Semi-Static Interference Coordination in OFDMA/LTE Networks: Evaluation of Practical Aspects

Donald Parruca
Fahad Aizaz
Chair of Communication and Distributed Systems
RWTH-Aachen University
Germany
parruca@umic.rwth-aachen.de

Soamsiri Chantaraskul
The Sirindhorn International Thai-German Graduate School of Engineering (TGGS)
King Mongkut's University of Technology North Bangkok
Thailand
soamsiri.c.ce@tggs-bangkok.org

James Gross
School of Electrical Engineering
KTH Royal Institute of Technology
Sweden
james.gross@ee.kth.se

ABSTRACT

To minimize interference in LTE networks, several inter-cell interference coordination (ICIC) techniques have been introduced. Among them, semi-static ICIC offers a balanced trade-off between applicability and system performance. The power allocation per resource block and cell is adapted in the range of seconds according to the load in the system. An open issue in the literature is the question how fast the adaptation should be performed. This leads basically to a trade-off between system performance and feasible computation times of the associated power allocation problems. In this work, we close this open issue by studying the impact that different durations of update times of semi-static ICIC have on the system performance. We conduct our study on realistic scenarios considering also the mobility of mobile terminals. Secondly, we also consider the implementation aspects of a semi-static ICIC. We introduce a very efficient implementation on general purpose graphic processing units, harnessing the parallel computing capability of such devices. We show that the update periods have a significant impact on the performance of cell edge terminals. Additionally, we present a graphic processing unit (GPU) based implementation which speeds up existing implementations up to a factor of 92x.

Categories and Subject Descriptors

I [**Computing Methodologies**]: SIMULATION AND MODELING; I.6.4 [**Model Validation and Analysis**]: Methods—*performance measures*

General Terms

Algorithms, Measurement, Performance

Keywords

OFDMA; LTE; ICIC; Inter-Cell Interference Coordination; GPU; GA; Genetic Algorithm; 4G; Cellular Networks; Interference; Proportional Fair Scheduling

1. INTRODUCTION

Since the introduction of LTE systems there has been extensive research on mitigating inter-cell interference stemming from frequency reuse one. In inter-cell interference coordination (ICIC), power is allocated for specific parts of the frequency spectrum such that mobile stations in neighbouring cells experience low interference. A vast amount of ICIC schemes have been proposed taking quite different approaches. One way to group them is by the time dynamics. Static ICIC approaches [1] and [2], decide on a fixed parametrisation of frequency or power allocation. On the one hand the deployment efforts are low and taken during network planning phase. On the other hand, such schemes lack adaptation to the instantaneous load distribution in the cells leading to inefficient allocation of resources. In order to increase the efficiency, dynamic ICIC approaches have been introduced. Dynamic ICIC approaches adapt the power assignments to different frequency shares according to the current distribution of terminals and/or to the instantaneous channel gains.

Depending on the execution periodicity, dynamic schemes can further be distinguished into two categories. Highly dynamic schemes [3] and [4] adapt their coordination parameters in the range of a few milliseconds, e.g. 1-100 ms. In this category the negotiated parameters are combined with the local scheduler in order to adapt to the instantaneous channel gains. The nature of such a coordination approach requires exchange of overhead information between base stations, and if this is supposed to run at a high periodicity then the signalling overhead is very high. In addition to this, the short update periods leave little room for computing near-optimal allocations. As a consequence, the application of ICIC on short time scales in the range of milliseconds is unlikely to be feasible in practice.

A solution to the above problems has been originally proposed in [5] where semi-static ICIC is introduced. The power

allocations are adapted over much longer time spans like seconds or longer. This is a form of soft frequency reuse (SFR) [6], where the overall spectrum is reused by all base stations, but the transmission power in each resource block is restricted to a certain level. Although the idea of infrequent coordination intervals seems appealing, it also brings with it a wide set of challenges. Due to the long time span of the coordination period and the random evolution of fast fading and scheduling decisions, the overall system performance for a given power allocation becomes a random variable itself, turning the interference coordination problem into a stochastic optimization problem where the expected system performance is to be maximized. Optimization on expected systems performance turns out to be a highly non-linear optimization problem depending on the fading statistics of the signal-of-interest as well as all interfering signals. In addition, these statistics somehow translate into a throughput behaviour which is further influenced by the modulation and coding schemes as well as the dynamic resource scheduling at the individual base stations. So two fundamental problems are to be solved: (i) Modelling the expected system performance and (ii) Optimizing the power allocations accordingly. We have already addressed both these issues: In [7] we have addressed stochastic performance models for interference limited LTE systems. In [8] we have then fundamentally addressed the issue of solving the optimization problem. In our work we showed that - in principle - the resulting stochastic optimization problems could be solved near-optimal by the application of genetic algorithms (GA).

There still remains the question how frequent the semi-static ICIC approach should be executed. This depends mainly on two issues: How fast do the channel states change (in comparison to the underlying statistical model used for the computations). That is essentially a question of mobility. On the other hand, the question arises how fast the power allocations can be generated. Basically this is a trade-off because mobility-wise one would expect that the more frequent the power allocations are updated, the better the system should perform. However, there must be a limit to the computation times which sets a hard limit on the periodicity. The contribution of this paper is to address this trade-off. We deal with it by: (i) Firstly, evaluating how sensitive the system performance is when operating semi-static ICIC at different update periods (ii) Secondly, we introduce a highly efficient implementation of our proposed genetic algorithm in [8] for semi-static ICIC based on general purpose graphic processing units, taking advantage of their high degree of parallelism. Besides the programming aspects of our implementation we show in particular that the reuse of solutions from previous allocation phases results in a drastic speed-up of the execution run-time of the coordination scheme. This has a direct impact on the system, where with decreasing update times for the semi-static ICIC, an improvement in performance is noticed. In contrast, related works on semi-static ICIC [9] and [8] perform their evaluations on static drops of terminals and synthetic simulation models. In general, neither paper looks at the impact of the update period nor at the computation times of the stochastic optimization problem.

The work is structured as follows. In the next section we present the system model that we are working with. Afterwards, our approach on performing semi-static ICIC and problem definition are introduced in Section 3. The implementation details of our GPU-based implementation are presented in Section 4 and the corresponding evaluations in Section 5. Finally, conclusions are drawn in Section 6.

2. SYSTEM MODEL

We consider the downlink communication of an LTE cellular network. The multiple-access technique used is Orthogonal Frequency Division Multiple Access (OFDMA). The frequency spectrum with carrier frequency f_{C} and bandwidth B is equally split into N subsequent chunks called resource blocks, which are the minimal unit that can be used for data transmission over the air interface. Each resource block n is comprised of N_{C} orthogonal subcarriers which are used for the transmission of N_{S} sequential OFDM symbols. Time is organized in time slots of duration T_{TTI} where for each time slot t base stations take scheduling decisions, which we refer to in the following as fast resource assignments. In this process, packets from local queues at the base station are matched to resource blocks for transmission to mobile stations. In the following we assume proportional fair scheduling (PFS) for the fast resource assignment originally introduced in [10].

Each base station k transmits data packets to its set $J(k)$ of mobile stations through the air interface. Meanwhile, through a different interface (called X-2 in the LTE context), each base station communicates with K other neighbouring ones for inter-cell coordination purposes, forming a so called coordination cluster. Semi-static coordination is periodically performed for each base station involved in the cluster. A virtual master node, which we will refer to as *Central Entity* (CE), decides for every base station involved in the cluster on the power allocations per resource block $p_{k,n}(t)$. It is valid for the time duration T_{C}, being the total time needed to collect the pathloss values $\overline{h^2_{k,j}}$ at the CE, executing the semi-static ICIC algorithm and communicating the power masks $p_{k,n}(t)$ back to the corresponding base stations. It is much longer than the fast resource assignment interval ($T_{\mathrm{TTI}} = 1$ ms). A schematic example of a cluster is shown in Figure 1. Network wide, mutually interfering base stations are grouped together forming independent inter-cell interference coordination clusters where their transmission powers are decided by the corresponding CE (which could be for example one of the base stations in the coordination cluster).

Every TTI, each mobile terminal j feeds its instantaneous channel capacity $C(\gamma_j, n(t))$ back to its serving base station. It can be considered as a mapping of SINR to rate function where specific values can be found in [11]. The actual channel capacity directly depends on the instantaneous signal-to-interference-plus-noise ratio (SINR):

$$\gamma_{j,n}(t) = \frac{h^2_{0,j,n}(t) \cdot p_{0,n}(t)}{\sum_{k=1}^{K} h^2_{k,j,n}(t) \cdot p_{k,n}(t) + N_0}, \qquad (1)$$

where $h^2_{k,j,n}(t)$, $k = 0, \ldots, K$ are the instantaneous channel fading gains with corresponding means $\overline{h^2_{k,j}}$ (pathloss) of the serving ($k = 0$) and interfering base stations ($k > 0$). Furthermore, N_0 is the noise power.

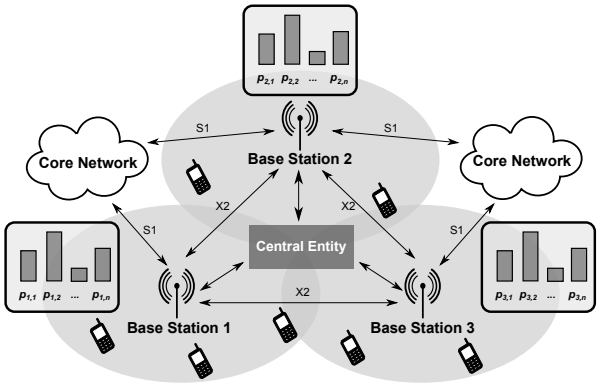


Figure 1: System model of semi-static ICIC for a cluster of three cooperating base stations.

For fast resource assignments according to PFS, the serving base station uses the instantaneous feedback to construct a priority metric $\hat{r}_j(t)$ based on the ratio of the instantaneous rate and the total scheduled data $\bar{r}_{j,n}$ over the last W time slots as the following:

$$\hat{r}_{j,n}(t) = \frac{C(\gamma_{j,n}(t))}{\bar{r}_{j,n}}. \tag{2}$$

The mobile station j_n^* having the highest priority is then scheduled for transmission:

$$\forall k, n : j_n^*(t) = \arg \max_{j \in J(k)} \hat{r}_{j,n}(t). \tag{3}$$

LTE operates with a unique modulation and coding scheme if multiple resource blocks are assigned to the same terminal. We assume in the following that this is utilized, i.e. no individual MCS selection per resource block is performed.

3. PROBLEM STATEMENT

ICIC is mainly in place to improve the cell-edge performance. Therefore, any optimization problem that improves this performance can be considered. In the following, we study a problem formulation where the CE decides on the power allocations $p_{k,n}(t)$ by solving the following max-min stochastic optimization problem:

$$\begin{aligned} \max \quad & \rho \\ \text{s.t.} \quad & \sum_{n=1}^{N} \mathcal{R}_{j,n}(p_{k,n}(t)) > \rho \quad \forall k, \quad \forall j \in J(k), \\ & \sum_{n=1}^{N} p_{k,n}(t) \leq P_{\max} \quad \forall k, \\ & p_{k,n}(t) \leq p_{\max} \quad \forall k, n. \end{aligned} \tag{4}$$

Above, $p_{k,n}(t)$ denotes the combination of transmission powers $p_{0,n}(t), \ldots, p_{K,n}(t)$ per resource block n and base station k in the cluster. Furthermore, the maximal allowed transmission power per resource block is $p_{\max}$ and the maximal one per base station $\sum_n p_{k,n}(t)$ is $P_{\max}$. The goal of the optimization problem is to maximize the expected rates of the terminals at the worst locations in the cluster area by an appropriate choice of the power allocations.

For the computation of the expected rates $\mathcal{R}_{j,n}(p_{k,n}(t))$ a stochastic model is needed taking into account the impact of the proportional fair scheduler on fading and interference limited wireless links. Considering the instantaneous SINR $\gamma_{j,n}(t)$ as a random variable $Z_{j,n}$, then from [7] the rate expectation $\mathcal{R}_{j,n}(p_{k,n}(t))$ for PFS based systems can be computed as:

$$\begin{aligned} \mathcal{R}_{j,n}(p_{k,n}(t)) &= \frac{N_{\mathrm{S}} \cdot N_{\mathrm{C}}}{T_{\mathrm{TTI}}} \\ &\cdot \int_0^{\infty} C(z) \prod_{\forall m \in J(k)/j} F_{Z_{m,n}} \left(\frac{\mathrm{E}[Z_{m,n}]}{\mathrm{E}[Z_{j,n}]} \cdot z \right) \cdot f_{Z_{j,n}}(z)\, dz, \end{aligned} \tag{5}$$

where $F_{Z_{j,n}}(z)$, $f_{Z_{j,n}}(z)$ and $\mathrm{E}[Z_{j,n}]$ are respectively the SINR cumulative function (CDF), the SINR probability density function (PDF) and the SINR expectation. Assuming the instantaneous fading gains $h^2_{k,j,n}$, as exponentially distributed, from [12] and [13] we have the PDF and CDF of SINR for multiple interfering sources as:

$$F_{Z_{j,n}}(z) = 1 - \prod_{k=1}^{K} \frac{1}{1 + z \cdot \frac{\overline{h^2_{k,j}} p_{k,n}}{\overline{h^2_{0,j}} p_{0,n}}} \exp\left(-\frac{z N_0}{\overline{h^2_{0,j}} p_{0,n}} \right). \tag{6}$$

Then, taking the derivative of function $F_{Z_{j,n}}(z)$ with respect to variable z we obtain:

$$\begin{aligned} f_{Z_{j,n}}(z) = & \left(\sum_{k=1}^{K} \frac{\overline{h^2_{k,j}} p_{k,n}}{\overline{h^2_{0,j}} p_{0,n} \left(1 + z \cdot \frac{\overline{h^2_{k,j}} p_{k,n}}{\overline{h^2_{0,j}} p_{0,n}} \right)^2} \right. \\ & \cdot \prod_{q \neq k} \frac{1}{1 + z \cdot \frac{\overline{h^2_{q,j}} p_{q,n}}{\overline{h^2_{0,j}} p_{0,n}}} + \left. \prod_{k=1}^{K} \frac{N_0}{\overline{h^2_{0,j}} p_{0,n} + z \cdot \overline{h^2_{k,j}} p_{k,n}} \right) \\ & \cdot \exp\left(-\frac{z N_0}{\overline{h^2_{0,j}} p_{0,n}} \right). \end{aligned} \tag{7}$$

The rate model in Formula (5) is solved through numerical methods. Its accuracy and validity has been already evaluated with system level simulations in [12].

As it can be noticed optimization variable $p_{k,n}(t)$ has a nonlinear relation to the rate expectation $\mathcal{R}_{j,n}(p_{k,n}(t))$. This makes it difficult to apply analytical methods in obtaining optimal power allocations. In our previous work [8], we showed, without considering algorithm computational runtimes and the mobility of terminals in the propagation environment, that such a problem can be solved near-optimally through genetic algorithms.

In contrast, in this work we are interested in characterizing the dependency of semi-static ICIC on the periodicity of executing the algorithm. On the one hand, we are interested in the overall system performance for different periodicities. On the other hand we are interested in characterizing the algorithm's run time for different system parametrizations. In the following we study how the system performance in realistic scenarios behaves for different values of the update period T_{C}. Secondly, we present an efficient semi-static ICIC

implementation that can be executed in parallel on a GPU with low execution times. We aim to reduce the computational times of the ICIC algorithm presented here, such that they are significantly smaller than T_{C}. This approach still has to collect the pathloss from the base stations and then also signal the new power assignments back to the base stations requiring extra overhead time.

4. IMPLEMENTATION OF GA IN GPU

Genetic algorithms are stochastic search methods used to solve optimization problems based on natural selection principles. A candidate solution, which in our case are the quantized power allocations $p_{k,n}(t) \in \{0, p^l, 2 \cdot p^l, \ldots, L \cdot p^l\}$, where $p^l = p_{\max}/L$ are encoded as a two dimensional matrix $((K+1) \times N)$, which in GA lingo we refer to as chromosome and individual parts of it as genes. An initial population of chromosomes is repetitively altered by operations of evaluation, selection, crossover, mutation and validation. One iteration of the above operations is called a generation.

In each generation an intermediate population is generated when selection is applied. It initiates a tournament among randomly selected chromosomes where the fitness, decided in the evaluation process, is compared. The fittest chromosome is selected for crossover. Genes of randomly picked parent chromosomes from the intermediate population are swapped to form the next offspring. Afterwards, with a low probability, mutation is applied to the genes of the offspring. It alters the transmission power $p_{k,n}$ of a gene by adding or subtracting the value p^l. In this stage, it can happen that the chromosomes do not fulfill the power constraint of the given optimization problem. Therefore, during the validation process, a check is performed for the newly generated population. In case the allocated transmission power per cell is above the limit $P_{\max}$, random genomes of the cell violating the constraint are picked up and their power is reduced. The process repeats itself until the constraint is met. Now, it is possible to evaluate the fitness of the chromosomes in the new population. It is the minimal expected rate among all terminals in the cluster for the given chromosome $p_{k,n}(t)$ which, is also the metric that we are optimizing: $\min_{\forall j} \left(\sum_n \mathcal{R}_{j,n}(p_{k,n}) \right)$. As it can be noticed, after each generation fitter candidate solutions are generated, bringing the evolution closer to the optimal solution (power combination).

We exploit the parallelism of GPUs for general purpose computation by using Compute Unified Device Architecture (CUDA) framework. CUDA framework provides a mapping between CUDA programming constructs and GPGPU hardware. CUDA uses a notion of *block* which maps to hardware streaming multi-processor. A number of blocks can be assigned to a multi-processor where they are time-shared internally by the CUDA programming environment. A defined number of *threads* can be configured for a block to run on streaming processors. Each thread executes a single instruction set called the *kernel*. The kernel code executes in batches of *warp* size of the device in a time-shared fashion, simultaneously over streaming processors. Each thread uses a number of private registers for its computation, whereas threads within a block share common limited memory called "shared memory". Threads and blocks are given a unique ID that can be accessed within the thread during its execution. These can be used by a thread to perform the kernel task on its part of the data resulting in a Single Instruction Multiple Data (SIMD) execution.

Our implementation of genetic algorithm on GPGPU exploits parallelism over several layers. We use multiple threads to evaluate a single chromosome in parallel and multiple blocks for different individuals simultaneously. This is realized by organizing the data in GPGPU memory in such a way that genes ($p_{k,n}$) of each individual can be accessed efficiently in a coherent manner by multiple threads handling it. This access pattern is known as coalesced access.

Each GA operation is implemented as a separate kernel function which executes one after another. The initialized population is retrieved from device memory. Each kernel operates on retrieved population in parallel and once all operators are executed, the new population is kept back in the global memory and thus a generation in GA life cycle is completed. We also take advantage of shared memory for keeping intermediate chromosomes during the process of selection, crossover and mutation which reduces the number of access to the global memory and enhances the overall performance of computations. This process is shown in Figure 2.

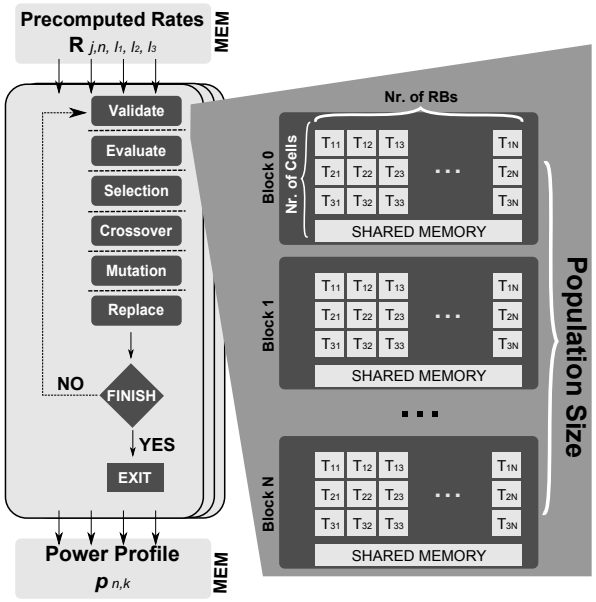


Figure 2: CUDA implementation of GA. Each thread $T_{k,n}$ manipulates the corresponding genes $p_{k,n}(t)$.

In literature related to GA implementations in GPU [14] and [15] the methodology applied by the authors for implementation of genetic algorithm on GPUs is similar. The basic idea exploited is to realize different phases of genetic algorithm as separate CUDA kernels and apply them on GPU for parallel execution. Despite all the similarities, the main difference that stands out in our approach is the evaluation of the fitness function. Based on the problem at hand, the fitness function needs to be evaluated for every generation on the fly for each of the candidate solutions. As specific power

combinations for each resource block $p_{k,n}$ might need to be evaluated multiple times during the evaluation process, recomputation of user rates would add a significant overhead. In our case, we simplified the problem by pre-computing the user rates $\mathcal{R}_{j,n}(p_{(l_1,l_2,l_3),n})$ for every permutation of the set (l_1, l_2, l_3) that are required during fitness calculations.

A further very efficient way to reduce the execution time of GA is by minimizing the number of generations needed to reach a near-optimal solution. This is done by taking advantage of the spatial correlation of pathloss. It can be expected that the topology of terminals does not drastically change between subsequent runs. As a consequence, the best individuals of the GA will also be similar. Consequently, the less generations it will take to reach the optimum. We can take advantage of this feature by seeding 10% of the new population with the fittest solution of the previous allocation phase. The number of generations needed to reach the optimum is also directly connected to the mobility of the terminals. The slower the mobility is, the more similar are the optimal solutions, resulting in less generations to be computed. Carefully initializing the population also helps in quickly reaching a near optimal solution. We initialize the population with random and commonly used approaches of power allocations in static allocations, i.e. uniform and orthogonal power allocations.

In our previous work [8], the same optimization problem was solved using a serial implementation with GALib library on standard CPUs. The runtime achieved for solving the optimization problem (4) by the genetic algorithm was in the order of minutes, being too slow for practical deployments.

5. PERFORMANCE EVALUATION

In the first part of the evaluation we consider the dependency of the system performance with respect to different update periods T_C in realistic scenarios where mobility is present. The evaluation is further extended by analysing the performance of the proposed semi-static ICIC approach to different comparison schemes widely used in literature. Afterwards, we investigate the run times of the GA implementation for the GPU and evaluate its ability to deliver near-optimal solutions.

5.1 Comparison Schemes

We compared the performance of three different inter-cell interference coordination approaches for a cluster of three cooperating base stations.

- The first one is our proposed semi-static ICIC scheme periodically generating power profiles $p_{k,n}(t)$. Due to illustration reasons we name it as SFR. Four different periods of T_C : 0.1, 0.5, 1 and 2 seconds were simulated by re-using the genomes of previous solutions. According to the results of the next section (Section 5.3), we set the number of iterations to 1000 before terminating the genetic algorithm.
- The second scheme considered is fractional frequency reuse (FFR). In FFR, the frequency spectrum is split into two parts exclusively dedicated to cell center and cell edge terminals. Terminals in either group get resources exclusively assigned from the one or the other spectrum by the proportional fair scheduler. The spectrum dedicated for cell-centred users (RB index 1-10) is fully reused in the neighboring base stations. Meanwhile, for cell-edge terminals, a bunch of 5 resource blocks is orthogonally reused in the neighbouring cells. The categorization of terminals as centrally or edge located is periodically done every 100 ms. It is locally performed for every cell according to a greedy algorithm introduced in [12]. First, all users are assumed as centrally located then, the ones with the worst predicted rate are subsequently reallocated to the cell-edge spectrum. The process is repeated as long as the estimated minimal rate of the cell is increased. For estimation, the rate prediction model from Formula (5) is used. This FFR scheme is dynamic and has the same optimization goal as the SFR scheme proposed here making it a fair candidate for comparison.
- Finally, two static ICIC schemes are also considered. In the first one, the whole spectrum is re-used in the neighbouring cells. The difference to our proposed semi-static ICIC, is that here the power allocation is not changed from RB to RB but is the same for all resource blocks (equalling $P_{\max}/N$) causing strong inter-cell interference. We refer to this scheme as frequency reuse one (FR1). In the second, orthogonal frequency bands in the neighbouring cells are used, i.e. the frequency band is split into disjoint sets of frequency bands and each base station is operating on one of those bands. In our case a frequency reuse of 3 (FR3) scheme was used, resulting in a network where no interference at all is present.

5.2 Simulation Parameters

As simulation environment we chose the urban area of the city of Munich. From the German radio-monitoring agency we obtained the geographical location and base station antenna orientation of a cellular network operator. From the vast amount of cells, we worked with a cluster of three sectors radiating inwards the simulation playground as shown in Figure 3a. The profile of 120° sectorized antennas and their corresponding orientation and position together with a 3D model of the city served as input for a ray-tracing algorithm to generate a RF propagation map of the simulated area. As a result, we could associate to every pixel the corresponding path-loss with the neighbouring base stations. The corresponding long-term SINR of the simulation environment is shown in Figure 3a.

Through the ONE mobility simulator [16], the node movement across the streets was replicated. After initially dropping terminals in the environment (30 drops in total) mobile terminals started to move with a velocity of 30 km/h according to a map based random walk model [16]. The position of terminals was continuously traced and mapped to the corresponding pathloss $\overline{h^2_{k,j}}$ with the surrounding base stations.

The pathloss values $\overline{h^2_{k,j}}$ were fed into our OMNET++ based system level simulator, mimicking the radio access of an OFDMA/LTE system. Fast fading based on Jake's model was simulated and proportional fair scheduling on fully buffered traffic queues was performed. The simulated bandwidth was 5 MHz ($N = 25$ resource blocks) with carrier fre-

quency of 1.8 GHz while the maximal transmit power $P_{\max}$ was set to 20 W. The total number of terminals considered was $J = 60$ and the noise power per resource block was assumed as $N_0 = -112$dBm. For every drop the downlink communication of 60 seconds duration was simulated.

5.3 System Performance Evaluation

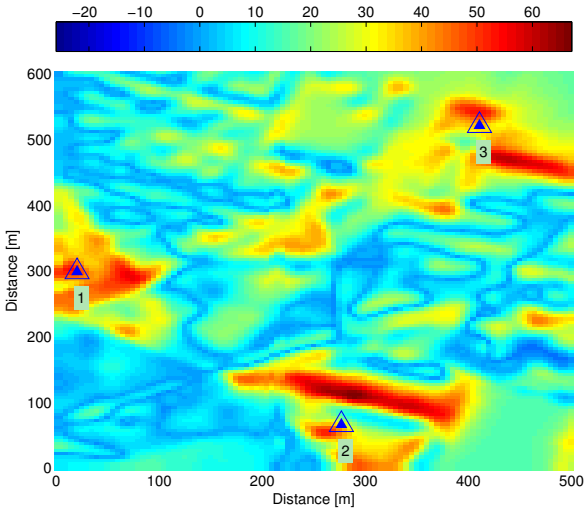
We basically considered the average rate per terminal as performance metric. To capture the rate variability of the terminals at different positions, we evaluated the average rates by building the empirical cumulative distribution functions (ECDF) over all terminals rates observed during the simulation, which serves as primary performance metric. It tells us the likelihood that the minimal observed data rate X in the cluster is smaller than or equal to a number x. The 5% percentile on the rate ECDF is a common metric used in standardization to evaluate the cell-edge performance which we will use in the following as well. The corresponding curves are given in Figure 3b. Observing the lower part of Figure 3b (the 5-th % percentile) we notice that the semi-static approach proposed here delivered the best performance for the cell-edge terminals. For different update intervals (0.1 vs 2 s) there is a performance difference, with the shorter periodicity of 0.1 s offering slightly better results.

The general rates ECDF metric gives information about the overall system performance. However, we are also interested to further highlight the performance of the optimization goal, namely the rate of cell-edge terminals for different update periods. In order to do so, we collect the minimal rates per simulation run and every second of simulation time. Based on these data the ECDF curves are computed, leading to Figure 3c. There, we can notice that the performance of cell-edge terminals is notably improved compared with the other comparison schemes and that shorter update periods result in a much better performance of cell-edge terminals. The quicker the update time is, the better the adaptation of power allocations to the topology of mobile terminals. With the time passing by, the position of mobile stations changes and so does also the pathloss with the neighboring base stations. This results in a mismatch between the predicted rates during the power mask optimization process and the actual rate obtained during network operation. The generated power allocations are not relevant to the topology of terminals with the time passing by resulting in a performance loss.

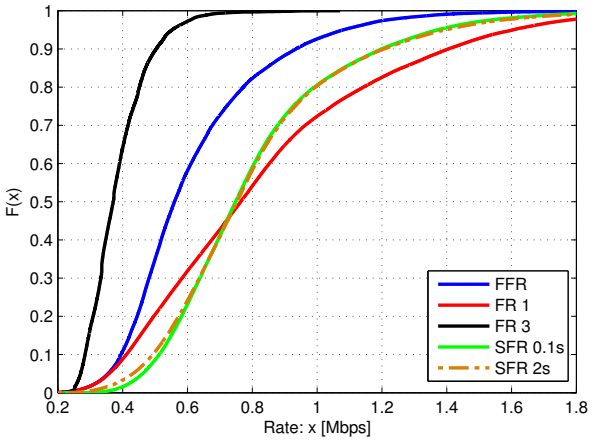
These observations bring us to the conclusion that significant gains can be harnessed in semi-static ICIC schemes for relatively low (smaller than 1 s) update periods. Therefore, efforts in minimizing the run-times of semi-static ICIC schemes are beneficial to the cell-edge terminals performance. It needs to be mentioned that the gain depends on the mobility of terminals and more investigations are required to consider also other mobilities, but were not performed in this paper due to space limitations. In the next section we evaluate the execution run-time of our proposed semi-static ICIC approach.

5.4 Evaluation of GAs Implementation

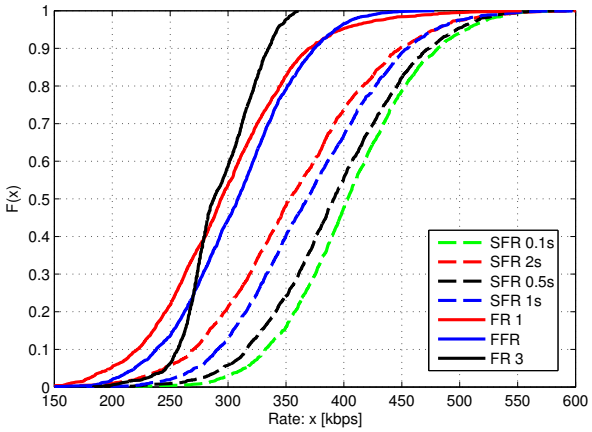
The run-time of our GA implementation for a fixed number of generations of 2000 and variable number of resource blocks and terminals is shown in Figure 4a. The measure-



(a) SINR in dB of simulation playground.



(b) General Rates ECDF Curves. The rate of all mobile terminals in the system is considered.



(c) Minimal Rates ECDF Curves. The rate of *only* cell-edge terminals in the system is considered.

Figure 3: Evaluation of different approaches for interference coordination by system level simulations.

ments were performed on a NVIDIA Quadro 6000 GPU. We notice that, the execution runtime increases linearly with the number of RBs and terminals. This is due to the fact that we exploit parallelism (in GPU) in multiple dimensions. The increase in the number of terminals impacts runtime significantly as compared to the number of RBs. Although all the chromosomes in the population are operated in parallel, the evaluation of the genome is a search operation through all the users (i.e. finding the minimal rate). In total, for 60 mobile stations and 3 cells, it took only 0.38 s for the GA to execute[1]. Compared to our previous implementation running on a four-core machine with conventional CPUs (AMD Phenom(tm) II X4 945 processor and 8 GB RAM) the pre-computation implemented also in parallel took 64.20 seconds to complete and the optimization time of GA took 21.4 seconds. In total we notice a 92x improvement on the pre-computation and a 56x improvement on the GA optimization. This is already a significant improvement due to the GPU implementation discussed previously.

In the following, we investigate the number of generations needed from the GA to find the near-optimal solutions for the given search space. This is an important factor as the GAs execution run-time is linearly connected with the number of generations. For benchmarking purposes, we use a linearised version of the optimization problem (4) in finding the optimal solution. The linearised problem is given as follows:

$$
\begin{aligned}
\max \quad & \rho \\
\text{s.t.} \quad & \sum_{n=1}^{N}\sum_{l_1=0}^{L}\sum_{l_2=0}^{L}\sum_{l_3=0}^{L} \hat{\mathcal{R}}_{j,n,l_1,l_2,l_3} \cdot x_{n,l_1,l_2,l_3} > \rho \quad \forall j, \\
& \sum_{n=1}^{N}\sum_{l_1=0}^{L}\sum_{l_2=0}^{L}\sum_{l_3=0}^{L} p(l_k) \cdot x_{n,l_1,l_2,l_3} \leq P_{\max} \quad \forall k, \\
& \sum_{l_1=0}^{L}\sum_{l_2=0}^{L}\sum_{l_3=0}^{L} x_{n,l_1,l_2,l_3} = 1 \quad \forall n, \\
& x_{n,l_1,l_2,l_3} \in \{0,1\}, \qquad (8)
\end{aligned}
$$

where x is the decision variable and rates $\hat{\mathcal{R}}_{j,n,l_1,l_2,l_3}$ are the precomputed input values to the optimization problem. Then, using an ILP solver we obtained the optimal solution to the ILP. The evaluation was performed for different mobilities of 3, 30 and 60 km/h of mobile terminals and with different execution periods of 0.2, 0.5, 1 and 2 seconds. The GA was run for each period 30 times in order to gather statistical confidence.

The average number of generations needed for the GA to reach 97% of the optimum is shown in Figure 4b. The average number of generations to reach near-optimal solutions was between 200 and 800 generations, respectively lasting 37 ms and 130 ms. For the studied scenario, the 3 km/h run times are slightly higher than the 30 and 60 km/h ones. This is a scenario specific behaviour as, the path-loss of mobile stations moving with 3 km/h changes more slowly than scenarios with higher mobility. The search space (precomputed rate expectations) for the 30 consequent runs of the GA is more similar than for higher mobilities. Hence, the similarity in the number of generations for the 3 km/h mobility.

We repeat the GA evaluations by reusing the power profiles generated in the previous solutions. The motivation is that the topology of mobile stations does not change drastically in consequent GA runs. Therefore, optimal power profiles in consequent runs differ slightly, helping the GA to evolve quicker to a near-optimal solution. The corresponding results are presented in Figure 4c. Observing it, we notice a drastic speed-up for high frequency adaptation (200 ms) and low mobility (3 km/h). Less than 4 ms (10 generations) were needed instead of 86 ms (500 generations) to reach near-optimal values for low mobility scenarios of 3 km/h, resulting in a 22x fold speed-up of the GPU implementation. Another trend to be noticed is also between the length of the period of optimization and the convergence to the optimum. The longer the coordination periods are the slower it takes for the GA to converge. The reason for that is that the topology of terminals differs more the longer the update periods are. The same can be said also for increasing mobility. The higher the mobility the more different will be the topology of terminals. Consequently, the number of generations to meet the optimum will also be larger.

For all mobility scenarios and update periods which we investigated so far, we could show that through an efficient implementation the run times can be reduced down to approximately 50 ms by using a GPU. That is a noticeable gain in computing time, taking into account that the previous CPU based implementation runs at 21.4 seconds.

6. CONCLUSIONS

Based on the evaluations performed in this work we observe that performance-wise smaller update periods of semi-static ICIC lead to better performance. However, the improvement by going from 0.5 seconds update period to 0.1 seconds is marginal. On the other hand, the paper shows by exploiting spatial correlation of path-loss and implementing the semi-static ICIC algorithm in GPU the computation time can be reduced from 20 seconds down to approximately 50 ms (and lower). Given these facts we conclude that parallel implementation in GPUs of semi-static ICIC is a practical approach in employing near-optimum semi-static ICIC for OFDMA/LTE networks.

7. ACKNOWLEDGEMENTS

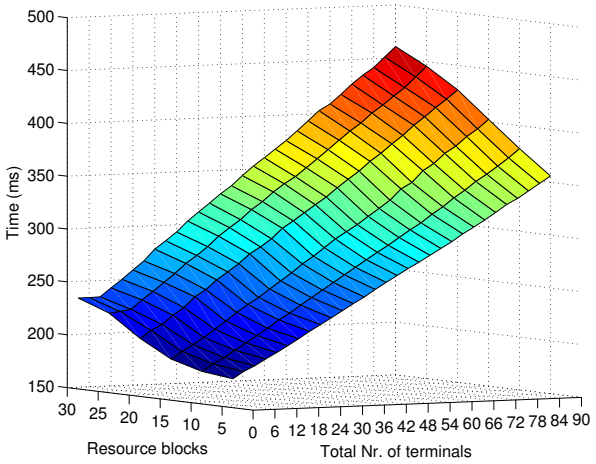
This work was partially supported by the DFG Cluster of Excellence on Ultra High-Speed Mobile Information and Communication (UMIC), German Research Foundation grant DFG EXC 89.

8. REFERENCES

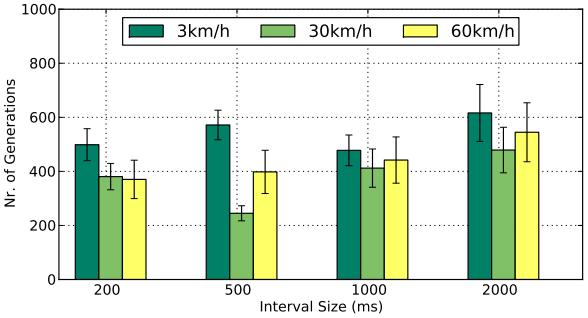
[1] M. Liang, F. Liu, Z. Chen, Y. F. Wang, and D. C. Yang, "A novel frequency reuse scheme for ofdma based relay enhanced cellular networks," in *Proc. of the IEEE Vehicular Technology Conference, 2009.*, Apr., volume = 2009.

[2] D. Gonzalez Gonzalez, M. Garcia-Lozano, S. Ruiz Boque, and J. Olmos, "An analytical view of static intercell interference coordination techniques in ofdma

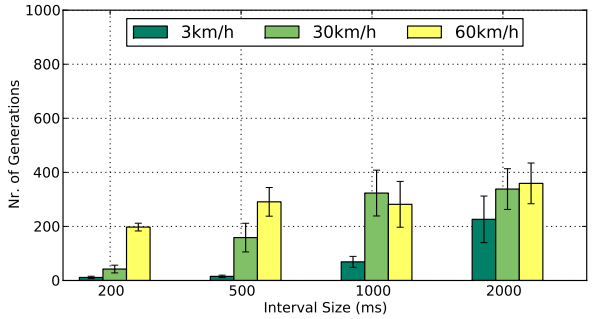
[1]The pre-computation on the worst case it took 0.7 s and is subject to further optimization.



(a) Runtime of GA optimization for 2000 generations.



(b) Mean number of generations to reach 97% of ILP optimum for a random start of GA.



(c) Mean number of generations to reach 97% of ILP optimum for reuse simulations.

Figure 4: Implementation evaluation of the genetic algorithm.

networks," in *Proc. of IEEE Wireless Communications and Networking Conference Workshops*, pp. 300–305, Apr. 2012.

[3] S. Khalifa, H. Hamza, and K. Elsayed, "Inter-cell interference coordination for highly mobile users in lte-advanced systems," in *Proc. of the IEEE Vehicular Technology Conference*, pp. 1–5, Jun. 2013.

[4] S. Wang, Y. Zhang, and G. Bi, "A decentralized downlink dynamic icic method for multi-cell ofdma system," in *Proc. of International Conference on Wireless Communications and Signal Processing*, pp. 1–5, Nov. 2011.

[5] R1-060368, "Performance of inter-cell interference mitigation with semi-static frequency planning for eutra downlink." 3GPP TSG RAN WG1#44 Denver, USA, Feb. 2006.

[6] M. Bohge, J. Gross, and A. Wolisz, "Optimal power masking in soft frequency reuse based OFDMA networks," in *Proc. of the European Wireless Conference*, (Aalborg, Denmark), pp. 162–166, May 2009.

[7] D. Parruca, M. Grysla, S. Goertzen, and J. Gross, "Analytical Model of Proportional Fair Scheduling in Interference-limited OFDMA/LTE Networks," in *Proc. of IEEE Vehicular Technology Conference*, Sept. 2013.

[8] D. Parruca, M. Grysla, H. Zhou, F. Naghibi, M. Petrova, P. Mähönen, and J. Gross, "On semi-static interference coordination under proportional fair scheduling in lte systems," in *Proc. of the European Wireless Conference*, VDE VERLAG GmbH, 2013.

[9] R. Madan, J. Borran, A. Sampath, N. Bhushan, A. Khandekar, and T. Ji, "Cell association and interference coordination in heterogeneous lte-a cellular networks," in *IEEE Trans. on Selected Areas in Communications*, vol. 28, pp. 1479–1489, Dec. 2010.

[10] A. Jalali, R. Padovani, and R. Pankaj, "Data throughput of CDMA-HDR a high efficiency-high data rate personal communication wireless system," in *Proc. of the IEEE Vehicular Technology Conference*, (Tokyo, Japan), May 2000.

[11] J. Ikuno, M. Wrulich, and M. Rupp, "System Level Simulation of LTE Networks," in *Proc. of the IEEE Vehicular Technology Conference*, (Taipei, Taiwan), May 2010.

[12] D. Parruca and J. Gross, "On the Interference As Noise Approximation in OFDMA/LTE Networks," in *Proc. of IEEE International Conference on Communicationsf*, June 2013.

[13] M. Haenggi and R. K. Ganti, *Interference in large wireless networks*. Now Publishers Inc, 2009.

[14] K. K. Rajvi Shah, P J Narayanan, "Gpu-accelerated genetic algorithms," in *Proc. of the Workshop on Parallel Architectures for Bio-inspired Algorithms*, pp. 27 – 34, 2010.

[15] J. Jaros, "Multi-gpu island-based genetic algorithm for solving the knapsack problem," in *Proc. of the IEEE Congress on Evolutionary Computation (CEC), 2012*, pp. 1–8, June 2012.

[16] A. Keränen, J. Ott, and T. Kärkkäinen, "The ONE Simulator for DTN Protocol Evaluation," in *Proc. of the International Conference on Simulation Tools and Techniques*, (New York, NY, USA), ICST, 2009.