Universität Stuttgart

INSTITUT FÜR
KOMMUNIKATIONSNETZE
UND RECHNERSYSTEME
Prof. Dr.-Ing. Dr. h. c. mult. P. J. Kühn

Institute of Communication Networks and Computer Engineering
University of Stuttgart
Pfaffenwaldring 47, D-70569 Stuttgart, Germany
Phone: ++49-711-685-68026, Fax: ++49-711-685-67983
email: mail@ikr.uni-stuttgart.de, http://www.ikr.uni-stuttgart.de

Optimized Frame Packing for OFDMA Systems

Marc C. Necker, Martin Köhn, Andreas Reifert, Joachim Scharf, Jörg Sommer
Institute of Communication Networks and Computer Engineering, University of Stuttgart
Pfaffenwaldring 47, D-70569 Stuttgart, Germany
Email: marc.necker@ikr.uni-stuttgart.de

***Abstract*— Orthogonal Frequency Division Multiple Access (OFDMA) is the basis for several emerging wireless systems, such as 802.16e (WiMAX) or 3GPP Long Term Evolution (LTE). In OFDMA, different users are multiplexed in time and frequency. In the 802.16e Adaptive Modulation and Coding (AMC) downlink, the data bursts for a particular terminal have a rectangular shape and need to be placed in the two-dimensional time/frequency plane. The position and shape of the rectangles is arbitrary, and it is the task of the frame packer to pack the frame efficiently, wasting as little space as possible. In this paper, we treat the frame packing problem as a strip-packing problem. We solve this combinatorial optimization problem by developing a suitable representation for a genetic algorithm. This algorithm can reach within 5% of the theoretical lower bound for the packing efficiency.**

I. Introduction

Orthogonal Frequency Division Multiplexing (OFDM) is a state-of-the-art spread-spectrum technique for present and future wireless broadband systems. In an OFDM-system, the available spectrum is subdivided into a large number of frequency subcarriers. Orthogonal Frequency Division Multiple Access (OFDMA) is a multiplexing technique, where different terminals are multiplexed in time and frequency based on the underlying OFDM system. This is a promising approach for future broadband wireless communication networks and has become the basis for several emerging cellular systems, such as 802.16e (WiMAX) or 3GPP Long Term Evolution (LTE).

The decision on the time and frequency ranges allocated for a particular mobile terminal is the responsibility of the MAC scheduler. The scheduler has two basic tasks. First, it needs to decide on the terminals which shall be served in a particular frame or time period, i.e., it needs to decide on the scheduling order and also the amount of scheduled data. Second, it needs to assign time and frequency resources to every transmission. This is also referred to as *frame packing*.

Scheduling in wireless networks has been studied extensively. For 802.16e systems, most work has focused on the scheduling order of terminals and the scheduled data amount, or on QoS architectures. In [1], Wongthavarawat et al. have proposed and evaluated an uplink QoS scheduler for an 802.16 system. Another uplink scheduling proposal was presented in [2] by Lee et al.

The problem of frame packing in 802.16e systems has gained far less attention. However, finding an optimal resource assignment is a non-trivial task. It is particularly difficult in the Adaptive Modulation and Coding (AMC) zone, which is well suited for advanced transmission techniques such as beam-forming. In this zone, terminals are assigned a rectangular area (referred to as *burst*) in the time/frequency OFDMA plane. Therefore, the scheduler needs to place a large number of rectangles in the time/frequency plane. These rectangles may have arbitrary measures under certain technology and traffic specific constraints. The goal of the scheduler is to efficiently pack the rectangles in the OFDMA-plane such that no free spaces are left over and the overhead is minimized.

In [3], Wan et. al. present a simple heuristic solution for the frame packing problem in combination with a scheduling algorithm for the downlink AMC zone. In this paper, we develop a near-optimal algorithm for the frame packing problem in the AMC downlink zone by treating it as a strip-packing optimization problem. We solve this combinatorial optimization problem by developing an appropriate genetic algorithm in a multi-service scenario with different traffic classes. We will eventually show that the proposed solution can efficiently pack all bursts in the AMC-zone and come within only 5% of the theoretical lower bound.

This paper is structured as follows. Section II introduces the 802.16e technology and the specific zone packing problem in detail. The strip-packing problem and possible solution approaches are introduced in section III. Subsequently, section IV develops the specific solution of the strip-packing problem in 802.16e, and section V evaluates the performance of the solution approaches. Finally, section VI concludes the paper.

II. Overview of 802.16e

A. 802.16e frame structure

In 802.16e Time Division Duplex (TDD) systems, every MAC-frame is subdivided into an uplink and a downlink subframe. Both subframes are further divided into zones, allowing for different operational modes. A sample frame structure is shown in Fig. 1. Every frame begins with a mandatory PUSC-zone (Partial Use of Sub-Carriers), which contains

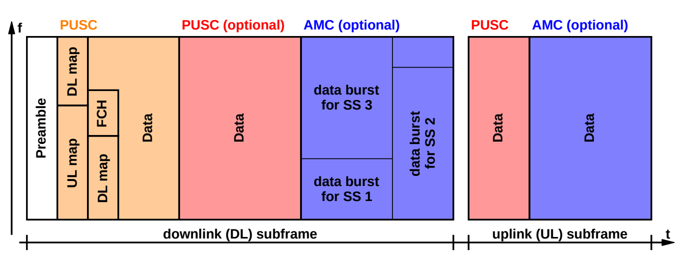

Fig. 1: Example of 802.16e frame structure

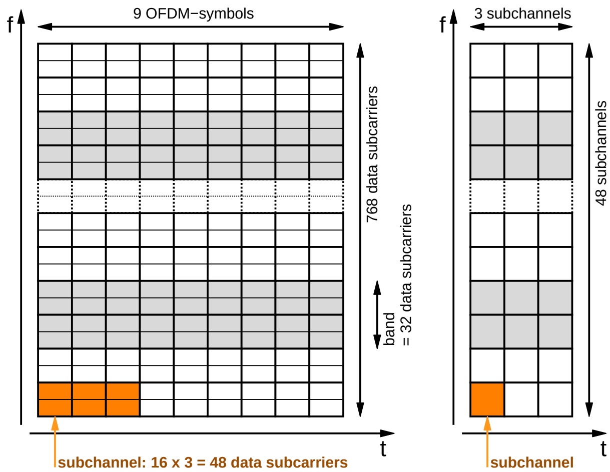

Fig. 2: Illustration of the AMC 2x3 mode

important control information. This includes the downlink and uplink maps, which describe the location and subcarrier occupation of transmission bursts for mobile terminals (also referred to as subscriber stations, SS). Commonly, one or more optional PUSC-zones follow, which contain data bursts to be transmitted with frequency diversity.

A second important zone type besides PUSC is the AMC-zone. In this zone, a set of contiguous subcarriers in frequency and OFDM symbols in time direction are allocated to one mobile terminal. The AMC-zone is particularly suited for advanced transmission techniques, such as beamforming antennas, adaptive modulation and coding, frequency-selective scheduling, or interference coordination mechanisms (see for example [4]). Figure 2 shows an example of the AMC zone layout on the left side. The figure shows the AMC 2x3 mode, which defines subchannels of 16 adjacent data subcarriers by 3 contiguous OFDM symbols. A subchannel corresponds to the resource assignment granularity for a particular SS. It is therefore possible to abstract the AMC zone by the two-dimensional resource field shown in the right part of Fig. 2, where subchannels are allocated to the different SS.

B. AMC-zone packing

Figure 3 (left) shows a possible allocation of subchannels to subscriber stations. Every subscriber station is allocated a differently sized and shaped rectangular area, as it is specified in the 802.16e standard [5]. The size depends on the amount of data to be scheduled and the present channel conditions. The shape mainly depends on the overall packing in that it must be chosen such that the resulting packing minimizes the total unoccupied space. An optimized packing is shown on the right side of Fig. 3. The bursts are compactly packed in the lower area of the plane while the empty space is aggregated at the top. This allows to place yet another large burst in the top area, which is more difficult with the fragmented empty space of the unoptimized packing.

Note that different shapes can host a different data amount. This becomes evident for SS 4 in Fig. 3, whose burst occupies 9 subchannels in the left frame and 8 subchannels in the right frame. Both is sufficient to carry a data amount that fits for example in 7 or 8 subchannels. This implies *wasted capacity* for certain shapes under a particular traffic demand.

802.16e allows to use adaptive modulation and coding (AMC) techniques. This means that the modulation scheme and code rate, which both together form the *burst profile*, may be chosen dynamically and individually for each SS depending on the present condition of the time-variant wireless channel. An SS in good channel conditions can be served with a higher order modulation scheme and a high code rate, e.g., with 64QAM 3/4. Consequently, with AMC, the size of a burst containing the same amount of data will vary depending on the channel state.

If frequency-selective packet scheduling (FSPS, see for example [6]) is used, the modulation and coding scheme and thus the burst size will even vary depending on the position in the time/frequency plane. This is due to the variation of the channel in the frequency direction in a multipath fading environment. FSPS requires detailed channel state information at the base station, which needs to be acquired via feedback or measurement signals from the SS (for example via channel sounding [5]). Moreover, the present base station generation does not support FSPS. Therefore, we will disregard this feature in the remainder of the paper. Instead, we assume AMC based on RSSI (Receive Signal Strength Indicator) and CINR (Carrier to Interference and Noise Ratio) information, which is a standard approach in 802.16e and leads to a constant burst size regardless of its position in the time/frequency plane.

C. Scheduling and service classes

802.16e allows to specify individual QoS parameters for different traffic flows. This allows to prioritize a real-time VoIP connection over a real-time streaming video, which in turn can be prioritized over a best-effort FTP download. An interactive web-session may be somewhere in-between. The scenario for the remainder of this paper will assume a number of real-time and interactive connections, which need to be served

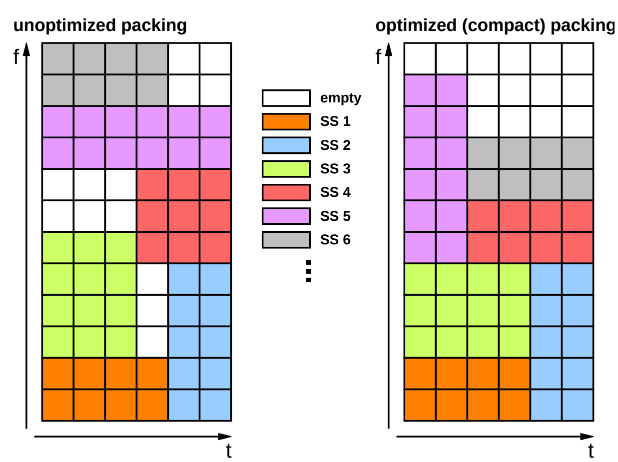

Fig. 3: Example of downlink AMC-zone packing

before other best-effort connections. The amount of data to be scheduled in one particular frame is determined in advance for the real-time and interactive connections such that the aggregated real-time and interactive traffic never exceeds the capacity of one frame. For example, it is known upfront how much data VoIP or streaming video requires in one particular frame. Once all real-time and interactive flows have been scheduled, the remaining space in the frame will be filled up with best-effort connections, which usually consist of elastic traffic, such as FTP downloads that can efficiently fill up empty space.

This scenario implies the following frame-packing problem. All bursts of real-time and interactive flows need to be packed as compact as possible (such as in Fig. 3, right side) in order to leave an as large as possible contiguous and rectangular empty space. This empty space can then easily be filled up with other rectangular bursts of the remaining best-effort flows. This problem is a variation of the well-known strip-packing problem. In the following section, we will give an overview of strip-packing and possible solution approaches to this optimization problem before presenting a specific solution for the AMC zone frame packing problem in section IV.

III. Strip-Packing and Evolutionary Algorithms

A. Classification of packing problems

In a 2-dimensional orthogonal packing problem a finite set of given rectangularly-shaped *items*, each characterized by height h_i and width w_i, has to be optimally placed in one or more bins such that the wasted unoccupied space in the bins is minimized. The items must not overlap and must not cross the bin boundaries. Items may be free-floating. Two basic variants are known: the *2-dimensional strip-packing problem* and the *2-dimensional bin-packing problem*. In the strip-packing problem only one bin with width W and infinite height H (i.e., sufficient height) has to be packed. The objective is to minimize the required height. In the bin-packing problem an infinite number (i.e., sufficient number) of bins of fixed size $W \cdot H$ are available. The objective is to minimize the number of required bins. Strip- and bin-packing problems are also known in literature as *cutting-stock problems* or *knapsack problems* [7]. Dyckhoff [8] describes problems related to *cutting and packing* and lists more than 150 annotated references.

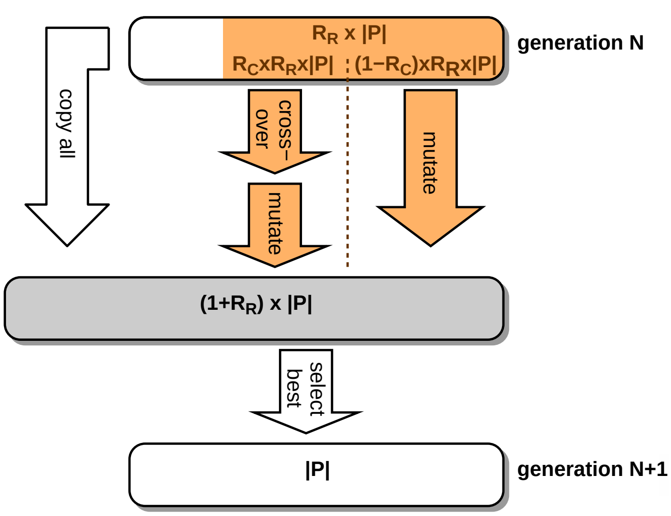

Fig. 4: Operation of the Steady State Genetic Algorithm

If the set of items is known in advance the problem is classified as an *offline* problem. Otherwise *online* algorithms are necessary that individually place the items without knowledge of number and/or shape of future items. Another major classification criteria is the *level-based* placement of items, i.e., whether items horizontally next to each other have to be placed on the same level of height or not.

Our problem is a variant of the original offline, non-level-based, 2-dimensional strip-packing problem, as we allocate rectangular areas of subchannels within one sufficiently large AMC-zone. Lodi et. al. [9] formulated an Integer Linear Program (ILP) that finds the optimal solution for a similar problem. This approach works only for small level-based problem instances, though. Several heuristics exist that find medium to good solutions (see for example [10]–[13]). We cannot apply these heuristics to our problem directly. All of the approaches assume fixed sizes and shapes of the items, some of them allow rotating or flipping the items at most. In our scenario the rectangular areas of subchannels are not predetermined in shape, as long as they cover a minimum area; i.e., we allow wasted capacity *within* the items.

Due to these properties we modified the existing approaches. We develop a genetic algorithm that takes the variable shapes of the items into account and combine it with a simple and fast heuristic (*Next-Fit-Decreasing-Height*, NFDH [9]) to obtain near optimal results. Such an approach has already been successfully employed on other variants of the strip-packing problem [14], [15], though with the above mentioned restrictions.

B. Evolutionary and genetic algorithms

It is possible to apply *evolutionary algorithms* to almost any optimization problem, in our case to find the optimal shapes and ordering of items. Roughly speaking, evolutionary algorithms perform a systematic random search of the optimum among all perceivable solutions. If the solutions are represented by an array of bits, numbers, or characters, we also speak of *genetic algorithms* with the solutions as *genomes*. The principle of evolutionary and genetic algorithms is illustrated in Fig. 4. Starting with a finite *population* of genomes, the population is evolved with each *generation* into a new and better population. This process is inspired by the biological evolution. Each new generation's population has partly, first, surviving genomes of the old population, second, *mutated* genomes, and third, the (in our case mutated) *crossover* of two genomes of the old population. A *fitness* function selects only the best of these genomes into the new generation. By favoring good genomes and/or slightly modifying them while keeping relevant structural properties a genetic algorithm is superior to pure random search. Refer to [16] or [17] for further information.

There are many variants of this basic algorithmic framework. For example, different mutation and crossover operations exist, that vary in complexity. Partial or whole populations can be replaced in each generation, even their size can change. We chose a steady-state genetic algorithm, where only a maximum fraction R_R of a population's genomes are replaced in the next generation. The size of the population remains always constant. The representation of our problem with genomes and the respective mutation and crossover operations are introduced in section IV.

IV. Application of Strip Packing to 802.16e

As explained in section II, a set of bursts $\mathbf{C}$ is selected for packing the AMC-zone. Each burst $i \in \mathbf{C}$ is characterized by its data volume c_i by means of the number of required subchannels. The AMC-zone is rectangular with W columns and H rows, i.e., $W \cdot H$ subchannels.

For each burst i, a single rectangular area within the AMC-zone must be allocated. That means, a lower left corner $(x_i,\ y_i)$ as well as an appropriate width w_i and height h_i must be selected such that the number of assigned subchannels is at least as large as the number of required subchannels ($w_i \cdot h_i \geq c_i$) and the selected area is entirely within the AMC-zone ($x_i + w_i \leq W$ and $y_i + h_i \leq H$). Furthermore, the selection of rectangles should leave the remaining unused subchannels organized in a rectangle of maximum size.

To solve this optimization problem, we apply a genetic algorithm. For this, we developed a representation of each relevant arrangement by a genome and defined mutators and crossovers operating on our genomes as well as a mapping of the solution's quality to the fitness of a genome.

A. Genome Modeling and Fitness

We model a genome as a list of *all* bursts that are selected to be placed within the current frame's AMC zone. Genomes can differ by the order of bursts as well as the shape of each burst. Accordingly, mutators and crossovers should reorder the list and/or change width and height of any number of data blocks.

To place the bursts in the AMC-zone we apply the Next-Fit-Decreasing-Height (NFDH) placement algorithm [9]. This algorithm iterates through the list of bursts and places each according to the following rules: The first burst is placed in the lower left corner. Any other burst is placed either to the right of the previously placed in the same row, if sufficient space is available and its height is less or equal to the height of the previously placed burst. Otherwise it is placed in the first column of the next free row.

Finally, we measure the fitness of a genome by the AMC-zone filling height since NFDH compacts the data blocks in the lower part of the AMC-zone.

Obviously, NFDH is not optimal by itself as some sets of bursts can never be placed with minimal resource occupancy independent of the list order. Nevertheless, we will show in section V that we can achieve near optimal results.

B. Mutation and Crossover

As already stated in our approach a genome consists of a list of bursts with assigned shape. For changing the order of the list as well as shapes we define three mutators and one crossover operator.

The *swap mutator* changes only the order of the list but not the shapes. For one swapping two bursts are picked randomly and their position in the list is exchanged. The amount of swappings inside one genome depends on the mutation rate R_M.

The *change shape mutator* changes only the shape of bursts but not their order within the list. As feasible shape we define one consisting of a width $w_i \leq W$ and a height $h_i \leq H$, which has at least c_i subchannels but no more than ξ wasted capacity units, i.e., $c_i \leq w_i h_i \leq c_i + \xi$. Among all feasible shapes a pre-selection is taken to discard shapes with extreme dimensions. From the remaining ones, one shape is selected randomly and assigned to the burst. The number of changed shapes depends analogous to the swap mutator on the mutation rate R_M.

The *swap and change shape mutator* exchanges the position of bursts in the list and simultaneously alters their shape. Again, the number of changes depends on R_M.

Finally the so-called *partial match crossover* [18] is chosen as crossover operator, which produces two new genomes out of two existing ones. After selecting a matching region with random position and length in the list, the sequence of the bursts inside the matching region is mutually exchanged. As each burst has to appear exactly ones in a genome, additional exchanges outside the matching region might be necessary. The probability of altering existing genomes by this crossover operator is given by R_C (see also Fig. 4).

C. Evaluation Metric

For evaluation of our genetic algorithm approach, we have chosen to use the ratio $\delta = F_{best}/F_{lb}$. Thereby F_{best} is the fitness of the so far best solution found by the algorithm. F_{lb} is a theoretical lower bound fitness and gives the minimum number of rows necessary for all bursts without the constraint of having rectangular shapes. With $F_{lb} = \lceil \sum_{i \in C} c_i / W \rceil$ we can easily calculate this value.

V. Performance Evaluation

A. Scenario

We consider four different service classes as shown in table I. In every traffic class, the given number of bursts needs to be scheduled in every MAC frame (precisely: in the considered AMC zone of the respective MAC frame). Every burst occupies a random number of subchannels in the range indicated in table I. The number of required subchannels per burst varies within one traffic class even though the data rate does not change due to adaptive modulation and coding. Note that the table lists the number of flows that are scheduled per frame, not the total number of flows in the system. Since a video stream is scheduled more frequently than an audio stream, it achieves a much higher data rate than the audio stream even though it requires only twice as many subchannels per frame.

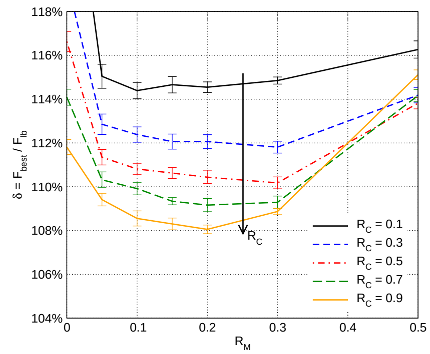

Fig. 5: Performance of *Change Shape mutator* for different R_M and R_C

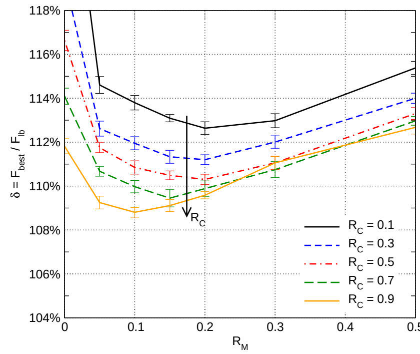

Fig. 6: Performance of *Swap mutator* for different parameters R_M and R_C

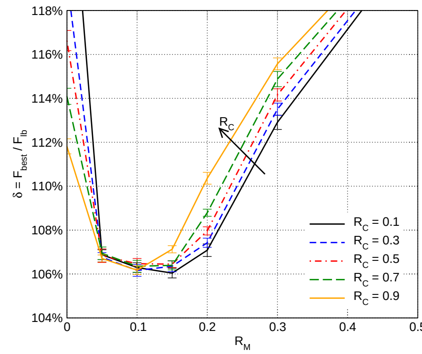

Fig. 7: Performance of *Swap and Change Shape mutator* for different R_M and R_C

As described in section II-C, the goal of the frame packing procedure is to pack the high priority service classes VoIP, streaming video, and streaming audio as efficiently as possible to leave enough contiguous space which can then be filled up by the elastic background FTP traffic.

As a stop criterion for the genetic algorithm, we choose a fixed number of generations N_{gen}. This is motivated by the fact that frame packing is a real-time problem. Consequently, a constant amount of time, and hence a constant number of generations, needs to be foreseen for it in the basestation.

B. Simulation and Optimization Environment

In order to evaluate the performance of the frame packing procedure, we perform Monte-Carlo simulations. In every Monte-Carlo drop, one frame is being packed by the genetic algorithm. For each of these frames, the number of subchannels required by a data burst is uniformly drawn from the range indicated in table I. For all results listed in the following section, a total number of 1000 frames was packed, which is sufficient to deliver excellent confidence intervals.

C. Comparison of Mutation Operators

Figures 5 through 7 show the performance of the different mutators depending on the mutation rate R_M and the crossover rate R_C. Plotted is the ratio $\delta = F_{best}/F_{lb}$ after $N_{gen} = 1000$ generations with a population size of $|P| = 100$ and a maximum wasted capacity of $\xi = 2$.

Both the Change Shape and the Swap Mutator utilize only one dimension in the possible mutation space, either changing the shape or swapping elements. The missing dimension is brought in only by the crossover operation. Consequently, the GA achieves a better performance with both mutators at higher crossover rates R_C. In contrast, the Swap and Change Shape Mutator explores both dimensions in the mutation space. A higher crossover rate increases the performance only for low mutation rates R_M, since a larger solution space can be traversed. For larger mutation rates a larger crossover rate harms the performance since newly created generations have much less in common with the original generations, which at some degree starts to contradict one basic principle of genetic algorithms. Compared to the other operators, the Swap and Change Shape Mutator achieves the best performance which is only about 6% away from the theoretical optimum.

Traffic class	Data rate [kbps]	Number of sub-channels per burst	Number of bursts per frame
VoIP	14.4	1—6	10
streaming video	300	6—32	5
streaming audio	44	3—18	5
FTP	elastic	elastic	variable

TABLE I: Traffic classes

D. Convergence and population size

One important criterion for the performance of the genetic algorithm is the time it takes to find a sufficiently good solution. Figure 8 plots the fitness $\delta = F_{best}/F_{lb}$ depending on the number of generations for the best configuration of the three mutators. With all mutators, the quality of the solution quickly decreases with an increasing number of generations, and good solutions can be obtained with only $N_{gen} = 100$ generations. While the Change Shape mutator and the Swap mutator converge quicker in the beginning, the Swap and Change Shape Mutator outperforms them for larger number of generations.

The same metric is plotted in Fig. 9 for the Swap and Change Shape mutator and different population sizes $|P|$. Naturally, the quality of the solution improves as the population size is increased. Alike, the algorithm converges faster since a larger solution space can be searched with the same number of generations. Note that for large $|P|$ and N_{gen} the lower performance bound can be reached by only 5%.

Larger $|P|$ and N_{gen} imply an increase in the computational complexity. In particular, the complexity is proportional to $N_{gen} \cdot |P|$. It is therefore of great interest to find the minimum N_{gen} and $|P|$ which deliver the best performance. Figure 10 plots the fitness δ of the best solution depending on $|P|$ and N_{gen}. The chart reveals that it is inefficient to increase either N_{gen} or $|P|$ while leaving the other parameter unchanged. Instead, the best ratio of solution quality and computational complexity can be achieved for $N_{gen} \approx |P|$.

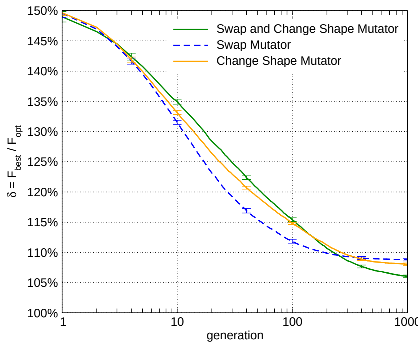

Fig. 8: Performance of mutators depending on number of generations, $|P| = 100$

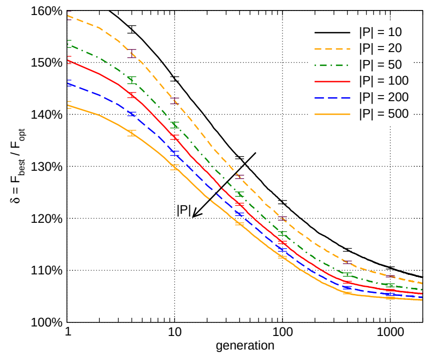

Fig. 9: Performance of *Swap and Change Shape mutator* depending on $|P|$

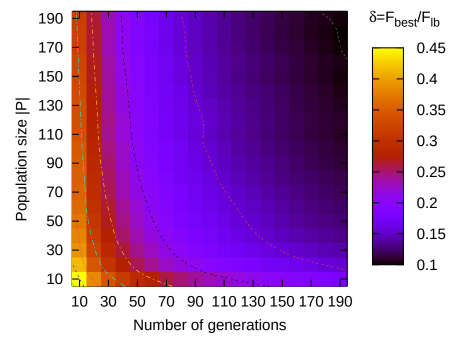

Fig. 10: Frequency Reuse 3: Mean throughput [kBit/s] in observation area

E. Complexity

Genetic algorithms are known for their relatively long computation times. In contrast to that, frame packing is a real-time problem. As we have seen in the previous section, the algorithm achieves a good performance already with 50–100 generations. Moreover, genetic algorithms have an inherent parallelism, since all mutations, crossover operations, and fitness calculations can be done in parallel. This makes them well suitable for a massively parallel hardware implementation on an FPGA or an ASIC. Several efficient hardware implementations of genetic algorithms have been reported in literature, mostly aiming at generic hardware accelerators. For our problem, we are currently developing a tailored hardware FPGA solution. We expect that a sufficient performance with respect to the quality of the packing and the computation time is possible on regular FPGA or ASIC hardware.

VI. Conclusion

Frame packing is crucial in OFDMA systems, such as 802.16e or 3GPP LTE since it has a big impact on the system performance. It is a complex problem, since a large number of data bursts needs to be placed in a two-dimensional zone as efficiently as possible. In this paper, we treated the frame packing problem in an 802.16e system as a strip-packing problem and applied genetic algorithms to solve this combinatorial optimization problem. The genetic algorithm reorders the scheduled bursts in such a way that a simple placement heuristic, such as NFDH in our case, can place the bursts as efficiently as possible. We showed that we can come within 5% of the theoretical lower bound, thus maximizing the system capacity. Our work constitutes an important step towards next-generation wireless broadband networks with maximized spectral efficiency. As frequency-selective scheduling becomes more important, future work will additionally have to consider the frequency selectivity of the wireless channel.

VII. Acknowledgments

This research was done in cooperation with Alcatel-Lucent Research and Innovation Department, Stuttgart.[1] The authors would like to thank Simon Martin for his contribution.

[1]Alcatel-Lucent Deutschland AG, Research & Innovation, Holderäckerstr. 35, 70435 Stuttgart, Germany. Contact: Michael Tangemann (Michael.Tangemann@alcatel-lucent.de).

References

[1] K. Wongthavarawat and A. Ganz, "Packet scheduling for QoS support in IEEE 802.16 broadband wireless access systems," *International Journal of Communication Systems*, vol. 16, no. 1, pp. 81–96, February 2003.

[2] H. Lee, T. Kwon, and D.-H. Cho, "An efficient uplink scheduling algorithm for VoIP services in IEEE 802.16 BWA systems," in *Proc. IEEE VTC 2004-Fall*, Los Angeles, CA, September 2004.

[3] L. Wan, W. Ma, and Z. Guo, "A cross-layer packet scheduling and subchannel allocation scheme in 802.16e OFDMA system," in *Proc. IEEE WCNC 2007)*, Kowloon, China, March 2007, pp. 1865–1870.

[4] M. C. Necker, "Local interference coordination in cellular 802.16e networks," in *Proc. IEEE VTC 2007-Fall*, Baltimore, MA, October 2007.

[5] IEEE 802.16e, *IEEE Standard for Local and metropolitan area networks, Part 16: Air Interface for Fixed and Mobile Broadband Wireless Access Systems, Amendment 2: Physical and Medium Access Control Layers for Combined Fixed and Mobile Operation in Licensed Bands*, Feb. 2006.

[6] A. Pokhariyal, T. E. Kolding, and P. E. Mogensen, "Performance of downlink frequency domain packet scheduling for the UTRAN long term evolution," in *Proc. IEEE PIMRC 2006*, Helsinki, Finland, September 2006.

[7] R. W. Haessler and P. E. Sweeney, "Cutting stock problems and solution procedures," *Journal of Operational Research*, vol. 54, no. 2, pp. 141–150, September 1991.

[8] H. Dyckhoff, G. Scheithauer, and J. Terno, "Cutting and packing," *Annotated Bibliographics in Combinatorial Optimization*, pp. 393–412, 1997.

[9] A. Lodi, S. Martello, and M. Monaci, "Two-dimensional packing problems: A survey," *European Journal of Operational Research*, vol. 141, pp. 241–252, 2002.

[10] P. E. Sweeney and E. R. Paternoster, "Cutting and packing problems: A categorized, application-oriented research bibliography," *Journal of the Operational Research Society*, vol. 43, no. 7, pp. 691–706, July 1992.

[11] E. G. Coffman, M. R. Garey, and D. S. Johnson, "Approximation algorithms for bin packing: A survey," *Approximation Algorithms for NP-Hard Problems*, pp. 46–93, 1996.

[12] A. Caprara and M. Monaci, "On the two-dimensional knapsack problem," *Operations Research Letters*, vol. 32, no. 1, pp. 5–14, Jan. 2004.

[13] N. Ntene and J. H. van Vuuren, "A survey and comparison of level heuristics for the 2D oriented strip packing problem," 2006, submitted to Applied Discrete Optimization.

[14] S.-M. Hwang, C.-Y. Kao, and J.-T. Horng, "On solving rectangle bin packing problems using genetic algorithms," in *IEEE International Conference on Systems, Man, and Cybernetics. 'Humans, Information and Technology'*, vol. 2, San Antonio, TX, USA, February 1994, pp. 1583–1590.

[15] S. Jakobs, "On genetic algorithms for the packing of polygons," *European Journal of Operational Research*, vol. 88, no. 1, pp. 165–181, January 1996.

[16] M. Srinivas and L. M. Patnaik, "Genetic algorithms: A survey," *Computer*, vol. 24, no. 6, pp. 17–26, June 1994.

[17] M. Tomassini, "A survey of genetic algorithms," in *Annual Reviews of Computational Physics III*, D. Stauffer, Ed. World Scientific, October 1995, pp. 87–118.

[18] D. E. Goldberg, *Genetic Algorithms in Search, Optimization, and Machine Learning*. Addison-Wesley Publishing Company, 1989.