Image Quantisation on a Massively Parallel Embedded Processor

Jan Jacobs[1], Leroy van Engelen[2], Jan Kuper[2], and Gerard J.M. Smit[2]

[1] Océ Technologies BV, P.O. Box 101,
5900MA Venlo, The Netherlands
jan.wm.jacobs@oce.com
[2] University of Twente, P.O. Box 217,
7500AE Enschede, The Netherlands

Abstract. Recent advances in embedded processing architectures allow for new powerful algorithms, which exploit the intrinsic parallelism present in image processing applications. This paper describes the results of the mapping process of stochastic image quantisation on a massively parallel processor. The problem can be modeled in a parallel way. Despite the fact that the implementation is IO bound, good speedups are achieved (16× compared to a standard image processing package running on a Pentium processor).

1 Introduction

Océ Technologies develops document systems for the office as well as for the design engineering market. Sample products are: printers, copiers, and scanners, which support professionals in their daily work. In order to maintain competitiveness Océ is interested in new algorithms and embedded architectures that raise quality and/or reduce development effort. In this paper, we focus on a parallel architecture and a relatively new algorithm in the context of *business graphics*. Business graphics are characterised by large areas filled with a single colour. This type of information, such as presentation sheets and charts (Fig. 1), is often scanned in an office environment. During scanning the image is sampled, which leads to distortion. One of the possible distortions is blurring, a kind of smearing, with the effect that more colours are introduced in a scan than necessary (Fig. 1, rightmost image). Reducing the number of colours in such scans is essential for image quality and can be useful as a first step in image compression. This process is called *colour quantisation*. Popular quantisation algorithms include median cut and octree algorithms [1]. These algorithms use a statistical approach: they count the occurrences of each colour and try to assign quantised colours using only this (frequency) information. The quality can be improved by including spatial (inter-pixel) relationships. In this paper we use one of the most recent image processing models, Markov Random Field (MRF) [2]. *Simulated Annealing*[3], which is an efficient stochastic procedure to solve combinatorial optimisation problems, is used to execute the MRF model in an iterative way till convergence is reached. Their combined advantage comes from the little a priori information on the world model and their suitability for parallel processing.

S. Vassiliadis et al. (Eds.): SAMOS 2007, LNCS 4599, pp. 139–148, 2007.

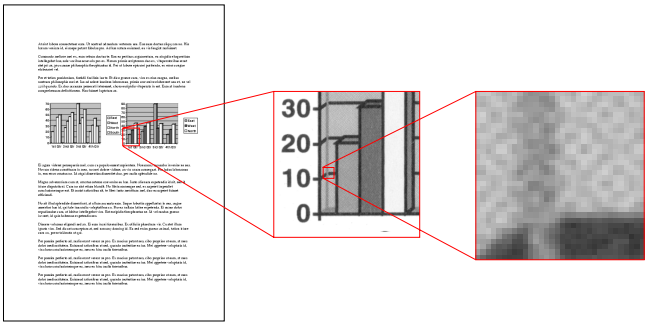


Fig. 1. Typical office scan containing text and charts. Scanning introduces image degradation: the number of unique grey-values increases from 10 to 229.

Present practice, however, makes such algorithms unusable since they are far too inefficient when run on sequential machines. It is our intention to find an embedded solution with a good performance-cost ratio, therefore we turn to massively parallel computing to implement these powerful algorithms. In our case we address the performance issues with the Aspex Linedancer processing array[4].

These considerations lead to the following research question:

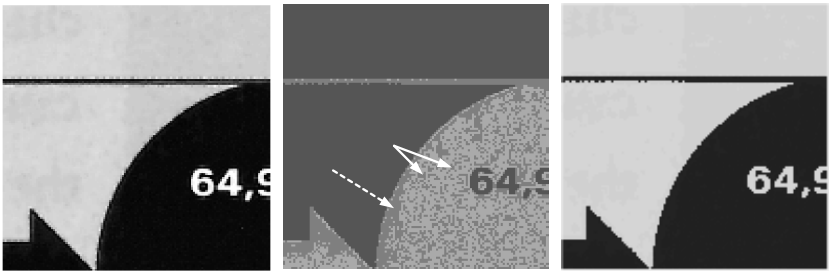
How to map image quantisation, based on Simulated Annealing and using an MRF image model, on a Linedancer massively parallel processing array?

This study will be presented here in the following manner. In Section 2 we introduce the theoretical concepts that lie at the base of our quantisation algorithm, we give a mathematical description of our image model and we briefly describe Simulated Annealing. The processor used in this research is also introduced. Next, Section 3 describes the parallel model and the related complexity estimates. Then the implementation is described in Section 4, followed by the results in Section 5. Finally, conclusions and recommendations are given in Section 6.

2 Background

2.1 Image Model for Quantisation

The basic problem is to recover a limited set of colours from a scanned business graphics original. The process, which reduces the number of colours by assigning them to a limited number of *classes* in an image, is called quantisation. For simplicity we restrict ourselves in this study to grey-value images since this does not alter the essence of both algorithm and mapping. See Fig. 2 for a result of a state of the art quantisation algorithm. To better observe quantisation artifacts, the quantised image is visualised in false colours, see Fig. 2(b). A false colour image is an image that depicts a subject in colours that magnifies the differences between values that are almost equal and, as a consequence, is good visible for human perception. Note for example the ringing around edges and the various speckles in Fig. 2(b), showing the substructure in the light and dark parts barely visible in Fig. 2(c). This false colouring can be steered by a grey-value



(a) Original image, 256 grey-values

(b) Resulting false colour image, 4 grey-values, dashed arrow indicates ringing, the others indicate speckles

(c) Resulting image, 4 grey-values, ringing just visible

Fig. 2. Example of state of the art quantisation algorithm

histogram, which can reveal such a situation. See for example the two adjacent peaks as depicted by the upper diagram in Fig. 3. *The problem we want to solve here is to raise the quality of the quantisation by a postprocessing step in an efficient way*.

First, we introduce some basic concepts, followed by two specific image models and conclude with a general image model based on the theory of MRF. The theory is described extensively in [2], the image model itself is taken from [5].

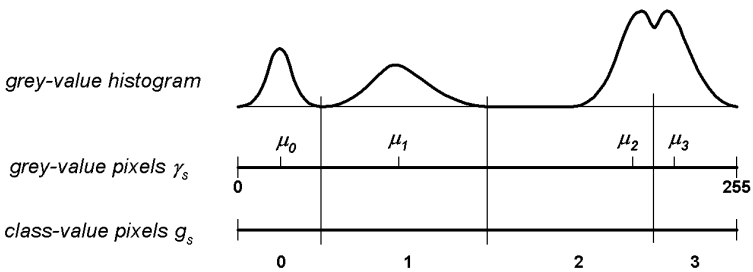


Fig. 3. Estimation of classes with associated class means

A pixel $s_{i,j}$ is denoted as a tuple (i,j), with $i \in H = \{0,\ldots,h-1\}$, $j \in W = \{0,\ldots,w-1\}$ in which w and h are the width and height of an image. We define $\mathcal{N}_{i,j} = \{(k,l) | \sqrt{(k-i)^2+(l-j)^2} \leq \delta, (k,l) \neq (i,j)\}$ as the *neighbourhood* $\mathcal{N}_{i,j}$ of pixel $s_{i,j}$. Thus, $\mathcal{N}_{i,j}$ contains all pixels within distance δ from $s_{i,j}$, except $s_{i,j}$ itself. See Fig. 4 for a neighbourhood with radius $\delta = 2$.

An image is defined on a grid of pixels $\mathcal{S} = \{s_{i,j} | i \in H, j \in W\}$. The scanning process produces grey-values that are assigned to pixels and denoted by $\gamma_s \in \{0,\ldots,255\}$. A desired property of quantisation is that it resembles the colour or grey-value of the

original. This so called *fidelity*, is optimised when the distance between grey-value and the mean value of the quantisation class (e.g. $\mu_{0\cdots3}$ in Fig. 3) is minimal for all pixels.

The purpose of quantisation is to determine the optimal quantisation class per pixel. Each class corresponds to an ordered sub-set of γ_s (e.g. $\{30\cdots40\}$) and this sub-set is represented by their class means μ_{g_s} (e.g. 36). These classes are identified by *labels* and denoted by $g_s \in \{0,\ldots,L-1\}$, where L represents the number of quantisation classes $1 \leq L \ll 256$. L is determined by inspecting the dominant peaks in the histogram, see for example Fig. 3 where $L = 4$.

A desired property of business graphics is the occurence of large planes with a single colour or label. This property, called *regularity*, is optimised when the dissimilarity between neighbouring labels is minimised.

The general MRF image model combines both the fidelity (grey-value) and the regularity (spatial relation) by simply minimising their weighted sum over all pixels. Finding **the** optimal label assignment is computationally very hard. However, reasonably

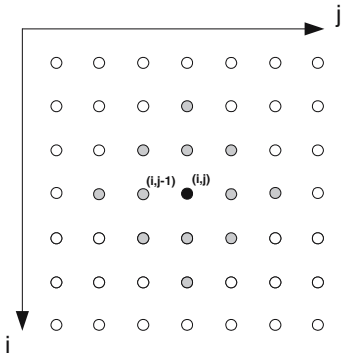


Fig. 4. Pixels in a grid with neighbourhood. The grey coloured pixels are all neighbours of the central pixel (i, j).

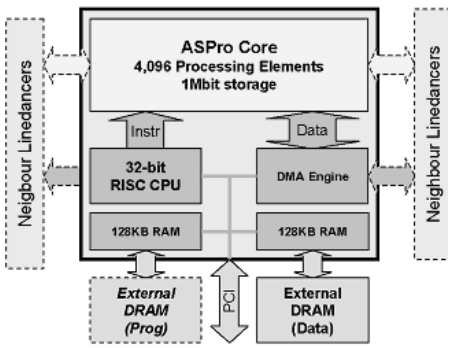


Fig. 5. Aspex Semiconductor's Linedancer

good solutions can be found by *Simulated Annealing*, an efficient procedure for solving combinatorial optimisation problems [3]. The algorithm repetitively executes the MRF model and searches a *state* (class-values or labels of all pixels in an image) where the weighted sum, or *energy*, is minimal. Each iteration the label of a single pixel is randomly chosen and its effect on the energy is computed. States which do decrease energy are always accepted (*deterministic acceptance*), but occasionally also slight increases are accepted in order to escape from local minima (*probabilistic acceptance*). In general the combination of MRF and Simulated Annealing is considered a powerful generic framework that can be used whenever an optimisation model can be constructed of a problem. See for example half-toning in [6], a more complex application than quantisation. For our purposes, however, the main advantage of this approach is that the algorithm can be run in parallel for all pixels, as will be shown in Section 3.

2.2 Associative Processing

Traditional computers rely on a memory that stores and retrieves data by its address rather than by its content. In such an organisation (von Neumann architecture), every accessed data word must travel individually between the processing unit and the memory. The simplicity of this retrieval-by-address approach has ensured its success, but has also produced some inherent disadvantages. One is the von Neumann bottleneck, where the memory-access path becomes the limiting factor for system performance. A related disadvantage is the inability to proportionally increase the bandwidth between the memory and the processor as the size of the memory scales up. Associative memory, in contrast, provides a naturally parallel and scalable form of data retrieval for both structured data (e.g. sets, arrays, tables, trees and graphs) and unstructured data (raw text and digitised signals). An associative memory can be extended to process the retrieved data in place, thus becoming an associative processor. This extension is merely the capability of writing a value in parallel into selected cells [4]. Applications range from handheld gaming, multi-media, wireless base stations, on-line transaction processing to heavy image processing, pattern recognition and data mining [4,7].

Aspex's Linedancer is an implementation of a parallel associative processor. The approach taken by Aspex Semiconductor is to use many simple associative processors in a SIMD arrangement (ASProCore). Each of the 4096 processing elements (PEs) on the Linedancer device has about 200 bits of memory (of which 64 bits are fully associative) and a single bit ALU, which can perform a 1 bit operation in 1 clock cycle. Operations on larger data types take multiple clock cycles. The aggregate processing power of the Linedancer depends entirely on parallel processing. For example: a 32-bit add will take many more clock cycles compared to a high-end scalar processor, but due to the parallelism 4096 additions can be performed in parallel. Multiple Linedancer devices can be connected together to create an even wider SIMD array.

The Linedancer device (shown in Fig. 5) includes an intelligent DMA controller, to ensure that data is moved in and out of the ASProCore concurrently with data processing, and a RISC processor (Sparc), to issue high level commands to the ASProCore and to set-up the DMA controller. All parts of the device run at the same clock frequency, which can be up to 400 MHz.

A Linedancer is programmed in an extended version of C, with additional syntax for controlling the ASProCore.

The reason Linedancer was chosen for this application is its scalable property towards the number of labels that can be processed by using its associative functionality (as opposed to other solutions, e.g. CNN [5]). Other reasons are scalable performance and its attractive performance-cost ratio.

3 Specification of the Algorithm

The flow of the system is depicted in Fig. 6. The module denoted by MRF and Simulated Annealing is the topic of this paper. To accelerate the Simulated Annealing procedure we follow the Modified Metropolis Dynamics (MMD) approach as described extensively in [5]. Contrary to MRF with its global energy, MMD strives for minimising a

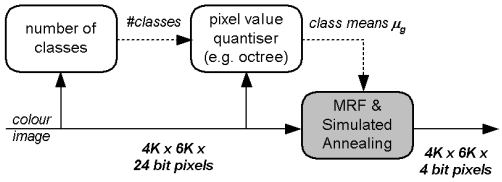


Fig. 6. Context of the quantisation module

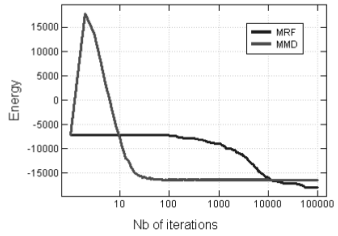


Fig. 7. The energy decrease of MRF and MMD

local energy $\mathcal{E}_s$ per pixel in parallel and therefore converges much faster when running on a parallel architecture. Although the MRF is in the long run somewhat better in quality (i.e. lower energy), MMD offers a better "quantisation result/compute time" ratio[5]. Fig. 7 illustrates the convergence power of MMD. Let γ_s be the observed grey-value image of pixel s, g_s be the quantisation class or label of pixel s, and g_r be the label of a pixel in the neighbourhood of s, then the energy of pixel s for the MMD approach is given by:

$$\mathcal{E}_s = \underbrace{(\gamma_s - \mu_{g_s})^2}_{\text{fidelity}} + \underbrace{\sum_{r \in \mathcal{N}_s} \beta\delta(g_s, g_r)}_{\text{regularity}}, \tag{1}$$

where

$$\delta(g_s, g_r) = \begin{cases} -1 & \text{if } g_s = g_r, \\ +1 & \text{if } g_s \neq g_r \end{cases} \tag{2}$$

Minimising the energy $\mathcal{E}$ will raise the quality of the quantisation. The fidelity term depends on the class means μ_{g_s}, which are constant, initialised by a previously executed module in the pipeline, see Fig. 6. The regularity term prefers neighbours having same labels (2), and β is a positive model parameter controlling the homogeneity of the regions of the image.

The Simulated Annealing procedure is coded in Algorithm 1. Here g represents the complete state of all pixels in an image and $\hat{g}$ represents a randomly chosen state of all pixels. An essential control variable in this algorithm is T or *Temperature*, named after related concepts in Physics [3]. A desired property of this procedure is the controlled and slow transition from a pseudo-stochastic to a deterministic phase. This transition corresponds to the transition from a broad search for global minima to the homing in on one –hopefully the best– minimum. Because T is high in the beginning, the system is able to jump to states that do (not too excessively) increase the energy (line 8), allowing escape from local minima. With T getting lower the system will behave more deterministic and fewer states that increase energy are accepted (lines 6 and 8). The threshold α controls the degree of probabilistic acceptance. The procedure can start off with an arbitrary state.

The values of parameters α, β and initial temperature T_0 are obtained from literature or based on preliminary computational experience. Typical values for these parameters are: threshold $\alpha \in [0.01, 1)$, regularity weighting $\beta \in [1, 100]$, temperature $T_0 \in [0, 16]$,

```
g ← initialisation state
for T ← T_0, T_0 · C, ..., T_0 · C^{n-1} do
    ĝ ← randomly chosen quantisation state g
    for all s ∈ S do
        ΔE_s ← E(ĝ_s) − E(g_s)
        if ΔE_s ≤ 0 then {Deterministic acceptance}
            g_s ← ĝ_s
        else if ΔE_s ≤ T · −ln α then {Probabilistic acceptance}
            g_s ← ĝ_s
        end if
    end for
end for
```

Algorithm 1. Modified Metropolis Dynamics

cooling factor $C \in [0.95, 1)$ and the number of iterations $n \in [50, 200]$. The typical dynamic behaviour of MMD versus MRF is illustrated by Fig. 7. In contrast to MRF, MMD settles around 100 iterations, independent of image size.

The complexity analysis of the sequential implementation of MRF is $O(n \cdot w \cdot h)$. Here w and h stand for the width and height of an image, respectively. The complexity analysis of the parallel implementation of MMD is $O\left(\frac{n \cdot w \cdot h}{\#PEs}\right)$, where $\#PEs$ stands for the number of Processing Elements.

4 Implementation Restrictions and Choices

To map this quantisation scheme on a Linedancer several implementation concerns have to be considered. Four of them: Look Up Table (LUT), tiling, bit-width of variables, and random number generation, are described below.

For most fine grain SIMD systems the size of the local memory is limited. In order to really be scalable in the number of labels, one must be able to retrieve the class means μ_{g_s} in an efficient way. The associative functionality of the Linedancer is very suitable in providing lookup functionality for all PEs.

Choosing a pixel-per-PE scheme means that a single Linedancer can host a 64×64 tile of pixels. To process larger images we use *tiling*, i.e. we divide the image in small chunks (of 64×64 pixels) that fit on the Linedancer. When running each tile, one after another for all n iterations, without providing for inter-tile communication, maximum speedup will be achieved but quality might be compromised. In order to counter this loss of quality a multi-pass scheme is used. In this way tiles are fetched multiple times, using overlapped fetch, effectively allowing for inter-tile communication.

The Linedancer does not support floating point arithmetic. For the various variables an accuracy analysis is made for determining the necessary bit-width in an integer arithmetic scheme. For the energy computation (1), the fidelity term takes the largest bit budget because of the square operation of a subtraction of two 8-bit values. The energy field is dimensioned to a 20-bit number representation, sufficient for storing the addition result of both the fidelity and regularity terms.

For every iteration a new state ($\hat{g}$) has to be generated in a random fashion. Pseudo random generators based on Linear Feedback Shift Register (LFSR) have low memory footprint and only need simple bit operations: XOR and Shift [8]. A 10 bit LFSR with only two tap points generates a pseudo random number sequence with cycle length $2^{10} - 1 = 1023$, which is sufficient.

5 Results and Discussion

Table 1 summarises the timing results of three distinct implementations for $L = 16$ quantisation classes. Two of them implement the MMD scheme, one executed on a 2 GHz Pentium Xeon with 1 GB RAM and one on the Linedancer. For comparison also a state of the art quantisation algorithm Octree[1] is used, which is part of the image processing package ImageMagick. The current Linedancer implementation is 16 times faster than Octree running on the above mentioned Pentium processor (and 128× faster compared to MMD on a Pentium).

Table 1. Execution times of quantisation for $L = 16$ classes: Octree and the MMD version both on a Pentium, and MMD on the Linedancer. All MMD processing is performed with $n = 100$ iterations.

# Pixels	Time (ms)		
	Octree on Pentium	MMD on Pentium	MMD on Linedancer
10000	108	517	5.95
40000	343	2070	23.1
160000	1171	8420	80.3
640000	4406	34600	280
2560000	16796	138000	1080

To give an idea how many cycles the different parts of the algorithm take, we measured the number of cycles taken for different stages of the algorithm. The results can be seen in Table 2. For the "For each neighbour"-parts, which take 3592 cycles each, 3200 (estimate based on a communication model) are spent on communication per part. This is approximately 73% of a total of 8754 cycles for each iteration per tile. However, clever reuse in communicating the neighbourhood could reduce this overhead, provided some memory space is available for storing intermediate results. Then 55% of a reduced total of 5207 cycles is spent in communication, yielding a speedup of 1.7.

A further improvement can be obtained by extending the Linedancer's synchronous inter-PE communication with a chordal ring, e.g. with an extra link for each PE with distance 64. This would yield a total speedup of 3.0 and would turn this realisation into a processing bound solution; only 20% of a reduced total of 2935 cycles is then spent in communication.

In quality terms the improvement w.r.t. state of the art quantisation algorithms is difficult to judge. The ringing at the edges and the speckles have disappeared when comparing the image in Fig. 2(b) and the image of Fig. 8(a). But the redistribution of classes lead to larger inhomogeneous areas and further study is needed to reduce this

Table 2. Number of measured cycles used for each iteration per tile of the algorithm. Nesting indicates loops, bold numbers indicate accumulated results.

Activity		# Cycles
Preparation		70
Processing one tile		
Calculate a new random labeling		44
For each label		**352**
$y - \mu_{g_s}$	$(16\times)$	22
Square		164
For each neighbour		**3592**
Add or subtract β (avg)	$(12\times)$	299.3
Load y		16
For each label		**352**
$y - \mu_{g_s}$	$(16\times)$	22
Square		164
For each neighbour		**3592**
Add or subtract β (avg)	$(12\times)$	299.3
Subtract energies, threshold values and update		70
Dump result		338
Total		**8754**

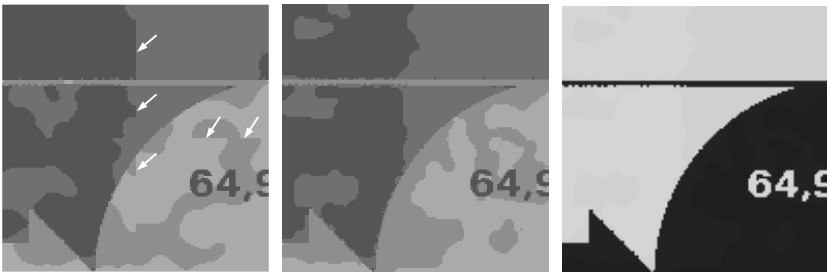
side-effect. The MMD implementation is single-pass, i.e. process each tile just once. However, single-pass results in annoying artifacts as can be seen in Fig. 8(a). To counter this, each tile can be processed multiple times, effectively allowing neighbouring tiles to communicate their regularity information, see image in Fig. 8(b). This can be done without performance degradation because on the Linedancer the dumping of the result of a previous tile and the loading of the next one can be completely hidden in the processing of the current tile.

6 Conclusions and Recommendations

An MMD implementation on the Linedancer has a high performance gain w.r.t. a state of the art sequential algorithm (speedup $16\times$), even in the case of a multi-pass approach. Careful engineering of the inter-PE communication could increase the speed by an extra factor of 1.7. When the processing array is extended with a chordal ring interconnection structure, with extra chords connecting PEs at distance 64, then a total speedup of approximately 3.0 can be obtained.

Though not the focus of this study some conclusions may be drawn on quality. First of all MMD promotes the redistribution of classes to larger uniform areas than the conventional method, as shown in the false coloured visualisations. The speckles and ringing effects at edges have disappeared. However, the redistribution of classes leads to larger inhomogeneous areas and further study is needed to reduce this side-effect.

From an architectural point of view we recommended an improvement in the inter-PE communication of the Linedancer.



(a) One pass, arrows indicate tiling artifacts (b) Two passes, no visible artifacts (c) Resulting image

Fig. 8. Quantisation by MMD on an image of 128 x 128 pixels, processed in chunks of 64 x 64 tiles

References

1. Freisleben, B., Schrader, A.: An evolutionary approach to color image quantization. In: Proceedings of 1997 IEEE International Conference on Evolutionary Computation (ICEC 97), Indianapolis, IN, USA pp. 459–464 (1997)
2. Kato, Z.: Modelisations markoviennes multiresolutions en vision par ordinateur. Application a la segmentation d'images SPOT. PhD thesis, University of Nice, [English translation] (1994)
3. Duda, R.O., Hart, P.E., Stork, D.G.: Pattern Classification, 2nd edn. John Wiley & Sons, Inc. Chichester (2001)
4. Aspex Semiconductor Ltd: Linedancer - overview (2005), `http://www.aspex-semi.com/pages/products/products_linedancer_overview.shtml`
5. Sziranyi, T., Zerubia, J., Czuni, L., Geldreich, D., Kato, Z.: Image segmentation using Markov random field model in fully parallel cellular network architectures. Real-Time Imaging 6, 195–221 (2000)
6. Geist, R., Reynolds, R., Suggs, D.: A markovian framework for digital halftoning. ACM Trans. Graph. 12(2), 136–159 (1993)
7. Krikelis, A., Weems, C.: Associative Processing and Processors, 1st edn. IEEE Computer Society Press, Los Alamitos (1997)
8. Golomb, S.W., Golomb, S.: Shift Register Sequences. Aegean Park Press, Laguna Hills, CA, USA (1981)