CRACKING THE COMPLEXITY OF FIXED-POINT REFINEMENT IN COMPLEX WIRELESS SYSTEMS

D. Novo[*], *I. Tzimi*[†], *U. Ahmad*[‡], *P. Ienne*[*], *and F. Catthoor*[‡]

[*]EPFL, Lausanne, Switzerland. Email: {david.novobruna, paolo.ienne}@epfl.ch

[†]University of Patras, Patras, Greece. Email: tzimi@ceid.upatras.gr

[‡]IMEC, Leuven, Belgium. Email: {ubaid, catthoor}@imec.be

ABSTRACT

Fixed-point arithmetic leads to efficient implementations. However, the optimization process required to size each of the implementation signals can be prohibitively complex. In this paper, we introduce a new *divide-and-conquer* method that is able to approach the quality of global methods in significantly less time. Firstly, our method sorts the signals in multiple groups considering the propagation path of the signals to the global application metric (i.e., bit-error rate). Then, the fixed-point configurations of the groups are resolved with fast local simulations. Finally, the global fixed-point configuration is composed by the group configurations using slow global simulations. The method is applied to the fixed-point refinement of an advanced wireless algorithm achieving close to 9 times speedup with respect to a reference statistical method without affecting the quality of the result.

Index Terms— Fixed-point, wireless DSP implementation

1. INTRODUCTION

Porting floating-point algorithm implementations to fixed-point arithmetic is an effective way to reduce delay, energy and area. Because fixed-point numbers do not require mantissa alignment, the circuitry is significantly simpler and faster. However, to be able to take advantage of the simplicity of fixed-point circuitry, the complexity burden shifts to the algorithm design process. In a fixed-point implementation, every signal needs to be carefully dimensioned to minimize implementation complexity while maintaining the precision required by the application. This process is referred as *fixed-point refinement*.

The fixed-point refinement can take up to 30% of the total implementation time [1] and includes two main steps. On the one hand, the range analysis is responsible of bounding the *Most-Significant Bit (MSB)*. The typical objective of range analysis is to avoid overflows, which result in high magnitude errors. On the other hand, the precision analysis is responsible of bounding the *Least-Significant Bit (LSB)*. The precision analysis must distribute a small error budget amongst the different signals in the implementation and it is often formulated as an iterative optimization problem. This iterative nature makes the precision analysis step very time consuming, being responsible for most of the fixed-point refinement time [2].

The existing precision analysis methods are either analytical or statistical. The analytical methods propagate the quantization noise to the outputs of the system, which enables an analytical computation of a Signal-to-Quantization-Noise Ratio (SQNR) metric [3, 4]. However, complex systems often include operators, such as decision-making operators, that cannot be modeled analytically and cut the propagation of the quantization noise to the output [5]. Furthermore, complex systems often include multiple outputs, which will result in multiple SQNR measuring points. The distribution of a global metric, such as *Bit-Error Rate (BER)*, amongst multiple SQNR points is hard (if not impossible) to model analytically. For these reasons, the precision analysis of large complex systems is commonly addressed with statistical methods [6, 7]. These methods rely on Monte Carlo simulations to directly estimate the application metric, e.g., BER, of a particular fixed-point configuration. However, as discussed in Section 3, these Monte Carlo simulations are long, and they grow as system complexity keeps on increasing. Accordingly, practical statistical precision analysis of complex systems can only afford a very limited number of fixed-point system simulations, resulting in very simplistic optimization methods that render suboptimal fixed-point configurations.

This paper includes two main contributions: (1) a remapping of the precision constraints governing the statistical precision analysis of wireless communication systems that reduces the number of Monte Carlo simulations; and (2) a novel hierarchical method to address the precision analysis of complex systems. Our method combines the SQNR metric of analytical methods with global statistical simulations to achieve high quality solutions at a very moderate simulation time. In particular, the method consists of three main steps: division of the system in multiple groups of signals, where each group of signals is responsible for one SQNR point; generation of a set of Pareto solutions for each group using fast quantization

978-1-4673-6238-2/13 $31.00

noise estimations; and combination of the partial group solutions to compose the global fixed-point configuration using global statistical simulations. As an illustration, we apply our method to the fixed-point refinement of an advanced wireless algorithm achieving close to 9 times speedup with respect to a reference statistical method without affecting the quality of the result.

2. FINITE PRECISION WIRELESS SYSTEMS

This section presents the formalization of the statistical precision analysis in wireless communication systems. These systems are typically characterized by their BER curve. As shown in Fig. 1, the BER decreases monotonically with the *Signal-to-Noise Ratio (SNR)*; the cleaner the channel, the more reliable the wireless link.

After fixed-point refinement, the wireless algorithms lose some precision due to the mapping of their signal values onto a finite number of levels. As a result, the BER curve of a fixed-point wireless system will always need a higher SNR to reach the same BER of its infinite precision version. Such an increase in SNR is typically known as Implementation Loss (IL). Accordingly, the fixed-point refinement of a wireless system is often defined as:

$$\arg\min C(\lambda) \text{ s.t. } \mathrm{IL} \leq \psi,\ \mathrm{BER} = \beta. \qquad (1)$$

where C corresponds to the cost metric objective of the optimization (e.g., energy, area, etc.), λ is the vector of signal bitwidths, β is the maximum BER tolerated by the transmission mode, and ψ is the desired implementation loss.

In practice, however, the BER can only be estimated after running a Monte Carlo simulation that takes an SNR point as input. Accordingly, the computation of IL requires an unbounded number of simulations, where the input SNR is swept until $\mathrm{BER} = \beta$ is satisfied. Instead, we propose to remap the original constraints to an equivalent set as follows:

$$\mathrm{IL} \leq \psi,\ \mathrm{BER} = \beta \ \Rightarrow\ \mathrm{BER} \leq \beta,\ \mathrm{SNR} = \eta + \mathrm{IL}, \qquad (2)$$

where η corresponds to the SNR at which the infinite precision version provides $\mathrm{BER} = \beta$. Thereby, the optimization constraints can be checked with a single Monte Carlo simulation that only needs to evaluates whether $\mathrm{BER} \leq \beta$.

Additionally, the optimization problem of Eq. 1 has been proven to be non-convex [2] and a heuristic approach is required. In this paper we use as reference the *Max-1 bit* heuristic [8], described in Fig. 2. This greedy heuristic gradually reduces an initial bitwidth configuration λ by decreasing at each iteration the signal that causes minimum BER degradation until the degradation constraint is violated.

Despite using such a simple heuristic, the fixed-point refinement time can become prohibitive in large systems. Accordingly, in the next section we propose a new method to significantly reduce the required optimization time.

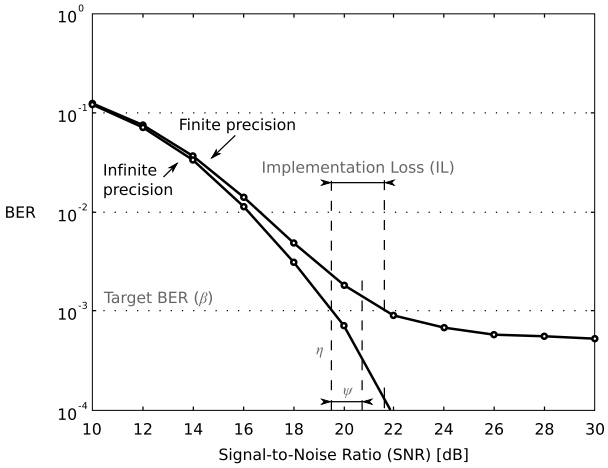


Fig. 1. Effects of finite precision computation in wireless applications. The SNR required to reach a given BER increases after fixed-point refinement.

```
1  input:  λ[1..num_sig], β;
2  output: λ[1..num_sig];
3  BER = BER_sim(λ); // run Monte Carlo simulation
4  while BER ≤ β // degradation constraint
5    for sig = 1 to num_sig // try all signals
6      λ(sig)          = λ(sig) − 1;
7      this_BER(sig) = BER_sim(λ);
8      λ(sig)          = λ(sig) + 1;
9    end for
10   // find signal causing minimum BER
11   this_sig     = find(this_BER == min(this_BER));
12   λ(this_sig) = λ(this_sig) − 1;
13   BER          = thisBER(this_sig);
14 end while
15 λ(this_sig) = λ(this_sig) + 1;  // recover bitwidths
```

Fig. 2. Pseudocode of the *Max-1 bit* heuristic [8].

3. HIERARCHICAL FIXED-POINT REFINEMENT

In this section we describe a new *divide-and-conquer* heuristic approach to reduce fixed-point refinement time.

Firstly, the global BER metric that characterizes the whole wireless chain needs to be reexpressed onto a new metric local to the target block of functionality. Inspired by previous work on analytical precision analysis methods [9], we select the expectation of the output quantization noise power as our local metric.

$$E(e_o^2) = \frac{1}{N}\sum_{i=1}^{N}(o_i - o_i^q)^2, \qquad (3)$$

where o_i is the ith output sample of the target functional block, o_i^q is the same sample after fixed-point refinement, e_o is the quantization error on output o, and N is ideally infinite. In practice, however, N is approximated to the smallest number that guarantees a certain convergence.

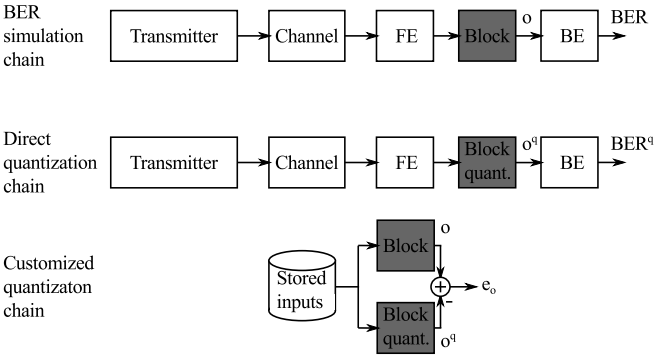
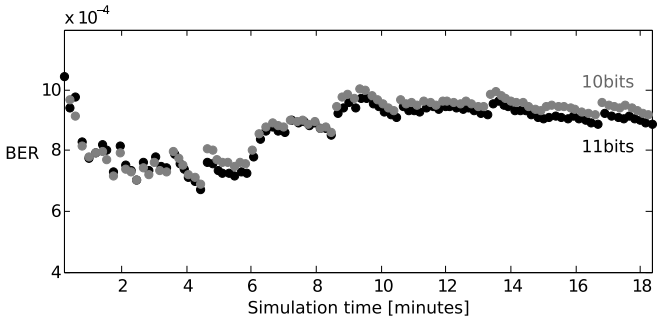


Fig. 3. Simulation chains. The highlighted block is the target of the fixed-point refinement.

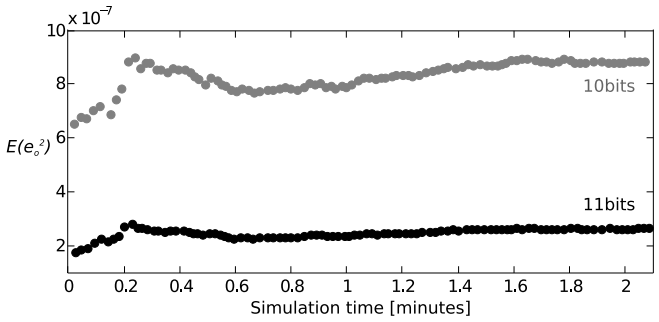
Actually, it is much faster to faithfully estimate $E(e_o^2)$ statistically than BER. Fig. 4(a) shows the evolution of BER in the simulation of 100 channels for two fixed-point configurations of the wireless algorithm introduced in Section 4. One configuration sets all decimal bits to 11bit while the second one sets them to 10bit. Although the 11bit configuration provides higher accuracy on the computations and a lower BER, the Monte Carlo simulation takes more than 8 minutes to make it evident. This is because the quantization noise of both configurations is so small that produces only a very small degradation in BER, which requires long simulation runs to be noticed. In fact, the BER degradation of the final fixed-point implementation has also to be very small, and so are all the decisions required in iterative fixed-point refinement heuristic described in Fig. 2.

Alternatively, Fig. 4(b) sows the noise simulations for the same fixed-point configurations and for the same 100 channels. Firstly, the noise simulation is much faster as it uses the *customized quantization chain* described in Fig. 3, which allows the evaluation of all the channels in about 2 minutes. Instead, the BER simulation uses the *direct quantization chain* requiring more than 18 minutes. Besides, the superior precision of the 11bit configuration is evident from the first sample. Accordingly, the noise simulation needs to simulate considerably less samples than the BER simulation to decide which configuration is more precise.

Importantly, ranking fixed-point configurations based on noise simulations is equivalent to ranking them based on BER simulations. This can be inferred from Fig. 1, where the BER monodically decreases with the increase in noise level. This noise can either come from the wireless channel or from the fixed-point approximation. The problem, however, is that functional blocks typically have multiple noise outputs and the noise budgeting between them is not trivial. Different noise outputs may be correlated in strange ways that are only evident through BER simulations. Thus, in the next section we propose a multistep method that can make use of noise simulations while still considering output correlations.



(a) Direct quantization chain.



(b) Customized quantizaton chain.

Fig. 4. Using $E(e_o^2)$ to identify the most accurate fixed-point configurations is faster than using BER.

3.1. Signal Grouping and Initial Noise Budgeting

Firstly, the number of noise outputs in the targeted block must be identified. Clearly, all the output signals are promoted to noise outputs. Less straightforward is the fact that all the inputs to decision-making operators inside the block also need to be promoted to noise outputs. The latter can be illustrated with the following example:

```
out = (in1 - in2 > 0) ? in2*in2 : in1*in1;
```

where the input to the condition `in1 - in2` influences the actual output and thus, its quantization noise needs to be kept under control. The lefthand side of Fig. 5 illustrates the corresponding *Data-Flow Graph (DFG)*.

Once the noise outputs are identified, a group of signals is created for each of the noise outputs. The DFGs of the two groups included in the previous example are illustrated in the middle and the righthand side of Fig. 5. A group contains all the signals affecting a particular noise output. If a signal affects multiple outputs (e.g., Q1 and Q2 in Fig. 5), the signal is assigned to the the group having the most sensitive output. Thereby, the signal will be refined according to the most constraining output.

Then, for each of the signal groups a normally distributed noise source, $N(0, \sigma^2)$, is injected at the noise output. Fig. 6 illustrates the noise injection process for the two groups of the example. Successive BER simulations sweep the variance of the noise (i.e., σ^2) until $BER = \beta$ which implies

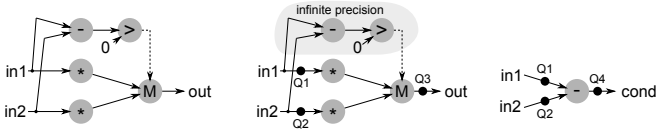


Fig. 5. Signal grouping. The original DFG (lefthand side) is divided in two groups (middle and righthand side).

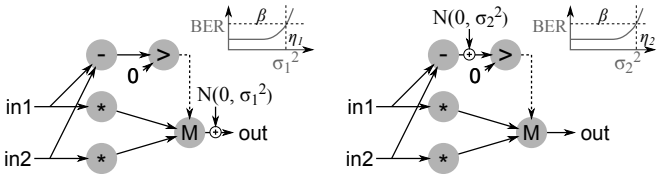


Fig. 6. Initial noise budgeting

that $\sigma^2 = \eta$. The latter corresponds to the maximum noise variance tolerated by the communication system at the given noise output. Since we only need an rough approximation of this noise variance, the required BER simulations can be designed to run very fast.

3.2. Group Level Optimization

Once the groups are established, the *Max-1 bit* heuristic, described in Fig. 2, is slightly adapted for the fixed-point refinement of the signals in a group. The metric to be monitored in each of the iterations is not the expensive BER but the output quantization noise power defined in Eq. 3. Accordingly, the constraint needs to be modified to $E(e_o^2) \leq \eta$. This process is illustrated in Fig. 7(a).

In reality, the assumption that the output quantization noise obeys a normal distribution of zero mean is just an approximation. Thus, we take the last fixed-point configuration of the group (the one that violated the constraint) and run a BER simulation. If the corresponding BER is still below β the iterative optimization is continued and for each bit reduction step the BER is checked until the β constraint is also violated.

In order to select only the set of *interesting* fixed-point configurations of the group, the latest configurations are successively selected for BER simulation until the BER is deviated only by a small percentage (i.e., $\Delta/100$) with respect to the infinite precision BER. The remaining configurations, whose influence to the BER is negligible, are discarded. This process is illustrated in Fig. 7(b).

3.3. Global composition

Once all the groups have been refined locally, the final fixed-point configuration is composed based on BER simulations. Thereby the correlations between groups can be considered and the BER degradation constraint can be guaranteed.

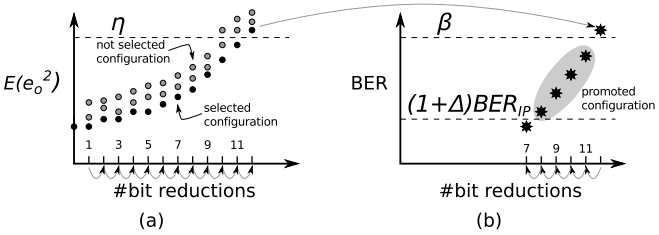


Fig. 7. Group level optimization. Noise simulations are used to find the initial cluster candidates (b), which are then pruned based on BER simulations.

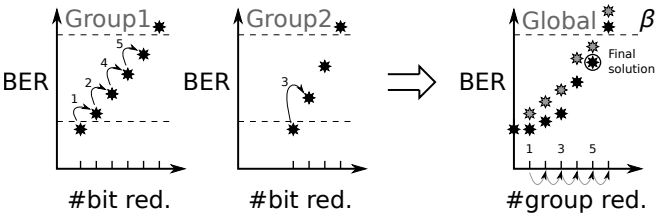


Fig. 8. Global composition. The final fixed-point configuration is composed from group configurations based on BER simulations.

Fig. 8 illustrates the global composition process that resembles the reference heuristic. Now, the heuristic choses at each iteration which group achieves a bit reduction with a smaller BER degradation. Importantly, the combinatorial problem has been reduced, as the number of clusters is typically much smaller than the number of signals.

4. RESULTS

This section presents and discusses the results of applying the proposed heuristic to an advanced wireless algorithm.

The *Hybrid Lattice Reduction (HLR)* algorithm, introduced by Ahmad et al. [10], is an effective way of improving the performance of modern wireless communication systems. Accordingly, this type of algorithm is expected to be implemented in future handhelds and be responsible for an important share of their complexity.Our HLR is implemented as a 4×4 MIMO design and integrated in a 3GPP *Long Term Evolution (LTE)* Matlab simulation chain. The channels are generated based on the standardized channel model.

HLR is a good example of the type of algorithms that are expected to run in future handhelds. HLR includes non-standard operations such as reciprocal or reciprocal square root and it exhibits a heavily input-dependent execution flow. Concretely, the HLR can execute 0 to 36 basic blocs depending on the input. This input dependent execution difficulties the fixed-point refinement process, as many different scenarios need to be considered, resulting in very long simulation times.

4.1. Fixed-Point Refinement Results

The HLR algorithm includes 11 different signals. The signals are divided into four groups of signals according to the criteria described in Section 3.1. Two of these groups (G1 and G2) are due outputs and the remaining two groups (G3 and G4) to decision-making operators. The number of signals per group is detailed in Table 1.

Figure 9 shows the simulation time of the proposed heuristic with respect to the reference method. The reference method includes an initial analysis to find the fixed-point configuration input to the algorithm described in Figure 2. This initial analysis assigns the same large number of decimal bits (20bits) to all the signals and decreases all the signals by one bit until the BER is slightly deteriorated (less than 10% over the floating point BER). The target implementation loss of the fixed-point refinement is 0.5 dBs at a BER of 10^{-3}, which is equivalent to a 20% degradation of the floating point BER at an SNR point of 20.3 dBs.

Our distributed heuristic is almost 9 times faster than the reference method for the single threaded execution. Note that the BER simulations dominate by more than 80% our overall optimization time. In the full parallel execution, the speedup is reduced to a factor 2.8 and the noise simulations start to dominate the overall optimization time. In such case, the replacement of our noise simulations by the analytical analysis methods discussed in Section 1 will enable a sizable acceleration of our optimization heuristic.

The signal wordlengths resulting from each of the refinements are shown in Figure 10. Interestingly, our method is able to find slightly smaller wordlengths than the global method. Accordingly, the heuristic presented in this paper is able to obtain slightly better quality results than the global heuristic in a significantly shorter amount of time while optimizing for the same fixed-point degradation constraints.

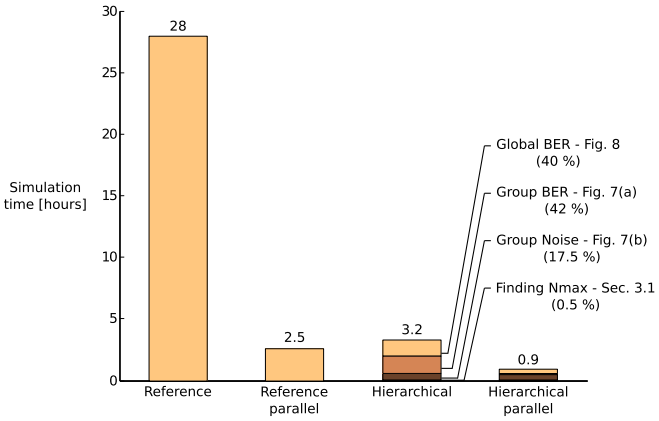


Fig. 9. Simulation time required by the reference and our hierarchical heuristic method to find an appropriate fixed-point configuration. Each method includes a single threaded version and a full parallel version. In both cases, our heuristic achieves significant speedups.

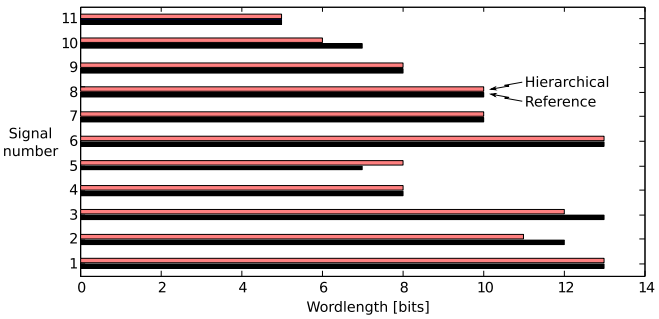


Fig. 10. Signal wordlengths resulting from the reference and our hierarchical heuristic. The fixed-point solution achieved with our heuristic is very similar to the reference solution.

4.2. Analysis and Discussion

The reference heuristic employed 56 bit reductions to find the optimized fixed-point configuration. Each bit reduction requires one BER simulation per signal leading to a total of 616 (56×11) BER simulations.

The distributed heuristic was able to achieve a significant speedup with respect to the global heuristic due to the reduction in the number of BER simulations, which was achieved in multiple phases. Firstly, our heuristic uses the fast noise simulations to find the Pareto configurations within a group. Table 1 shows the number of fixed-point configurations for each group after this analysis. A total of 70 fixed-point configurations for the four groups, which could result in a maximum of 280 (70 confs $\times$ 4 groups) BER simulations in the final group combination step.

Then, a second analysis further prunes the oversized fixed-point configurations that have a marginal influence on the BER. This analysis is analogous to the one used in the reference heuristic to find the initial uniformed wordlength fixed-point configuration. However, the upper bound in the reference heuristic is oversized for most of the signals (Table 1 indicates that a signal in G2 is setting the upper bound). By working at the group level, the total number of relevant fixed-point configurations was further reduced to 26 (the column *#BERconf* in Table1 shows the detailed breakdown), which can result in a maximum of 104 (26 confs $\times$ 4 groups) BER simulations in the final group combination. However, this pruning analysis is also based on BER simulations and requires one simulation per promoted fixed-point configuration.

Finally, the promoted fixed-point configurations of each group were combined in a final step. The latter required 6 group reduction steps to reach the final fixed-point configuration. Therefore, our heuristic employed a total of 50 BER simulations: 26 to prune the groups and 24 (6 reductions $\times$ 4 groups) to combine the groups. This shortened by more than 12 times the number of BER simulations of the refer-

Group	#signals	#Nconf	#BERconf
G1	6	24	8
G2	2	9	8
G3	1	14	6
G4	2	23	4

Table 1. Signals and fixed-point configurations per group.

ence heuristic, however, our heuristic includes additional enabling analyses that reduced to a about 9 times the effective simulation speedup.

5. RELATED WORK

Precision refinement has been studied extensively in previous work. Constantinides et al. [11] propose a *Mixed Integer Linear Programming (MILP)* technique for optimum wordlength allocation for the synthesis of *Linear Time-Invariant (LTI)* system. Unfortunately, this technique can only solve very small circuits and the authors acknowledge that a heuristic method is required to address more complex circuits.

Cantin et al. [8] review different optimization heuristics, including the reference heuristic used in this paper. All the analyzed heuristics are global and most of them can be made hierarchical by following the principles detailed in this work.

Parashar et al. [12] also proposed a hierarchical approach to fixed-point refinement. Similarly, they optimize a cost metric for a maximum noise budget at the output of each hierarchical blocks, however, they suggest to completely drop BER simulations. They do not discuss how a single application metric, such as BER, can be decomposed into multiple single source noise outputs (including blocks with multiple outputs). In fact, the different single source noise outputs often exhibit important correlations. Although their approach can work for some particular cases, it will lead to unsatisfactory results in complex systems including multiple noise outputs. In such cases, the approach is likely to either violate the BER degradation constraint or render a suboptimal fixed-point configuration since it cannot exploit the correlations between noise outputs. Instead, our solution guarantees by construction that the final implementation fulfills the specified fixed-point degradation at the expense of adding a minimum number of BER simulations.

6. CONCLUSIONS

The precision optimization heuristic proposed in this paper is able to combine the fast but local noise simulations with the global but slow BER simulations to deliver a fixed-point refinement flow that can achieve global quality at a very moderate simulation time. In particular, we address the fixed-point refinement of a MIMO wireless algorithm showing that our method achieves close to a 9 times speedup with respect to a reference statistical method without compromising the quality of the result. Besides, we propose a remapping of the precision constraints that govern the statistical precision analysis of wireless communication systems, which can be applied to any statistical method. Future work will apply the proposed heuristic to more benchmarks and extend the local noise simulations with analytical methods. Thereby, a further acceleration will be achieved, specially in full parallel executions.

7. ACKNOWLEDGMENTS

This work was partially supported by the European Union program FP7-PEOPLE-2011-IEF, under Project 299405.

8. REFERENCES

[1] T. Grötker, E. Multhaup, and O. Mauss, "Evaluation of HW/SW tradeoffs using behavioral synthesis," *Proceedings of ICSPAT*, 1996.

[2] G. Constantinides, P.Y.K. Cheung, and W. Luk, *Synthesis and optimization of DSP algorithms*, Springer, 2004.

[3] C. Shi and R.W. Brodersen, "Automated fixed-point datatype optimization tool for signal processing and communication systems," in *Proceedings of DAC*, 2004, pp. 478–483.

[4] D. Menard, R. Rocher, and O. Sentieys, "Analytical fixed-point accuracy evaluation in linear time-invariant systems," *IEEE TCAS-I*, vol. 55, no. 10, pp. 3197–3208, 2008.

[5] Karthick Parashar, Daniel Menard, Romuald Rocher, Olivier Sentieys, David Novo, and Francky Catthoor, "Fast performance evaluation of fixed-point systems with un-smooth operators," in *Proceedings of ICCAD*, 2010, pp. 9–16.

[6] W. Sung and K.I. Kum, "Simulation-based word-length optimization method for fixed-point digital signal processing systems," *IEEETSP*, vol. 43, no. 12, pp. 3087–3090, 1995.

[7] K.I. Kum and W. Sung, "Combined word-length optimization and high-level synthesis of digital signal processing systems," *IEEE Transactions on CAD*, vol. 20, no. 8, pp. 921–930, 2001.

[8] M.A. Cantin, Y. Savaria, and P. Lavoie, "A comparison of automatic word length optimization procedures," in *Proceedings of ISCAS*, 2002, vol. 2, pp. II–612.

[9] D. Menard and O. Sentieys, "Automatic evaluation of the accuracy of fixed-point algorithms," in *Proceedings of DATE*, 2002, p. 529.

[10] U. Ahmad, M. Li, S. Pollin, A. Amin, L. Van der Perre, and R. Lauwereins, "Hybrid lattice reduction algorithm and its implementation on an SDR baseband processor for LTE," in *Proceedings of EUSIPCO*, 2011, pp. 91–95.

[11] George A Constantinides, Peter YK Cheung, and Wayne Luk, "Optimum wordlength allocation," in *Proceedings of FCCM*, 2002, pp. 219–228.

[12] Karthick Parashar, Romuald Rocher, Daniel Menard, and Olivier Sentieys, "A hierarchical methodology for word-length optimization of signal processing systems," in *Proceedings of VLSID*, 2010, pp. 318–323.