CQI to average channel quality. Then, the scheduler does not consider channel fluctuations to decide when a transaction should be served.

As shown, $\alpha = 0.98$ leads to the highest utility and throughput. Fortunately, the Lookahead scheduler proves robust against impairments in channel prediction accuracy and shows a graceful degradation for different values of α. All in all, simple prediction algorithms with periodic CQI updates are sufficient for Lookahead scheduling. The large utility gains, compared to PF, can be sustained even with non-ideal CQI. Consequently, the performance of a Lookahead scheduler is mainly affected by accurate knowledge of the average CQI and of the transaction properties.

V. Conclusion

From the above results we conclude that simple CQI predictors, such as a moving average filter, are powerful companions to new schedulers. Although CQI updates are more valuable than sophisticated predictors, the average-based predictor still provides acceptable performance while adding no overhead for CQI feedback.

The performance degradation with our simple predictor is still acceptable when compared to full CQI knowledge. This is even the case when large transaction sizes require a long prediction window. Consequently, simple predictors are a feasible link layer extension to profit from the gains of Lookahead scheduling.

References

[1] M. Proebster, M. Kaschub, T. Werthmann, and S. Valentin, “Context aware resource allocation for cellular wireless networks,” *EURASIP Journal on Wireless Communications and Networking (WCN)*, Oct. 2011, submitted to. [Online]. Available: http://www.ikr.uni-stuttgart.de/en/∼proebster

[2] G. Maier, F. Schneider, and A. Feldmann, “A first look at mobile hand-held device traffic,” in *Proceedings of the 11th International Conference on Passive and Active Network Measurement (PAM)*, vol. 6032, Apr. 2010, pp. 161–170.

[3] H. Falaki, R. Mahajan, S. Kandula, D. Lymberopoulos, R. Govindan, and D. Estrin, “Diversity in smartphone usage,” in *Proc. Ann. Int. Conf. on Mobile Systems, Applications, and Services (MobiSys)*, Jun. 2010.

[4] U. Paul, A. P. Subramanian, M. M. Buddhikot, and S. R. Das, “Understanding traffic dynamics in cellular data networks,” in *Proc. Ann. Joint Conf. of the IEEE Computer Societies (INFOCOM)*, May 2011.

[5] M. Proebster, M. Kaschub, and S. Valentin, “Context-aware resource allocation to improve the quality of service of heterogeneous traffic,” in *Proceedings of the IEEE International Conference on Communications (ICC)*, Jun. 2011.

[6] A. H. Sayed, *Adaptive Filters*. John Wiley & Sons, Inc., 2008.

[7] Z. Shen, J. G. Andrews, and B. L. Evans, “Short range wireless channel prediction using local information,” in *Proc. 37th Asilomar Conference on Signals, Systems and Computers*, Pacific Grove, USA, Nov. 2003, pp. 1147 – 1151.

[8] M. Goldenbaum, R. A. Akl, S. Valentin, and S. Stańczak, “On the effect of feedback delay in the downlink of multiuser OFDM systems,” in *Proc. Conf. on Information Sciences and Systems (CISS)*, Mar. 2011.

[9] F. Frederiksen and T. Kolding, “Performance and modeling of WCDMA / HSDPA transmission / H-ARQ schemes,” *Proceedings of the 56th IEEE Vehicular Technology Conference, VTC 2002-Fall*, pp. 472–476, 2002.

[10] “Physical layer aspects for evolved universal terrestrial radio access (UTRA),” 3GPP WSG RAN, Tech. Rep. TR 25.814, 2006.

[11] P. Dent, G. Bottomley, and T. Croft, “Jakes fading model revisited,” *Electronics Letters*, vol. 29, no. 13, 1993.

[12] “Selection procedures for the choice of radio transmission technologies of the UMTS,” ETSI, Tech. Rep. TR101112/UMTS30.03 V3.2, 1998.

[13] R. Irmer (ed.), “Radio access performance evaluation methodology,” NGMN White Paper V1.3, 2008.

[14] S. Ryu, B. Ryu, H. Seo, and M. Shin, “Urgency and efficiency based packet scheduling algorithm for OFDMA wireless system,” in *Proceedings of the IEEE International Conference on Communications (ICC)*, vol. 4, May 2005, pp. 2779 – 2785.

[15] R. B. Miller, “Response time in man-computer conversational transactions,” in *Proceedings of the December 9-11, 1968, fall joint computer conference*, Dec. 1968.

[16] S. Niida, S. Uemura, and H. Nakamura, “Mobile services: User tolerance for waiting time,” *IEEE Veh. Technol. Mag.*, vol. 5, no. 3, pp. 61 –67, Sep. 2010.