Design and Implementation of High-Performance Computing Algorithms for Wireless MIMO Communications

Theses of the Ph.D. Dissertation

Csaba Máté Józsa, M.Sc.

Supervisors

Géza Kolumbán, D.Sc.
Doctor of the Hungarian Academy of Sciences
and
Péter Szolgay, D.Sc.
Doctor of the Hungarian Academy of Sciences

Pázmány Péter Catholic University
Faculty of Information Technology and Bionics
Multidisciplinary Doctoral School of Sciences and Technology

Budapest, 2015

1 Introduction

The most important driving forces in the development of wireless communications are the need for higher link throughput, higher network capacity and improved reliability. The limiting factors of such systems are equipment cost, radio propagation conditions and frequency spectrum availability. The ever increasing need for higher transmission rates motivated researchers to develop new methods and algorithms to reach the Shannon capacity limit of single transmit and receive antenna wireless systems. Research in information theory [8] has revealed that important improvements can be achieved in data rate and reliability when multiple antennas are applied at both the transmitter and receiver sides, referred to as MIMO systems [9]. The key feature of MIMO systems is the ability to turn multipath propagation, traditionally a pitfall of wireless transmissions, into a benefit for the user, thus, the performance of wireless systems is improved by orders of magnitude at no cost of extra spectrum use. The probability of error in a MIMO system can be minimized by transmitting different representations of the same data stream on different parallel transmit branches, i.e., controlled redundancy in both space and time is introduced. The capacity of the radio link in a MIMO system can be increased by transmitting independent data streams on different transmit branches simultaneously and within the same frequency band.

The complexity of MIMO detectors used over different receiver structures depends on many factors, such as antenna configuration, modulation order, channel, coding, etc. In order to achieve optimal Bit Error Rate (BER) for Additive White Gaussian Noise (AWGN) channels Maximum Likelihood (ML) detection has to be used. The exhaustive search implementation of ML detection has a complexity that grows exponentially with both the number of elements in the signal set and the number of antennas, thus, this technique is not feasible in real systems. The Sphere Detector (SD) seems to be a promising solution to reduce significantly the search space. The fundamental aim of the SD algorithm is to restrict the search to lattice points that lie within a certain sphere around a given received symbol vector. The search space reduction does not affect the detection quality because the closest lattice point inside the sphere will also be the closest lattice point for the whole lattice. The drawbacks of the SD algorithm are: (i) the complexity still suffers of an exponential growth, when increasing the number of antennas or the modulation order, (ii) the SD detection transforms the MIMO detection problem into a depth-first tree search that is highly sequential, and (iii) during every tree search several different paths have to be explored leading to a variable processing time.

When MIMO is applied to multi-user communication systems, spatial diversity can be achieved even if the Mobile Stations (MSs) are not equipped with multiple antennas. However, since the MSs do not know other users' channels, the entire processing task must be done at the Base Station (BS), especially symbol precoding to cancel multi-user interference. The approach of finding the optimal solution for detection and precoding requires a computational complexity that grows extremely high for larger MIMO systems. However, it might happen that the theoretical performance can be determined only by high complexity simulations. In this case the efficient use of Massively Parallel Architectures (MPAs) can significantly decrease the processing time.

Another approach is to precondition or preprocess the problem, and afterwards perform lower complexity signal processing algorithms (i.e. linear detection, precoding). A promising preprocessing technique that can be applied for both precoding and detection is the Lattice Reduction (LR) of the channel matrix. Recent research shows that the performance of linear and non-linear MIMO precoding and detection achieves full diversity order even with less-complex linear detection methods when used in conjunction with LR. The computational cost of LR algorithms can become critical for very large MIMO arrays. In this case the complexity or the processing time is mostly influenced by the preprocessing algorithms. In conclusion we get that the price that has to be paid when using MIMO systems is the increased complexity of hardware components and signal processing algorithms and most of these algorithms can not be efficiently mapped to modern parallel architectures because of their sequential components.

Due to the major advances in computing architectures and programming models the production of relatively low-cost, high-performance MPAs such as GP-GPUs or Field Programmable Gate Arrays (FPGAs) have been introduced. Research conducted in several scientific areas has shown that the GP-GPU approach is very powerful and offers a considerable improvement in system performance at a low cost. Furthermore, market leading smartphones have sophisticated GP-GPUs, and high-performance GP-GPU clusters are already available. Consequently, complex signal processing tasks can be offloaded to these devices. With these powerful MPAs the relatively high and variable computational complexity algorithms could be solved for real-time applications or they could speed-up the time of long running simulations.

The trend of using MPAs in several heavy signal processing tasks is visible. Computationally heavy signal processing algorithms like detection [10], [11], decoding [12], [13] and precoding [14] are efficiently mapped on to GP-GPUs.

The underlying architecture is seriously influencing the processing time and the quality of the results. Since the existing algorithms are mostly sequential, it is necessary to redesign completely the existing algorithms in order to achieve peak performance with the new MPAs. By using these powerful devices new limits are reached, so in this thesis my goal is twofold: (i) to design efficient and highly parallel algorithms that solve the high complexity ML detection problem and (ii) to design and implement highly parallel preconditioning algorithms, such as lattice reduction methods that facilitates the use of low complexity signal processing algorithms without degrading significantly the overall system performance.

2 Methods used in research

The goal of my research was to solve computationally demanding signal processing problems in the field of wireless communications with modern MPAs, such as GP-GPUs, and multi-core CPUs. The main challenge was to identify and develop the *mathematical and algorithmic transformations* of the sequential, high-complexity problems in such a way that an efficient mapping to these parallel architectures became possible.

In the first part of my thesis I consider the optimal hard-output *ML detection* in MIMO systems. The complexity of the ML detector increases exponentially with the number of antennas and the modulation order. In order to significantly reduce the complexity, the *SD algorithm* was proposed in [15] and applied in a decoding context in [16]. The fundamental aim of the SD algorithm is to restrict the search to lattice points that lie within a certain sphere around a given received symbol vector. The search space reduction does not affect the detection quality because the closest lattice point inside the sphere will also be the closest lattice point for the whole lattice.

During detection the optimum search path for the symbol vectors is different. Since different parts of the search tree are explored by the detection algorithm, a variable processing time is expected. In order to moderate the effects of the variable complexity (i) *a column norm based ordering method* shown in [17] and (ii) *a dynamic computing load distribution* strategy were applied. Specifying the order of symbol detection, based on metrics involving the channel matrix, was shown to lead to less computations. The probability of choosing the right search path on the top levels of the tree can be increased by first detecting symbols with higher post-detection SNR or SINR. Consequently, a non-optimal symbol detection on a lower level does not lead to a major step-back on the tree.

The variable processing time of the symbol vectors leads to an imbalance in the execution time of the thread blocks of a kernel. Until the execution of the thread blocks is not finished the GP-GPU resources allocated to a kernel will not be freed. The long-time resource allocation prevents the overlapping execution of multiple kernels on different streams. With a dynamic load balancing method the tail effect is negligible, thus, the overlapping execution of multiple kernels becomes possible and the goal of alleviating the variable processing time is achieved.

In the second part of my thesis the focus is on a powerful preprocessing tool, namely the *LR method* [18]. Lattice reduction aims to find a ”better” basis whose vectors are more orthogonal and shorter than the original ones, in the sense of Euclidean norm. Lattice reduction improves the condition number, the orthogonality defect and the Seysen measure. Several LR algorithms exist in the literature that differ in computational complexity and achieved performance. However, the most extensively used polynomial-time algorithm is the *Lenstra-Lenstra-Lovász (LLL) algorithm* introduced in [19]. Because of its wide applicability and several favorable properties, my research focused on improving this method and making it suitable for MPAs.

In [20] it was shown that the performance of *linear and non-linear detectors* can be improved when used in conjunction with LR techniques and full diversity order is achieved with the reduced basis. Since many detection schemes heavily rely on the usage of the channel matrix, it is straightforward to regard the channel matrix as a lattice generator matrix.

In MISO systems the multi-user interference must be canceled at the transmitter, this method is referred to as *precoding*. According to [21] linear methods, such as Zero-Forcing precoding, and non-linear methods, such as Tomlinson-Harashima precoding and vector perturbation techniques perform better if the channel matrix is not badly conditioned. Moreover, full diversity is achieved even for very large systems.

The tools used to solve the above mentioned computationally challenging signal processing tasks were modern *multi-core CPUs*, such as Intel Core i7-3820, Intel Xeon X5680, Intel Xeon E5-2650 v3, and *massively parallel architectures*, such as NVIDIA GeForce GTX 690 and NVIDIA Tesla C2075 and K20 GP-GPUs. A number of parallel programming models were employed to support the hierarchical parallelism present in modern computer systems. For *coarse-grained shared memory parallelism* I used simultaneous multithreading using *OpenMP*. For *fine-grained parallelism* Single Instruction Multiple Threads (SIMT) was implemented using *CUDA*.

3 New scientific results

Thesis group I. Design of new parallel Sphere Detector algorithms achieving hard-output true-ML performance and their efficient mapping to multi-core and many-core architectures.

(Related articles [1], [3].)

Thesis I.a.

I proposed a new Parallel Sphere Detector (PSD) algorithm to achieve true-ML bit error rate performance in hard-output MIMO detection. The high degree of parallelism of the PSD algorithm is based on a novel hybrid tree traversal where depth-first search and breadth-first search methods are efficiently combined, furthermore, at each intermediate stage, path metric based parallel sorting networks are employed to achieve a faster convergence. I showed that the PSD algorithm achieves an efficient work distribution in a highly multi-threaded environment reducing the number of visited tree nodes by a single thread with $88\% - 96\%$*, and the speed-up factor of the detection throughput of the PSD algorithm in* 4×4 *MIMO systems is* $2 - 50$ *times higher for different signal-to-noise ratios compared to the sequential case.*

The real-valued MIMO system model is described as

$$\mathbf{y} = \mathbf{H}\mathbf{s}_t + \mathbf{v}$$

where $\mathbf{y} \in \mathbb{R}^M$ is the received symbol vector, $\mathbf{v} \in \mathbb{R}^M$ is the additive channel noise, $\mathbf{s}_t \in \Omega^N$ is the transmitted symbol vector, Ω is the symbol set, and the superposition of the transmitted symbols is modeled by the channel matrix $\mathbf{H} \in \mathbb{R}^{M \times N}$. The optimal hard-output ML detector is defined as

$$\hat{\mathbf{s}}_{ML} = \underset{\mathbf{s} \in \Omega^N}{\arg\min} ||\mathbf{y} - \mathbf{H}\mathbf{s}||^2.$$

The ML estimate of the transmitted symbol vector is found by solving an integer least-squares problem which is analogous to finding the closest lattice point of lattice $\mathbf{\Lambda} = \{\mathbf{H}\mathbf{s} | \mathbf{s} \in \Omega^N\}$ to a given point $\mathbf{y}$.

With the unconstrained least-squares solution $\hat{\mathbf{s}} = \mathbf{H}^\dagger \mathbf{y}$, where $\mathbf{H}^\dagger$ denotes the Moore–Penrose pseudoinverse, and the QR factorization of the channel defined as $\mathbf{H} = \mathbf{QR}$, the ML detection problem can be reformulated as

$$\hat{\mathbf{s}}_{ML} = \arg\min_{\mathbf{s}\in\Omega^N} ||\mathbf{R}(\mathbf{s} - \hat{\mathbf{s}})||^2.$$

The lattice point $\mathbf{Hs}$ is included by the sphere $S(\mathbf{y}, d)$ with center point $\mathbf{y}$ and radius d if the following inequality is satisfied $||\mathbf{R}(\mathbf{s} - \hat{\mathbf{s}})||^2 \leqslant d^2$,

$$\left\| \begin{pmatrix} r_{11} & r_{12} & \cdots & r_{1N} \\ 0 & r_{22} & \cdots & r_{2N} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & r_{NN} \end{pmatrix} \begin{pmatrix} s_1 - \hat{s}_1 \\ s_2 - \hat{s}_2 \\ \vdots \\ s_N - \hat{s}_N \end{pmatrix} \right\|^2 \leqslant d^2.$$

Starting with dimension N, the elements of the symbol set are inserted into the partial symbol vector and the inequality condition is evaluated. The search process is analogous to a depth-first tree search that is highly sequential, consequently, this problem cannot be efficiently solved in a multi-threaded environment.

The PSD completely eliminates the sequential parts of the SD algorithm. The tree traversal of the PSD algorithm is implemented by a novel hybrid tree search method, where the algorithm parallelism is assured by the efficient combination of depth-first search and breadth-first search algorithms. Because of the hybrid tree search only distinct levels of the tree are evaluated that are denoted by the parameter lvl_x. On these levels the number exp_{lvl_x} of partial symbol vectors are expanded simultaneously. During the expansion of a partial symbol vector $(lvl_{x-1} - lvl_x)$ number of new symbols are added to the original symbol vector. The simultaneous expansion of $exp_{lvl_{x-1}}$ number of partial symbol vectors on level lvl_{x-1} will create $eval_{lvl_x} = exp_{lvl_{x-1}} \cdot |\Omega|^{(lvl_{x-1} - lvl_x)}$ number of new partial symbol vectors on level lvl_x. Note, that a hybrid search is realized at this point, because with parameters exp_{lvl_x} the extent of the breadth search, while with parameters lvl_x the extent of the depth search is controlled. Since several new (partial) symbol vectors are created after the expansion stage, the parallel path metric update becomes possible, thus, the resources of an MPA can be efficiently exploited.

Figure 1 shows the average number of expanded nodes per thread for different MIMO systems and symbol set configurations. The signal space of real-equivalent 8×8 MIMO with symbol set size of $|\Omega| = 8$ has about 1.6×10^7 symbol vectors. For an SNR of 5 dB the PSD expands into about 310 nodes per thread while the Automatic Sphere Detector expands into

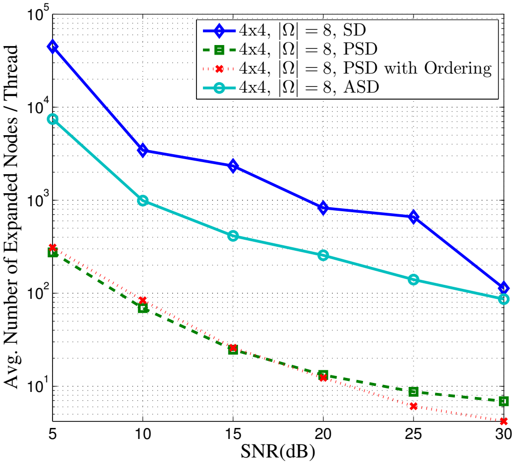

Figure 1: Comparison of the average number of expanded nodes per thread for 4×4 MIMO systems and $|\mathbf{\Omega}| = 8$ for the sequential, parallel and automatic Sphere Detector algorithms.

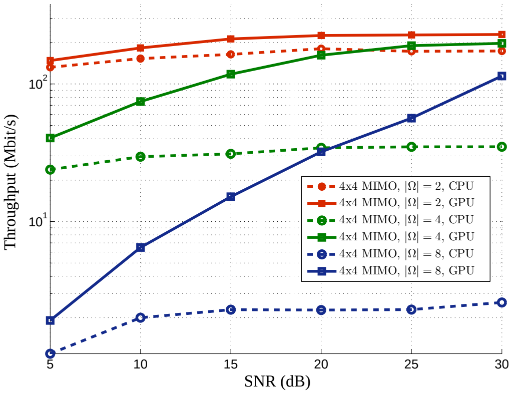

Figure 2: The comparison of the average detection throughput of (i) the Parallel Sphere Detector algorithm implemented on a GP-GPU architecture and (ii) the sequential Sphere Detector executed on every thread of a multi-core CPU.

about 7500 nodes per thread. Consequently, the total workload of a thread running the PSD algorithm is reduced by 96%.

In Fig. 2 the average detection throughput achieved with (i) the PSD algorithm implemented on the GTX690 GP-GPU and (ii) the sequential SD executed simultaneously on every thread of an Intel Xeon CPU E5-2650 v3 was compared. At 30 dB SNR for a 4×4 MIMO and $|\Omega| = 4$ the detection throughput is increased 6 times, and for $|\Omega| = 8$ the throughput is increased 50 times by the GP-GPU.

Thesis I.b.

I defined highly parallel, dynamic building blocks for the Expansion and Evaluation pipeline of the PSD algorithm as a function of available parallelism. Based on the building blocks, I identified a set of parameters that determine the extent of parallelism and memory footprint. I showed that the achieved average detection throughput of the GP-GPU mapping outperformed every existing true-ML detector and many non-ML GP-GPU, ASIC, DSP and FPGA implementations.

Throughout the detection process the most heavily used operations are the vector expansion and evaluation. In order to remove every possible bottleneck and to make a parallel implementation possible, I have introduced a the Expansion and Evaluation pipeline (EEP). The stages of the EEP are defined as: (i) the *Preparatory Block*, (ii) the *Selecting, Mapping and Merging Block*, (iii) the *Path Metric Evaluation Block*, and (iv) the *Searching or Sorting Block* as shown in Fig. 3.

In the *Preparatory Block* virtual identifiers are computed simultaneously by tt number of threads where the k-th thread is denoted by t_{id}^k. The virtual identifiers are computed in the following manner:

$$\begin{aligned} VT_{lvl_x}^k = \{vt_{lvl_x} | vt_{lvl_x} = (t_{id}^k + n \cdot tt) \mod |\Omega|^{(lvl_{x-1} - lvl_x)}, \\ n = 0 : \lceil eval_{lvl_x}/tt \rceil - 1\}, \end{aligned}$$

$$\begin{aligned} VB_{lvl_x}^k = \{vb_{lvl_x} | vb_{lvl_x} = \lfloor (t_{id}^k + n \cdot tt)/|\Omega|^{(lvl_{x-1} - lvl_x)} \rfloor, \\ n = 0 : \lceil eval_{lvl_x}/tt \rceil - 1\}. \end{aligned}$$

In the *Selecting, Mapping and Merging* block previously evaluated partial symbol vectors are selected and further expanded. In the *Selecting* phase, previously evaluated partial symbol vectors $s_{lvl_{x-1}}^N$ are selected based on the thread's virtual block identifiers $vb_{lvl_x} \in VB_{lvl_x}^k$. In the

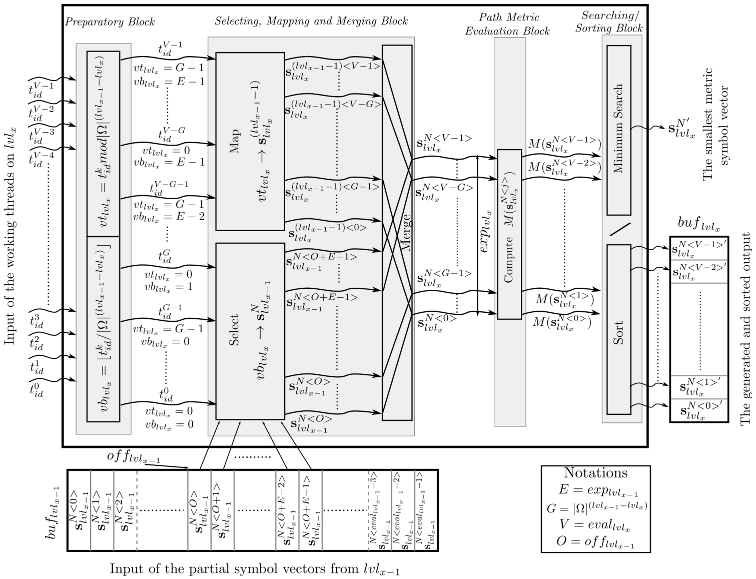
Preparatory Block
Selecting, Mapping and Merging Block
Path Metric Evaluation Block
Searching/ Sorting Block
Input of the working threads on lvl_x
$vt_{lvl_x} = t^k_{id} mod |\Omega|^{(lvl_{x-1}-lvl_x)}$
$vb_{lvl_x} = \lfloor t^k_{id} / |\Omega|^{(lvl_{x-1}-lvl_x)} \rfloor$
Map
$vt_{lvl_x} \rightarrow \mathbf{s}_{lvl_x}^{(lvl_{x-1}-1)}$
Select
$vb_{lvl_x} \rightarrow \mathbf{s}^N_{lvl_{x-1}}$
Merge
Compute $M(\mathbf{s}^{N<?>}_{lvl_x})$
exp_{lvl_x}
Minimum Search
Sort
$\mathbf{s}^{N'}_{lvl_x}$
The smallest metric symbol vector
buf_{lvl_x}
The generated and sorted output
$buf_{lvl_{x-1}}$
$off_{lvl_{x-1}}$
Input of the partial symbol vectors from lvl_{x-1}
Notations
$E = exp_{lvl_{x-1}}$
$G = |\Omega|^{(lvl_{x-1}-lvl_x)}$
$V = eval_{lvl_x}$
$O = off_{lvl_{x-1}}$

Figure 3: The *Expand and Evaluate* pipeline of the PSD algorithm.

Mapping phase the virtual thread identifiers $vt_{lvl_x} \in VT^k_{lvl_x}$ are mapped to $\mathbf{s}_{lvl_x}^{lvl_{x-1}-1}$ partial symbol vectors. Finally, in the *Merging* phase each selected vector $\mathbf{s}^N_{lvl_{x-1}}$ and mapped symbol vector $\mathbf{s}_{lvl_x}^{lvl_{x-1}-1}$ is merged as $\mathbf{s}^{N<j>}_{lvl_x} = (\mathbf{s}_{lvl_x}^{lvl_{x-1}-1<j>}, \mathbf{s}^{N<j>}_{lvl_{x-1}})$.

In the *Path Metric Evaluation* block, the path metric of the expanded partial symbol vectors is updated. This is one of the most time-consuming steps, however, to reduce the time required the path metrics are updated in parallel by several threads.

The *Searching or Sorting* block of the EEP is one of the most important stages during the detection. Depending on the level of processing, either sorting or a minimum search is performed. The minimum search is applied only when the detection has reached the last processing level, while sorting is applied on all other levels. The sorting is done with the use of sorting networks. Due to their data-independent structure, their operation sequence is completely rigid. This property makes this algorithm parallelizable for the GP-GPU architecture. The minimum search algorithm relies on the parallel prefix sum algorithm.

As a result, I elaborated a highly parallel expansion and evaluation pipeline where no frequent thread synchronization is required. This enables a very efficient utilization of an MPA. I compared the average detection throughput of the PSD algorithm achieved with optimal ML implementations known from the literature. The PSD algorithm outperformed each of them. Further comparison was made with non-optimal FPGA, DSP, ASIC and GP-GPU implementations. The average detection throughput of the PSD was better in the majority of cases. Although, some FPGA and VLSI based non-optimal detectors showed a better performance, but those solutions suffer from a loss in BER performance.

Thesis I.c.

I proposed a dynamic computing load scheduling algorithm that combines in a very efficient manner the system level and device level parallelism. The result of the elaborated scheduling is a dynamic binding between the symbol vectors and the thread blocks, that allows to configure grids with significantly less thread blocks. By reducing the size of the grids, the resources of the streaming multiprocessors are shared between several grids, thus, the concurrent executions of kernels on multiple streams are enhanced. Thereby, the idle time of the processing units, caused by the variable complexity of the symbol detection, is minimized and the average detection throughput achieved is increased.

The *system level* parallelism is implemented by the parallel processing of fading blocks of a received frame. Consequently, the number of kernels launched is equal to the number of independent channel realizations. Every grid assigned to a kernel launches several thread blocks (TBs) and the detection of the symbol vectors associated to one channel realization is done by the threads of the TBs. The configuration of the grids, namely the binding of the TBs and symbol vectors, is critical since this influences the concurrent execution of the kernels.

A straightforward binding requires a high number of TBs, because the resources of the GP-GPU will be available for a long time duration only for one kernel, thus, the concurrent execution of the kernels of different streams is limited. By reducing the number of TBs and keeping the load constant, the varying detection time of the different symbol vectors could amplify the tail effect. This means that only a few TBs of a grid are working and the resources of the streaming multiprocessors are not

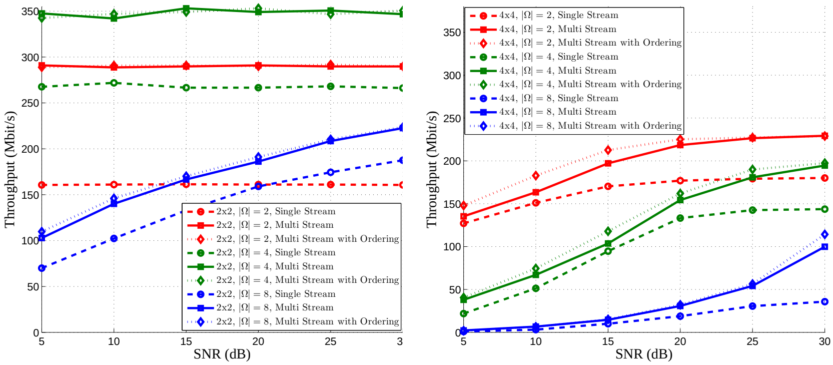

Figure 4: The Parallel Sphere Detector average detection throughput for (a) 2 × 2, (b) 4 × 4 MIMO systems obtained with single stream and multiple stream kernel executions.

freed up.

In the proposed dynamic computing load scheduling algorithm the number of TBs in a grid is significantly smaller compared to the straightforward binding case. The work for a TB is dynamically distributed, namely, when the detection of one symbol vector is finished, the PSD algorithm executed by the threads of the TB evaluates the next unprocessed symbol vector. By means of this technique, the tail effect introduced by the varying processing time of different symbol vectors is balanced. As a result, the *device level* parallelism, namely, the concurrent execution of multiple kernels on different streams, is enhanced.

The effect of dynamic computing load scheduling is shown in Fig. 4 for a 2×2 and a 4×4 MIMO system where the size of the symbol sets are $|\Omega| = 2$, 4 and 8. An increase of $15\% - 30\%$ for $|\Omega| = 2$, 4 and $38\% - 64\%$ for $|\Omega| = 8$ of average detection throughput have been achieved.

Thesis group II. Channel preprocessing techniques for true-ML hard-output MIMO detection.

(Related articles [1], [3].)

Thesis II.a.

I experimentally proved that the computational complexity of the PSD algorithm is reduced considerably by defining the detection order based on the inverse channel row norms. The aim of ordering is to detect symbols with lower signal strength on levels where a full breadth-first search is performed. This approach maximizes the probability that the best path metric partial symbol vector is the optimal choice on these levels. I showed that the applied inverse channel based row norm ordering increases the average detection throughput and decreases the number of expanded nodes.

Detectors based on successive interference cancellation are seriously influenced by the order of detected symbols. In case if the detected symbol is different from the symbol sent then symbol cancellation introduces noise instead of lowering the number of interferers. Several ordering metrics have been introduced in the literature [17]. The most important ordering metrics are based on the (i) signal-to-interference plus noise ratio (SINR), (ii) signal-to-noise ratio (SNR), and (iii) channel matrix column norms.

The metrics based on SINR and SNR involve complex computations. A simpler metric based on the column norms of the channel matrix can be represented as:

$$\mathbf{y} = \mathbf{H}\mathbf{s}_t + \mathbf{v} = \mathbf{h}_1 s_1 + \mathbf{h}_2 s_2 + \cdots + \mathbf{h}_n s_n + \mathbf{v} \tag{1}$$

where $\mathbf{h}_i$ represents the i-th column of the channel matrix $\mathbf{H}$. The ordering metric is based on the norms of the column vectors $||\mathbf{h}_i||$. As a result, the received signal strength is proportional with the ordering metric.

Algorithms based on successive interference cancellation require to detect the strongest symbols first. However, the PSD starts the detection process with the symbols having the lowest metric, because at the top of the tree a full breadth-first search is performed and the search is continued with the best path metric symbol vectors. Since every possibility is examined the error probability introduced by the lower signal strength is minimized.

The effect of matrix preprocessing based on decreasing ordering of the norms of the row vectors of the inverse channel matrix was evaluated. By applying channel preprocessing an extra increase of $5 - 10\%$ in average detection throughput was achieved, as shown in Fig. 4.

Thesis group III. Complexity reduced parallel Lattice Reduction algorithms mapped to massively parallel and heterogeneous platforms.

(Related articles [2], [4], [5].)

Thesis III.a.

I proposed a parallel Cost-Reduced All-Swap LLL (CR-AS-LLL) lattice reduction algorithm where the cost reduction consists in delaying the update of the off-diagonal Gram-Schmidt coefficients when the size reductions and column swaps are performed. I elaborated a GP-GPU mapping of the CR-AS-LLL algorithm relying on a two-dimensional thread block configuration. I showed that efficient work distribution, memory access, inner product and size reduction computation are achieved with the proposed mapping. The average computational time of the GP-GPU mapping achieves one order of magnitude improvement compared to the multi-core CPU mapping.

After every size reduction or column swap the Gram-Schmidt coefficients are updated in the original parallel All-Swap LLL algorithm. However, a lot of unnecessary computations are performed, because the frequent size reductions and column swaps change the value of the Gram-Schmidt coefficients several times. In the proposed CR-AS-LLL algorithm only the $\mu_{k,k-1}$ Gram-Schmidt coefficients are updated regularly because the evaluation of the LLL conditions depend only on these parameters. The rest off the coefficients are updated after finishing the swaps and size reductions operations.

When mapped to GP-GPU, the performance of the CR-AS-LLL algorithm depends on the efficiency of the *work distribution* among the available GP-GPU threads and the implementation of the most frequently used operations, such as *dot products*, *size reductions* and *column swaps.* Figure 5 presents a possible mapping for the main parts of the CR-AS-LLL algorithm. The kernel is launched with a one dimensional *grid* whose size is determined by the number of lattice basis processed simultaneously. The thread blocks $\mathrm{TB}(T_x, T_y)$ launched have a two dimensional configuration, where T_x and T_y denote the number of threads in the x and y dimension. The number of threads T_y is defined based on the size of the original basis, i.e., $T_y = \min(n/2, 32)$. By enabling the usage of $T_x = \min(n, 32)$ threads in the x dimension, the threads that belong to

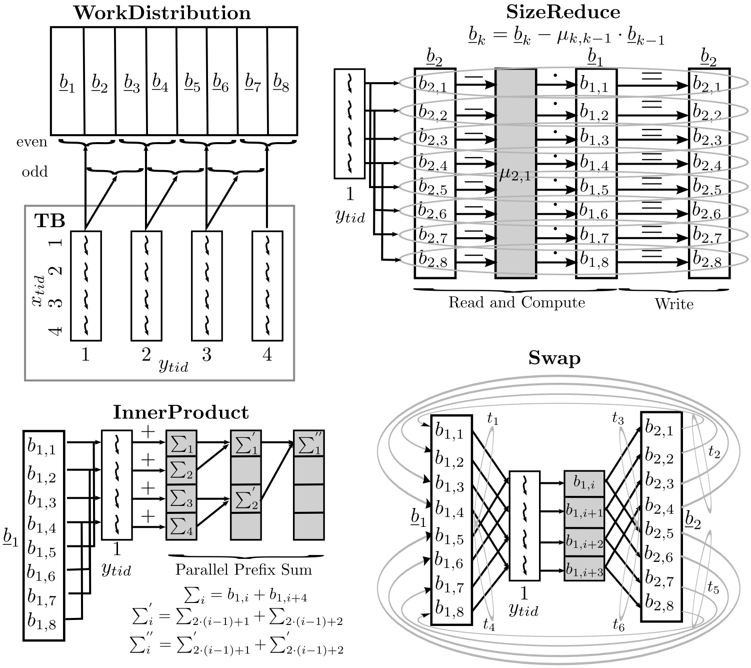
WorkDistribution
$\underline{b}_1$ $\underline{b}_2$ $\underline{b}_3$ $\underline{b}_4$ $\underline{b}_5$ $\underline{b}_6$ $\underline{b}_7$ $\underline{b}_8$
even
odd
TB
x_{tid}
1 2 3 4
y_{tid}
SizeReduce
$\underline{b}_k = \underline{b}_k - \mu_{k,k-1} \cdot \underline{b}_{k-1}$
$\underline{b}_2$
$\underline{b}_1$
$b_{2,1}$ $b_{2,2}$ $b_{2,3}$ $b_{2,4}$ $b_{2,5}$ $b_{2,6}$ $b_{2,7}$ $b_{2,8}$
$\mu_{2,1}$
$b_{1,1}$ $b_{1,2}$ $b_{1,3}$ $b_{1,4}$ $b_{1,5}$ $b_{1,6}$ $b_{1,7}$ $b_{1,8}$
Read and Compute
Write
InnerProduct
$\underline{b}_1$
$\sum_1$ $\sum_2$ $\sum_3$ $\sum_4$
$\sum_1^{'}$ $\sum_2^{'}$
$\sum_1^{''}$
Parallel Prefix Sum
$\sum_i = b_{1,i} + b_{1,i+4}$
$\sum_i^{'} = \sum_{2\cdot(i-1)+1} + \sum_{2\cdot(i-1)+2}$
$\sum_i^{''} = \sum_{2\cdot(i-1)+1}^{'} + \sum_{2\cdot(i-1)+2}^{'}$
Swap
t_1 t_2 t_3 t_4 t_5 t_6
$\underline{b}_2$
$b_{1,i}$ $b_{1,i+1}$ $b_{1,i+2}$ $b_{1,i+3}$

Figure 5: The high-level work distribution among the GP-GPU threads and the mapping of the size reduction, inner product and column swap operations for the Cost-Reduced All-Swap LLL lattice reduction.

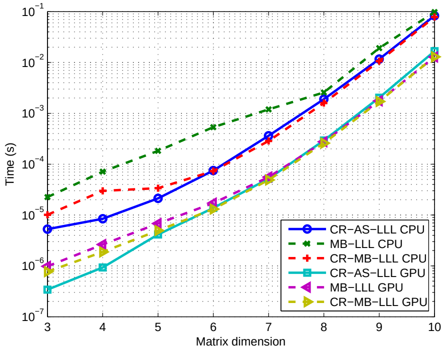

Figure 6: Computational time of algorithms CR-AS-LLL, MB-LLL and CR-MB-LLL for matrix dimensions $2^3 - 2^{10}$.

the same y dimension will form a *warp*. Consequently, the elements of matrices $\mathbf{B}, \mathbf{B}^*$ stored in the global memory are accessed through the coalesced memory pattern exploiting the available memory bandwidth. The size of the low latency *shared memory* is limited. Thus, only those Gram-Schmidt coefficients are stored in this memory which are required to evaluate the LLL conditions. Shared memory also plays an important role in computing the dot products and in the column swap procedures.

Figure 6 compares the average computational time of the CR-AS-LLL mapped on a GP-GPU and a CPU. The GP-GPU outperforms the CPU for every matrix dimension with speed-up ranging from 6 to 15.

Thesis III.b.

I proposed the Cost-Reduced Modified-Block LLL (CR-MB-LLL) algorithm where two levels of parallelism are identified and exploited enhancing the lattice reduction of higher dimensional lattice basis. The higher level parallelism follows the block reduction concept where the original lattice basis is divided into several smaller sized sub-matrices and, on the lower level, the parallel lattice reduction of the sub-matrices is done by the CR-AS-LLL algorithm. I showed that for large matrices the CR-MB-LLL algorithm is more efficient than the CR-AS-LLL algorithm.

The problem division to several sub-problems that can be executed concurrently can be regarded as one level of parallelism. In addition, if a sub-problem could benefit from a multi-threaded environment it can be regarded as a second level of parallelism. Previous parallel LR implementations have focused only on multi-core architectures. The main drawback of the low number of threads offered by modern CPUs (compared to GP-GPUs) is that low-level parallelism cannot be exploited in an efficient manner. During the algorithm design, low-level parallelism is usually omitted and the levels of parallelism are also restricted. In case of GP-GPUs, the high number of CUDA cores makes the parallel execution of a high number of threads possible offering significant performance improvements.

The CR-MB-LLL algorithm is designed to exploit the benefits of a highly multi-threaded environment. The CR-MB-LLL algorithm splits the original basis into several sub-problems with lower dimension and performs parallel LLL reduction on them. Because the LLL reduction of the subgroups and the boundaries check can be done independently, no frequent synchronization is required. Thus, coarse grained parallelism is achieved by creating the sub-problems. The GP-GPU mapping of the CR-MB-LLL algorithm is similar to the one presented in case of the CR-AS-LLL algorithm, because the procedures used are performed with a two dimensional TB configuration even in the case of a boundary check.

The CR-MB-LLL algorithm reduces further the computational complexity of the MB-LLL algorithm. In the MB-LLL algorithm, the submatrices affected by boundary swap have to be LLL reduced and the Gram-Schmidt coefficients have to be updated. The complexity reduction in the CR-MB-LLL algorithm is achieved by eliminating the GS coefficients update in the submatrices after the execution of the CR-AS-LLL and with the simplified swap procedure.

As shown in Fig. 6, the computational time of the CR-MB-LLL is $25-40\%$ lower in case of small and medium-sized matrices compared to the MB-LLL algorithm. Furthermore, the block concept implemented in the CR-MB-LLL achieves 30% speed-up for large matrices compared to the CR-AS-LLL.

Thesis III.c.

I proposed a heterogeneous platform and a suitable mapping for the Cost-Reduced Modified-Block LLL algorithm where the scheduling of kernels is implemented by a CPU and the processing tasks are executed by GP-GPU kernels. I compared the performance of the proposed heterogeneous platform with a dynamic parallelism based GP-GPU mapping and a parallel CPU implementation. I showed that the average computational time is better by one order of magnitude for smaller and middle sized matrices when a heterogeneous platform is used.

The schematic of the heterogeneous platform is shown in Fig. 7. The CPU threads launch (i) the *CR-AS-LLL* kernels in order to LLL reduce the sub-matrices, (ii) the *Boundary Check* kernels for checking the LLL conditions at the boundaries of the sub-groups and (iii) the *Coefficients Update* kernel to update the Gram-Schmidt coefficients and to perform the size reductions wherever it is required.

The control logic of the dynamic scheduling is implemented by the CPU threads. A different CUDA stream is assigned for every CPU thread, making the concurrent kernel execution possible and reducing the idle time of the CUDA cores. The status of the sub-matrices is updated continuously in the GP-GPU global memory and it is communicated to the CPU, thus, the size of the grids assigned to *CR-AS-LLL*

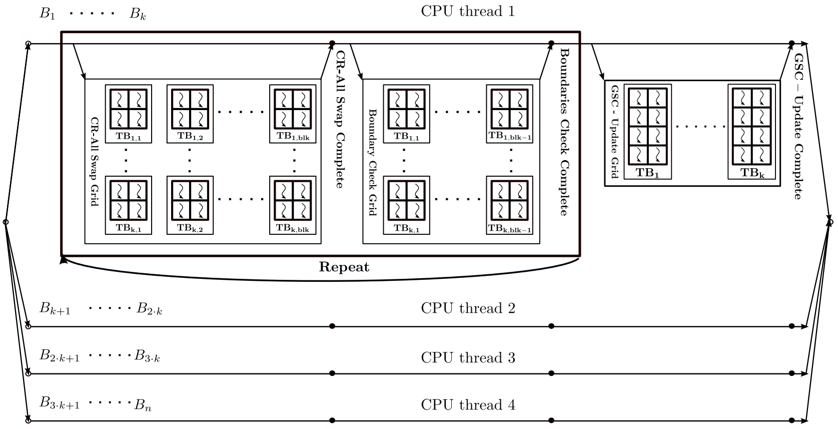

Figure 7: Kernels scheduling on the heterogeneous platform for the Cost-Reduced Modified-Block LLL lattice reduction algorithm.

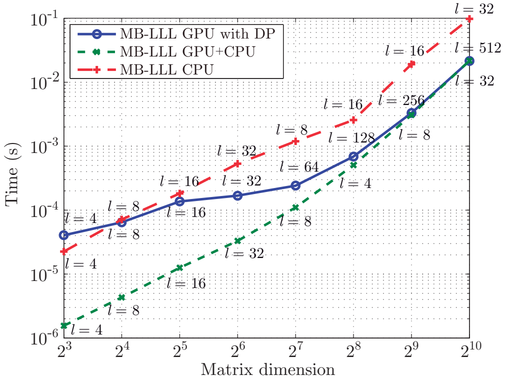

Figure 8: Computational time of the MB-LLL algorithm on different architectures, where l denotes the size of the processed blocks.

and *Boundary Check* kernels is dynamically adjusted according to the number of modified sub-matrices in every iteration. The *Coefficients Update* kernel starts after all the matrices assigned to one CPU thread are completely processed.

Figure 8 shows the computational times of the MB-LLL algorithm based on three different architectures for different matrix dimensions. The performance was evaluated on a Tesla K20 GP-GPU and an Intel Core i7-3820 processor. The heterogeneous platform clearly outperforms the solutions based on dynamic parallelism in the case of small matrices and the CPU for all of the cases. The processing times show similar performance for large matrices when the GP-GPU is involved. The conclusion is that the data transfer between CPU and GP-GPU required by the heterogeneous system is less time consuming than the overhead of the kernel launch with dynamic parallelism and the limitation of the concurrent execution of kernels on different streams.

4 Application of the results

Lattice reduction is a powerful concept for solving diverse problems involving point lattices. It is a topic of great interest, both as a theoretical tool and as a practical technique. Since point lattices and lattice reduction plays a key role in numerous fields of applications, my goal was to enhance the performance of the polynomial-time LLL lattice reduction algorithm.

The results presented in Thesis group III. prove that my goal was successfully achieved, since I reduced the complexity of the LLL algorithm, I identified and exploited several levels of parallelism that lead to efficient algorithm mapping to different parallel architectures and heterogeneous platforms. By exploiting the resources of this powerful architectures the processing time of the LR was significantly decreased. The following enumeration gives a brief summary where the results of Thesis group III. can be applied.

- In the field of *wireless communications* my results could enhance: (i) the equalization of frequency-selective channels [22], (ii) the equalization in precoded orthogonal frequency division multiplexing systems [23], (iii) the source and channel coding in scenarios with multiple terminals [24], and the preprocessing of sphere decoding [25]. When used in conjunction with LR methods, lower complexity linear and non-linear detection and precoding methods achieve full diversity order [20], [21]. The computational complexity of these methods is mostly determined by the preprocessing LR algorithm, however, my results presented in Thesis group III. significantly reduce the complexity of the LLL algorithm achieving a better processing time.

- My results can be applied in the field of *image processing* for improving the speed of radar imaging, magnetic resonance imaging and color space estimation in JPEG images as shown in [26] and [27].
- In the field of *combinatorial mathematics* it is possible to phrase many different problems as questions about lattices. Lattice problems arise in integer programming [28], subset sum problems [29], factoring polynomials with rational coefficients [19], and diophantine approximation just to name a few of them. My results presented in Thesis group III. could speed-up the solution of these problems.
- As shown in [30] methods based on LR have been used in *cryptography* where the processing time has a critical role.

Research in information theory has revealed that important improvements can be achieved in data rate when multiple antennas are applied at both the transmitter and receiver sides [8]. Unfortunately, with the increased performance the complexity of the associated signal processing problems is also increased. The complexity of the optimal ML detection in MIMO systems increases exponentially with the number of transmit antennas and modulation order, thus, its use in practical systems is prohibitive. The SD algorithm was developed and refined in [15], [29], [25] in order to significantly reduce the search space. However, the sequential components of the SD algorithm are a serious limitation in a parallel environment.

In Thesis group I. with the PSD algorithm, I proposed a highly parallel algorithm that eliminated the sequential components and bottlenecks of the SD algorithm and the efficient mapping to massively parallel architectures could be realized. In Thesis group II., I further improved the performance of the PSD algorithm by defining a detection ordering based on the inverse channel matrix row norms. These results made possible to significantly improve the computation time of the optimal BER curves in larger MIMO systems under different circumstances that was very time-consuming until now.

It was shown that the SD algorithm is analogous to the closest lattice point (CLP) problem, or equivalently, the shortest vector problem (SVP) [25], [31], [32]. Since optimal LR techniques, such as the Minkowski and Hermite-Korkine-Zolotareff LR algorthms, iterativetly perform CLP searches and cryptography problems can be traced back to CLP and SVP problems, my results presented in Thesis groups I. and II. can be applied to enhance the solution of these problems.

5 Acknowledgements

First of all, I am most grateful to my supervisors, *Profs. Géza Kolumbán* and *Péter Szolgay* for their motivation, guidance, patience and support.

I owe my deepest gratitude to *Prof. Tamás Roska* for the inspiring and thought-provoking discussions and for opening up new horizons. I deeply thank *Prof. Árpád Csurgay* and *Dr. András Oláh* for the motivation and encouragement, giving me the strength to go on.

I am immensely grateful to *Prof. Antonio M. Vidal* for inviting me to his research group at the Institute of Telecommunications and Multimedia Applications (iTEAM) of the Polytechnic University of Valencia and sharing with me his wide experience in computational mathematics every time I needed it. I am very grateful to *Prof. Alberto González*, *Dr. Gema Piñero* and *Dr. Francisco J. Martínez-Zaldívar* for their generous advice and help in the field of wireless communications.

I would like to thank all my friends and colleagues with whom I spent these past few years *Dóra Bihary, Bence Borbély, Zsolt Gelencsér, Antal Hiba, Balázs Jákli, András Laki, Endre László, Norbert Sárkány, István Reguly, János Rudán, Zoltán Tuza, Tamás Fülöp, András Horváth, Miklós Koller, Mihály Radványi, Ádám Rák, Attila Stubendek, Gábor Tornai, Tamás Zsedrovits, Ádám Balogh, Ádám Fekete, László Füredi, Kálmán Tornai, Ákos Tar, Dávid Tisza, Gergely Treplán, József Veres, András Bojárszky, Balázs Karlócai, Tamás Krébesz.* Special thanks go to *István Reguly* for more than a decade of friendship and of continuous collaboration.

Thanks also to my colleagues at the iTEAM: *Emanuel Aguilera, Jose A. Belloch, Fernando Domene, Laura Fuster Pablo Gutiérrez, Jorge Lorente, Luis Maciá, Amparo Martí, Carla Ramiro* for all the good moments spent together while working at the lab.

I thank the Pázmány Péter Catholic University, Faculty of Information Technology and Bionics, for accepting me as a doctoral student and supporting me throughout grants TÁMOP-4.2.1/B-11/2/KMR-2011-0002 and TÁMOP-4.2.2/B-10/1-2010-0014.

Finally, I will be forever indebted to my parents *Enikő* and *Máté*, brother *István*, and grandparents *Böbe, Éva, Márton* for enduring my presence and my absence, and for being supportive all the while, helping me in every way imaginable.

Last, but not least, I would like to express my sincere gratitude to my wife *Ildikó*, who offered me unconditional love and support throughout the course of this thesis.

References

Author's journal publications

[1] **Csaba M. Józsa**, Géza Kolumbán, Antonio M. Vidal, Francisco J. Martínez-Zaldívar, and Alberto González. "Parallel Sphere Detector algorithm providing optimal MIMO detection on massively parallel architectures". In: *Concurrency and Computation: Practice and Experience* (2015). DOI: 10.1002/cpe.3488.

[2] **Csaba M. Józsa**, Fernando Domene, Antonio M. Vidal, Gema Piñero, and Alberto González. "High performance lattice reduction on heterogeneous computing platform". In: *The Journal of Supercomputing* (2014), pp. 1–14. ISSN: 0920-8542. DOI: 10.1007/s11227-014-1201-2.

Author's conference publications

[3] **Csaba M. Józsa**, Géza Kolumbán, Antonio M. Vidal, Francisco-José Martínez-Zaldívar, and Alberto González. "New Parallel Sphere Detector Algorithm Providing High-Throughput for Optimal MIMO Detection". In: *2013 International Conference on Computational Science (ICCS 2013)*. Vol. 18. Barcelona, Spain, 2013, pp. 2432 –2435. DOI: http://dx.doi.org/10.1016/j.procs.2013.05.417.

[4] **Csaba M. Józsa**, Fernando Domene, Gema Piñero, Alberto González, and Antonio M. Vidal. "Efficient GPU implementation of Lattice-Reduction-Aided Multiuser Precoding". In: *Wireless Communication Systems (ISWCS 2013), Proceedings of the Tenth International Symposium on.* Ilmenau, Germany, Aug. 2013, pp. 1–5. ISBN: 978-3-8007-3529-7.

[5] Fernando Domene, **Csaba M. Józsa**, Antonio M. Vidal, Gema Piñero, and Alberto González. "Performance analysis of a parallel Lattice Reduction algorithm on many-core architectures". In: *The 13th International Conference on Computational and Mathematical Methods in Science and Engineering (CMMSE 2013).* Vol. 2. Almeria, Spain, June 2013, pp. 535–542. ISBN: 978-84-616-2723-3.

[6] Tamás Krébesz, **Csaba M. Józsa**, and Géza Kolumbán. “New carrier generation techniques and their influence on bit energy in UWB radio”. In: *Circuit Theory and Design (ECCTD), 2011 20th European Conference on.* IEEE. Aug. 2011, pp. 801–804. DOI: 10.1109/ECCTD.2011.6043838.

[7] Tamás Krébesz, Géza Kolumbán, and **Csaba M. Józsa**. “Ultra-wideband impulse radio based on pulse compression technique”. In: *Circuit Theory and Design (ECCTD), 2011 20th European Conference on.* IEEE. Aug. 2011, pp. 797–800. DOI: 10.1109/ECCTD.2011.6043839.

Related publications

[8] Emre Telatar. “Capacity of Multi-antenna Gaussian Channels”. In: *European Transactions on Telecommunications* 10.6 (1999), pp. 585–595. ISSN: 1541-8251.

[9] Ezio Biglieri, Robert Calderbank, Anthony Constantinides, Andrea Goldsmith, Arogyaswami Paulraj, and H. Vincent Poor. *MIMO Wireless Communications.* New York, NY, USA: Cambridge University Press, 2007. ISBN: 0521873282.

[10] Michael Wu, Yang Sun, Siddharth Gupta, and Joseph R. Cavallaro. “Implementation of a High Throughput Soft MIMO Detector on GPU”. In: *J. Signal Process. Syst.* 64.1 (July 2011), pp. 123–136. ISSN: 1939-8018.

[11] Wang Hongyuan and Chen Muyi. “A Fixed-Complexity Sphere Decoder for MIMO Systems on Graphics Processing Units”. In: *Information Engineering and Computer Science (ICIECS), 2010 2nd International Conference on.* Dec. 2010.

[12] Rongchun Li, Yong Dou, Dan Zou, Shi Wang, and Ying Zhang. “Efficient graphics processing unit based layered decoders for quasi-cyclic low-density parity-check codes”. In: *Concurrency and Computation: Practice and Experience* 27.1 (2013), pp. 29–46. ISSN: 1532-0634.

[13] Rongchun Li, Yong Dou, and Dan Zou. “Efficient parallel implementation of three-point viterbi decoding algorithm on CPU, GPU, and FPGA”. In: *Concurrency and Computation: Practice and Experience* 26.3 (2014), pp. 821–840. ISSN: 1532-0634.

[14] Fernando Domene, Sandra Roger, Carla Ramiro, Gema Pinero, and Alberto Gonzalez. “A reconfigurable GPU implementation for Tomlinson-Harashima precoding”. In: *Acoustics, Speech and Signal Processing (ICASSP), 2012 IEEE International Conference on*. 2012.

[15] M. Pohst. “On the computation of lattice vectors of minimal length, successive minima and reduced bases with applications”. In: *ACM SIGSAM Bulletin* 15.1 (1981), pp. 37–44.

[16] E. Viterbo and E. Biglieri. “A universal decoding algorithm for lattice codes”. In: *14 Colloque sur le traitement du signal et des images, FRA, 1993*. GRETSI, Groupe d’Etudes du Traitement du Signal et des Images. 1993.

[17] P.W. Wolniansky, G.J. Foschini, G.D. Golden, and R. Valenzuela. “V-BLAST: an architecture for realizing very high data rates over the rich-scattering wireless channel”. In: *Signals, Systems, and Electronics, 1998. ISSSE 98. 1998 URSI International Symposium on*. IEEE. Sept. 1998, pp. 295–300.

[18] D. Wubben, D. Seethaler, J. Jalden, and G. Matz. “Lattice Reduction”. In: *Signal Processing Magazine, IEEE* 28.3 (May 2011), pp. 70–91. ISSN: 1053-5888.

[19] Arjen Klaas Lenstra, Hendrik Willem Lenstra, and László Lovász. “Factoring polynomials with rational coefficients”. In: *Mathematische Annalen* 261.4 (1982), pp. 515–534.

[20] Huan Yao and Gregory W. Wornell. “Lattice-reduction-aided detectors for MIMO communication systems”. In: *Global Telecommunications Conference, 2002. GLOBECOM ’02. IEEE*. Vol. 1. Nov. 2002, pp. 424–428.

[21] Christoph Windpassinger, Robert FH Fischer, Tomáš Vencel, and Johannes B Huber. “Precoding in multiantenna and multiuser communications”. In: *IEEE Trans. Wireless Commun.* 3.4 (2004), pp. 1305–1316.

[22] Wai Ho Mow. “Maximum likelihood sequence estimation from the lattice viewpoint”. In: *Information Theory, IEEE Transactions on* 40.5 (Sept. 1994), pp. 1591–1600. ISSN: 0018-9448.

[23] Xiaoli Ma, Wei Zhang, and A. Swami. “Lattice-reduction aided equalization for OFDM systems”. In: *Wireless Communications, IEEE Transactions on* 8.4 (Apr. 2009), pp. 1608–1613. ISSN: 1536-1276.

[24] R. Zamir, S. Shamai, and U. Erez. “Nested linear/lattice codes for structured multiterminal binning”. In: *Information Theory, IEEE Transactions on* 48.6 (2002), pp. 1250–1276. ISSN: 0018-9448.

[25] E. Agrell, T. Eriksson, A. Vardy, and K. Zeger. “Closest point search in lattices”. In: *Information Theory, IEEE Transactions on* 48.8 (2002).

[26] A. Hassibi and S. Boyd. “Integer parameter estimation in linear models with applications to GPS”. In: *Signal Processing, IEEE Transactions on* 46.11 (Nov. 1998), pp. 2938–2952. ISSN: 1053-587X.

[27] R.N. Neelamani, R.G. Baraniuk, and Ricardo de Queiroz. “Compression color space estimation of JPEG images using lattice basis reduction”. In: *Image Processing, 2001. Proceedings. 2001 International Conference on.* Vol. 1. 2001, pp. 890–893.

[28] Ravi Kannan. “Improved algorithms for integer programming and related lattice problems”. In: *Proceedings of the fifteenth annual ACM symposium on Theory of computing.* ACM. 1983, pp. 193–206.

[29] C. P. Schnorr and M. Euchner. “Lattice basis reduction: Improved practical algorithms and solving subset sum problems”. In: *Mathematical Programming* 66 (1 1994), pp. 181–199.

[30] PhongQ. Nguyen and Jacques Stern. “Lattice Reduction in Cryptology: An Update”. English. In: *Algorithmic Number Theory.* Ed. by Wieb Bosma. Vol. 1838. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2000, pp. 85–112. ISBN: 978-3-540-67695-9.

[31] M.O. Damen, H. El Gamal, and G. Caire. “On maximum-likelihood detection and the search for the closest lattice point”. In: *Information Theory, IEEE Transactions on* 49.10 (2003), pp. 2389–2402.

[32] B. Hassibi and H. Vikalo. “On the sphere-decoding algorithm I. Expected complexity”. In: *Signal Processing, IEEE Transactions on* 53.8 (2005).