Rapid Acquisition of Gold Codes and Related Sequences Using Iterative Message Passing on Redundant Graphical Models

Fabio Principe*, Keith M. Chugg†, and Marco Luise*

*Dipartimento Ingegneria dell'Informazione
University of Pisa, Pisa 56122 - Italy
E-mail: {*fabio.principe, marco.luise*}*@iet.unipi.it*

†Communication Sciences Institute
University of Southern California, Los Angeles CA 90089-2565
E-mail: *chugg@usc.edu*

***Abstract*— A recent study, [1], demonstrated that iterative message passing algorithms (iMPAs) can be applied to rapidly acquire pseudo-noise (PN) sequences with low-complexity. Furthermore, a second work, [2], showed that significant benefits can be obtained using redundant graphical models, in case of *linear feedback shift register* (LFSR) sequences with sparse generator polynomials. Starting from these results, we address the problem of fast detection to Gold codes and LFSR sequences with dense generators. We will prove that these two aspects are closely related, and that, constructing redundant Tanner Graphs (TGs) from sparse higher-degree generator polynomials, it is possible to rapidly acquire these PN sequences at low signal-to-noise ratio (SNR), and with a low complexity. We also propose another distinct approach for acquiring Gold sequences using iterative methods based on a hierarchical model for the two LFSR generators that comprise a Gold code.**

I. Introduction

THE primary function of a direct-sequence spread-spectrum (DS/SS) receiver is to *despread* received signals. This is operation accomplished in two stages. The first one, referred to as *PN acquisition* ([3], [4], and [5]) produces a coarse alignment of the incoming PN signal and its local replica. Then, starting from this rough alignment, a fine synchronization is realized and maintained by a delay locked loop unit. This second stage is defined *PN tracking* ([3], [4], [5]). Therefore, the acquisition stage is the most critical point to guarantee a rapid and correct synchronization between receiver and transmitter.

The standard techniques used to detect PN sequences are: full parallel search, serial search (see [3]), and hybrid search ([3], [4]). The first method is the *maximum-likelihood algorithm* (MLA), [3], which is often infeasible due to high complexity, especially in case of long sequences. A simple serial search has a low-complexity, but its acquisition time is often prohibitively long. The hybrid search is a trade-off between these two methods.

In this context, a new technique to acquire PN codes has been, recently, presented in [1]. This method is based on running an iMPA on graphical models (a specific example being a *Tanner Graph*, TG [6]) with cycles ([7], [8]). Related research applying message passing (MP) algorithms to the acquisition problem was conducted independently in [9], and [10]. Instead of correlating the received signal with a local PN replica (as in all standard methods), this algorithm uses all the information, provided by the incoming signal, as messages to be run on a predetermined TG, thus approximating the ML method. This results a sub-optimal algorithm, that searches all possible PN phases in parallel with a complexity typically lower than the full parallel implementation, and an acquisition time shorter than the simple serial algorithm. Furthermore, focusing on LFSR sequences with sparse feedback polynomials, it was shown in [2] that significant improvements in terms of acquisition probability can be obtained using redundant graphical models (RGMs) made up of a set of redundant parity check equations.

In this paper we address the problem of fast acquisition of Gold sequences and LFSR sequences with non-sparse generators. Indeed, it is possible to demonstrate that these two problems are closely related, because Gold codes can be described by high-order LFSR (typically non-sparse) generators (see [4], [5], and [11]). This allows us to formulate a unified treatment on LFSR sequence acquisition using iMP detectors. The approach considered is to search for non-primitive higher-degree generator polynomials, that are sparse, and use these to construct RGMs for running iMPAs. Therefore, we present a simple method for searching for such sparse non-primitive generators and show the benefits of using these in redundant TGs through computer simulations. We also propose another distinct approach for acquiring Gold codes using iterative methods based on a hierarchal model for the two LFSR sequences that comprise a Gold code.

Furthermore, in order to evaluate the performance achievable using the proposed models, the GPS/SBAS[1] Gold sequences [12] have been simulated. The results obtained show good acquisition probability at low-SNR and low complexity, in agreement with the conclusions of [1], [2], but for a broader class of PN sequences.

The remainder of this paper is structured as follows: section II introduces the DS/SS signal model used to measure detector performance, and gives a brief overview on iMPA. Section III shows the equivalence between LFSR sequences and Gold codes. RGMs, generated by high-degree sparse polynomials,

[1]GPS stands for Global Positioning and SBAS stands for Satellite Based Augmentation Systems.

and their performance are discussed in Section IV. Section V describes hierarchical models to acquire Gold codes. Finally conclusions and suggestions for future work are reported in section VI.

II. Signal Model and Detection Algorithm

At the receiver, a basic and standard characterization of a DS/SS signal, during the acquisition stage, is

$$z_k = \sqrt{E_c} \cdot y_k + n_k = \sqrt{E_c} \cdot (-1)^{x_k} + n_k \qquad (1)$$

where z_k is a noisy sample received by detection unit, $y_k = (-1)^{x_k}$ is an antipodal modulation of $x_k \in \{0,1\}$ that is a PN sequence chip (sequence period is N), and n_k is an additive white gaussian noise (AWGN) mean value 0 and variance $N_0/2$. The model in (1) is diagrammed in Fig. 1. In other words, our assumptions are to acquire a *pilot* signal in case of *coherent* detection. This is a simplified representation of a DS/SS system, because it does not consider carrier frequency and phase offset, jammers, chip over-sampling, etc., but it is in agreement with the basic treatments reported in [1], [2], and [3].

The pattern, reported in Fig. 1, is made up of: a *PN Sequence Generator*, that outputs a predetermined PN sequence (LFSR sequences will be considered), a *channel*, that introduces a propagation delay ($\Delta \geqslant 0$) and AWGN, and a *detection unit* that acquires incoming signals, estimating their phase delay.

As described in [3], [4], and [5], all standard acquisition algorithms operate by correlating received signals with their shifted local replicas, until the right alignment is obtained. In this way, sequence delays can be directly evaluated from the local shifting.

A different approach is proposed in [1]. Consider a vector of M observations, $\mathbf{z}$, the MLA detector can be formulated as

$$\hat{\mathbf{y}} = \arg\max_{\mathbf{y}_i} \left[p(\mathbf{z}|\mathbf{y}_i) \right]$$

where $\mathbf{y}_i$ is a vector that contains M shifted chips of the transmitted PN sequence, and $p(\mathbf{z}|\mathbf{y}_i)$ is the likelihood function of $\mathbf{y}_i$ and $\mathbf{z}$. In other words, it estimates the shifted sequence (among N possible shifted versions) that maximizes the likelihood function. Therefore, an acquisition problem is, definitely, similar to a decoding problem, [4].

Exploiting the experience of iterative decoding of modern codes ([13], [14]), the ML estimation can be implemented by a MPA run on a graphical model without cycles (a *tree graph*). These optimal algorithms are often too complex to implement, so graphical models with cycles (e.g., TGs) are often used in practice. Indeed, these cyclic models yield sub-optimal solutions with lower complexity and it has been experimentally observed that, with proper model design, the performance can be close to that of the MLA. These graphical models are, basically, made up of sets of *variable nodes*, directly associated to incoming soft information, and *check nodes*, that identify all the parity equations (local constrains). Unfortunately, a systematic method for designing the best graphical model for a given problem does not exist.

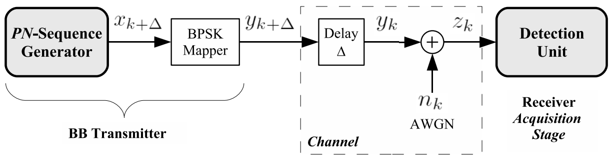


Fig. 1. DS/SS signal model.

Complete treatments on standard MPAs are reported in [6], [7], [8], and [15]. We simply add that, an iMPA, roughly, passes soft information between nodes in its TG, and each iteration ends when all nodes are activated. For this reason, in order to correctly implement an iMPA, one must define its *activation schedule*, which is the order that all variable and check nodes are activated, including when the algorithm is terminated. Typically, these algorithms end either when their estimated vectors verify all parity checks or when the max number of iteration, I_{max}, is obtained.

The last step is to define the processing used to perform the message updating. As reported in [7], [8], and [15], there are two main algorithms: *Sum-Product*, and *Min-Sum* algorithm. We only consider the Min-Sum algorithm (MSA) version because it is simple and does not require an estimate of E_c/N_0.

III. M-Sequences and Gold Codes

LFSR sequences are implemented by a simple r-stage shift register (SR) with a linear combination of its elements in a feedback path. A general *Fibonacci representation*, [3], of these sequences is depicted in Fig. 2. As the picture shows, at generic time k, assuming that x_k is the SR output and x_{k+i} (with $i \in [0, r]$) is the content of the i^{th} register, the following parity constraint is satisfied

$$\begin{aligned} 0 &= g_r \cdot x_k \oplus g_{r-1} \cdot x_{k+1} \oplus g_{r-2} \cdot x_{k+2} \oplus \dots \\ &\quad \dots \oplus g_2 \cdot x_{k+r-2} \oplus g_1 \cdot x_{k+r-1} \oplus g_0 \cdot x_{k+r} \\ &= \bigoplus_{i=0}^{r} g_{r-i} \cdot x_{k+i} \end{aligned} \qquad (2)$$

where $\oplus$ is modulo-2 addition and $g_i \in \{0,1\}$, $0 \leqslant i \leqslant r$ are feedback coefficients (also referred to as *taps*). A common way to represent a r-stage LFSR is by its *generating polynomial*, that contains its tap configuration, as

$$\begin{aligned} P(D) &= g_0 + g_1 \cdot D + \dots + g_{r-1} \cdot D^{r-1} + g_r \cdot D^r \\ &= \sum_{i=0}^{r} g_i \cdot D^i \end{aligned} \qquad (3)$$

where D is the delay unit, and r is the polynomial degree. Of course, for a given length r, g_0 and g_r are always 1.

M-Sequences are a subset of these sequences, because their generating polynomials are *primitive* polynomials (see [3]). This characteristic implies that their period is the maximum achievable with a r-stage LFSR generator (referred to as *maximal* LFSR sequences too). Hence, let N be the period of a r-stage m-sequence, its value is $N = 2^r - 1$. Another consequence of this property is that these sequences are univocally identified by their generating polynomials, because the initial word of their SRs (except for *zero* word) only

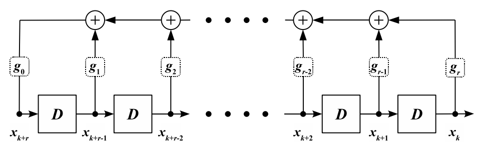


Fig. 2. Fibonacci representation of a r-stage LFSR generator.

produces an initial sequence shifting (also named initial *code phase*).

Because of these features, maximal LFSR sequences have been successfully employed in a wide range of SS systems, and they are also used to generate other spreading sequences – e.g., *Gold codes*.

Gold sequences are very common in *code division multiple access* (CDMA) systems, because they offer very low cross-correlation values, fundamental to reduce interference problems in multi-user applications ([3], [4], [5], and [11]).

These codes are generated by modulo-2 addition of a *preferred pair* of m-sequences with the same period, N (see also [11]). An example of Gold generator is reported in Fig. 3. It is a simplified representation of the GPS/SBAS Gold code generator, [12]. Of course, the addition is made chip by chip by synchronous clocking. In this way, the two code generators maintain the same phase relationship and the generated code has the same period, N, of the two initial m-sequences. Furthermore, for a fixed preferred pair of maximal LFSR sequences, a set of $N + 2$ different Gold codes (including the two generating m-sequences) can be obtained by varying the initial state of the two generating SRs ([11] and [4]). For example, in the case of GPS/SBAS codes (Fig. 3), where $N = 1023$, there are 1025 different Gold sequences.

There also exits an equivalent way for producing Gold sequences using a single higher-order LFSR generator. Indeed, as demonstrated in [11], a Gold code can be generated by a r-stage LFSR unit (the scheme is in Fig. 2) with the tap configuration

$$P(D) = P_{c'}(D) \cdot P_{c''}(D) \tag{4}$$

where $P_{c'}(D)$ and $P_{c''}(D)$ are the polynomials that specify the feedback connections of the two q-stage SRs, where $q = \frac{r}{2}$, that output the generating m-sequences $\mathbf{c}'$ and $\mathbf{c}''$ (as shows in Fig. 3). For example, in case of GPS/SBAS Gold sequences, the two m-sequence polynomials are

$$P_{c'}(D) = D^{10} + D^3 + 1 \tag{5a}$$

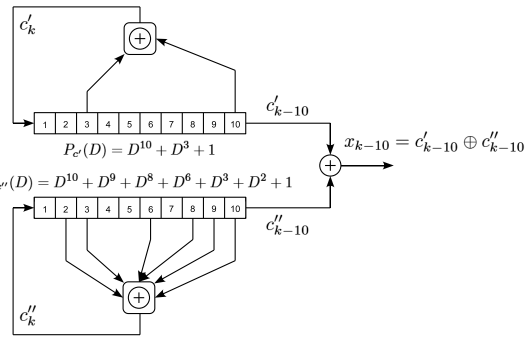
$$P_{c''}(D) = D^{10} + D^9 + D^8 + D^6 + D^3 + D^2 + 1 \tag{5b}$$

$q = 10$ implies $r = 2 \cdot q = 20$. From (4), the high-order LFSR (or Gold generating polynomial) is

$$P(D) = D^{20}+D^{19}+D^{18}+D^{16}+D^{11}+D^8+D^5+D^2+1 \tag{6}$$

this important result allows Gold codes to be treated as LFSR sequences.

The equivalent LFSR for a Gold sequence typically has a generator that is not sparse. Thus, the previously studied methods are not directly applicable. In the following section we describe how non-primitive, sparse, generators for these LFSR sequences can be found and used to provide effective iMPAs.



c'_k
1 2 3 4 5 6 7 8 9 10
c'_{k-10}
$P_{c'}(D) = D^{10} + D^3 + 1$
$x_{k-10} = c'_{k-10} \oplus c''_{k-10}$
$P_{c''}(D) = D^{10} + D^9 + D^8 + D^6 + D^3 + D^2 + 1$
1 2 3 4 5 6 7 8 9 10
c''_{k-10}
c''_k

Fig. 3. GPS/SBAS Gold generator.

IV. Graphical Models and Simulation Results

This section is divided in two parts. The first one describes the way to design graphical models with redundancy, while the second part shows the benefits achievable using redundant graphical models (RGM) generated by high-degree sparse polynomials.

A. Redundant Tanner Graphs

A graphical model (or TG) can be mathematically described by its *parity matrix*, that contains all the edges between its variable and check nodes. In other words, assuming that the j^{th} column is directly associated with the j^{th} variable node and that the i^{th} row correspond to the i^{th} check node, the matrix element $h_{i,j}$ is 1 only if the i^{th} check node and the j^{th} variable node are connected, otherwise $h_{i,j} = 0$.

Now, in agreement with [1], several TGs can be generated in function of the particular LFSR sequence to be acquired. Nevertheless, the simplest and most general way to build graphical models is based on generating polynomials of LFSR sequences. Indeed, considering (2) and (3), it is clear that each generating polynomial can identify a set of parity checks, that can be used to construct a simple TG. So, its parity matrix is

$$\mathbf{H} = \begin{pmatrix} g_r & \cdots & g_0 & 0 & \cdots & \cdots & 0 \\ 0 & g_r & \cdots & g_0 & 0 & \cdots & 0 \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & \vdots \\ 0 & \cdots & 0 & g_r & \cdots & g_0 & 0 \\ 0 & \cdots & \cdots & 0 & g_r & \cdots & g_0 \end{pmatrix}_{N_r \times N_c} \tag{7}$$

where $N_r = M - r$ is the number of rows (or number of parity equations), $N_c = M$ number of columns, r is the generating polynomial degree, M is the number of incoming observations (received soft information), and g_i (with $0 \leqslant i \leqslant r$) is the i^{th} polynomial coefficient. In the (7), each row is a shifted repetition of the polynomial vector ($\mathbf{p} = [g_r \cdots g_0]$, associated to $P(D)$) of one column. We refer to this TG as a *basic graphical model* (BGM).

The regular structure of the BGM may cause the associated iMPA may perform poorly [1], [8], [15]. *Redundant graphical models* (RGMs) have been introduced to alleviate this effect. They are, roughly, made up of a set BGMs that are put together to form one big TG. Each BGM is based on one equivalent generating polynomial of the same LFSR sequence to be detected. The use of such redundancy has been shown to improve performance when each BGM has poor cycle structure.

Consider a set of $n+1$ equivalent generating polynomials that enforce the same LFSR sequence structure. A RGM can be defined by simply grouping all the BGMs generated by these polynomials (as shown in (7)). The final parity matrix is

$$\mathbf{H_{RGM}} = \begin{pmatrix} \mathbf{H_0} \\ \mathbf{H_1} \\ \vdots \\ \mathbf{H_n} \end{pmatrix}_{N_r' \times N_c'} \tag{8}$$

where N_r' and N_c' are, respectively, the number of rows and columns. They are computed as

$$N_r' = [(n+1)\cdot M] - \sum_{j=0}^{n} r_j$$
$$N_c' = M$$

where M is the number of observations, and r_j (with $0 \leqslant j \leqslant n$) is the degree of the j^{th} equivalent polynomial.

An interesting family of RGMs was introduced in [2]. It is based on the following Galois field property (see [3])

$$[P(D)]^{2^n} = P\left(D^{2^n}\right)$$

where $P(D)$ is a LFSR generating polynomial. Therefore, fixed n, a set of equivalent high-degree generating polynomial is identified by: $P\left(D^2\right)$, $P\left(D^4\right)$, $\cdots$, $P\left(D^{2^n}\right)$. Each polynomial can generate its BGM (from the (7)), and the union of all these BGMs produces a RGM (from the (8)) of order n ($n=0$ clearly means a RGM made up of only one BGM based on $P(D)$). Furthermore, n is selected in agreement with

$$r_0 \cdot 2^n \leqslant M-1 \;\Rightarrow\; n \leqslant K \cdot \log\left(\frac{M-1}{r_0}\right) \tag{9}$$

where r_0 is $P(D)$ degree, M is the number of received observations, and $K = [\log(2)]^{-1}$. In other words, n is the largest integer that verifies the (9) inequality. We define these graphical models *Yeung*-RGM of order n (also pointed out YRGM$_n$).

B. Equivalent Sparse Polynomials with High-Degree

As demonstrated in [2], YRGMs offer great benefits in terms of acquisition probability at low-SNR, in the case of *sparse* generating polynomials (i.e., with only 3 or 4 nonzero coefficients). However, for *dense* generating polynomials (i.e., more than 4 nonzero coefficients) experiments suggest that poor performance is obtianed using TG models. This problem is common to many m-sequences and Gold codes that, having a dense polynomial, cannot be efficiently acquired using iMPAs on YRGMs.

In order to address this problem, the key idea is to find equivalent higher degree generating polynomials that are sparse and use these as BGMs to build new RGMs, following (7) and (8). These models will be denoted by adding the superscript *esp*, to identify RGMs generated by equivalent sparse polynomials of higher degree – e.g., RGM^{esp}. Furthermore, we will show these RGMs^{esp} provide better performance and lower complexity than YRGMs generated by initial dense polynomials. Now the problem is to search and identify these high degree equivalent polynomials with 3 or 4 nonzero coefficients. This result can be achieved by linear combination of the initial generating dense polynomial with its delayed versions until its high degree sparse versions are obtained.

An example is useful to well explain this procedure and illustrates the potential improvements. Consider the m-sequence identified by the following dense generating polynomial

$$P(D) = D^{12} + D^{11} + D^9 + D^7 + D^6 + D^5 + 1$$

with octal representation is $[15341]_8$. A equivalent sparse polynomial can be computed as [2]

$$\begin{aligned} P_{esp}(D) = P(D) \cdot [&D^{19} + D^{18} + D^{17} + D^{15} + D^{14} + D^{13} \\ &+ D^{12} + D^{10} + D^8 + D^6 + D^5 + D^2 + 1] \\ = D^{31} &+ D^2 + 1 \end{aligned}$$

with octal representation is $[20000000005]_8$. It is evident that $P_{esp}(D)$ is more sparse than $P(D)$ and its degree is higher. So, $P_{esp}(D)$ can be used as BGM to generate a YRGM^{esp} of order n (YRGM$_n^{esp}$). Assuming the signal model treated in section II, and considering 1024 observations, a comparison between the YRGM$_5^{esp}$, YRGM$_6$, BGM, and MLA is reported in Fig. 4. All of these models perform a full-parallel search with different performance and complexity. In particular, the YRGM$_5^{esp}$ gains about 7 to 8 dB with 30 iterations on the YRGM$_6$ that requires 100 iterations (the gain is larger than 10 dB if it is compared to the BGM). The complexity depends on the number of iterations run and number of edges per variable/check node. Assuming one min operation is equivalent to a sum operation, it is possible to evaluate the following complexity factors:[3]

$$\frac{C_{YRGM_5^{esp}}}{C_{YRGM_6}} < \frac{1}{24} \quad \text{and} \quad \frac{C_{YRGM_5^{esp}}}{C_{MLA}} \leqslant \frac{1}{4}$$

where C_{Mod} points out the complexity of one model (Mod is YRGM$_5^{esp}$, or YRGM$_6$, or MLA). These two factors measure the complexity of the YRGM$_5^{esp}$ with respect to the others. In the both cases, they demonstrate that the complexity is lower than the YRGM$_6$ (based on the primitive polynomial) and the MLA.

Nevertheless, not all cases are so simple to process. Indeed, some LFSR sequences are characterized by equivalent sparse polynomials with very high degrees that require a quite complex computational search to be identified and evaluated. In these cases, an exhaustive search on one sequence period is performed by a software simulation, finding all equivalent sparse polynomials that will be used to build RGMs^{esp}. An example is provided by GPS/SBAS Gold codes. Indeed, they are generated by the dense generating polynomial showed in (6) that has equivalent sparse polynomials with very high degree. Performing an exhaustive search (on one period, 1023 chips), it is possible to identify 341 equivalent sparse

[2]The sum is made modulo-2.

[3]The complexity is measured counting the number of sum operators per iteration and multiplying it by the number of iterations.

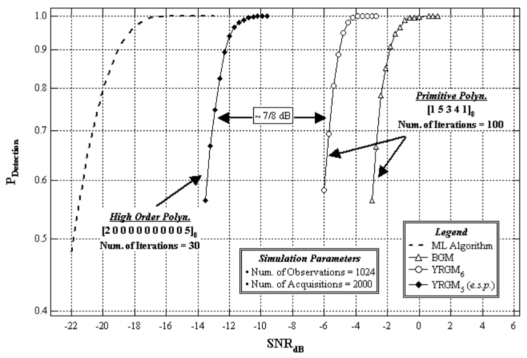


Fig. 4. Performance in case of the m-sequence $[15341]_8$ (r = 12).

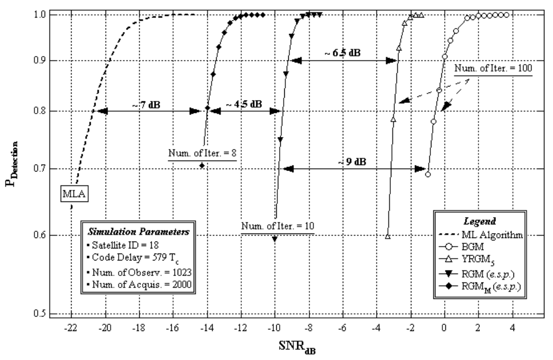


Fig. 5. Performance in case of the GPS/SBAS codes.

polynomials: only 1 with 3 coefficients, and 340 with 4 coefficients, that offer the possibility to generate a large number of different RGMs^{esp} with only 3/4 edges per check node and without 4-cycles. A pair of RGMs^{esp} are compared to the YRGM$_5$, the BGM and the MLA in Fig. 5. In particular, the RGM$_M^{esp}$ is the largest RGM achievable with all the 341 GPS/SBAS equivalent sparse polynomials, and the RGM^{esp} is generated grouping the BGMs obtained by the following sparse equivalent polynomials

$$\begin{aligned} P_{Eq,1}(D) &= D^{682} + D^{341} + 1 \\ P_{Eq,2}(D) &= D^{111} + D^{46} + D^{5} + 1 \\ P_{Eq,3}(D) &= D^{222} + D^{92} + D^{10} + 1 \\ P_{Eq,4}(D) &= D^{444} + D^{184} + D^{20} + 1 \\ P_{Eq,5}(D) &= D^{888} + D^{368} + D^{40} + 1. \end{aligned}$$

As in the previous example, both of these models present benefits in terms of detection probability with respect to the YRGM$_5$ and BGM, with fewer iterations. Furthermore, the complexity is lower than that of the YRGM$_5$ (generated by (6)), as follows

$$\frac{C_{RGM_M^{esp}}}{C_{YRGM_5}} < \frac{1}{3}, \quad \frac{C_{RGM^{esp}}}{C_{YRGM_5}} < \frac{1}{90}, \quad \frac{C_{RGM^{esp}}}{C_{MLA}} < \frac{1}{2}.$$

$C_{RGM_M^{esp}}$ and $C_{RGM^{esp}}$ are smaller than C_{YRGM_5}. Furthermore, $C_{RGM^{esp}}$ is smaller than C_{MLA}, but it is possible to verify that $C_{RGM_M^{esp}} > C_{MLA}$. Indeed, this method is most suitable for longer Gold sequences.

Both of these examples demonstrate equivalent sparse polynomials can be efficiently used to generate RGMs, on which low-complexity iMPAs are run, achieving good performance at low-SNR.

Another parameter to be considered, when iMPAs are implemented, is the memory required to store all node messages during each iteration. The memory requirements depends on the selected TG and, more specifically, on the number of edges. For this reason, large RGMs typically have large memory requirements. To address this problem, a different activation schedule is proposed. In the previous examples, all variable or check nodes were activated in parallel at the same time implying that all messages along edges could be stored. The memory requirements for large RGMs can be reduced by using a different activation schedule and a modified message passing algorithm. The activation schedule is based on breaking the RGM into a set of smaller sub-TGs, each one containing a portion of the parity checks (i.e., two different sub-TGs do not share common parity checks), and running the iMP sequentially on all sub-TGs (see also [16], [17]). This yields a reduction in memory requirements if the messages between variable and check nodes in a given sub-TG are not stored while iterating other sub-TGs. This, in fact, is not standard iterative message passing since these internal messages would normally be required for the next activation of the iMPA on the sub-TG. Nevertheless, this yields a significant decrease in memory requirements with a small performance degradation. More precisely, assume there are I_2 sub-TGs, one iteration is made for each sub-model (*inner iterations* = 1), and its soft output metrics become the input metrics in the next sub-model. The MP ends either when all check nodes of one sub-TG are verified or when I_1 outer iterations are performed. In this way, the computational complexity is the same as in the previous examples, but the the required memory is only that needed to store all messages of the largest sub-TG and, therefore, it is reduced. A pictorial representation of this schedule is reported in Fig. 6. In addition to not storing the internal messages between sub-TG iterations, we use a min-sum algorithm with damping factor α (see also [16], [17], and [18]).

A performance comparison between these two methods is displayed in Fig. 7. Indeed, the RGM$_M^{esp}$ acquisition is compared to the results achievable splitting this graph in 2 and 10 sub-TGs, with a damping factor 0.1. It is quite evident that the split models maintain the same rapid convergence of RGM$_M^{esp}$, but their performance can change in function of α and the characteristics of sub-TGs in which the initial RGM

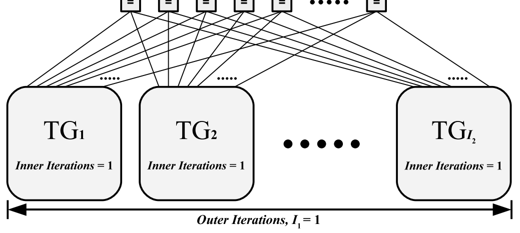


Fig. 6. Multi-TGs activation schedule.

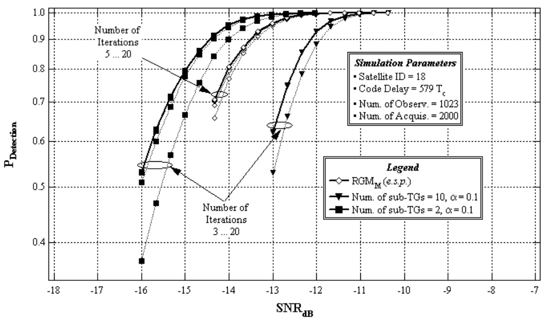


Fig. 7. Comparison between the two activation schedules.

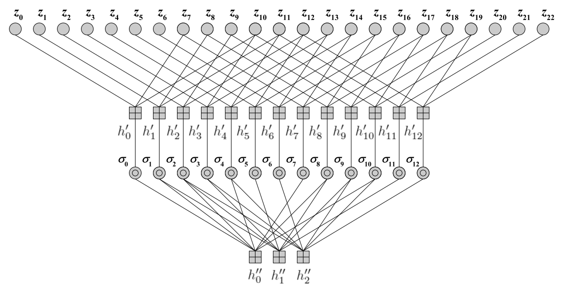

z_0 z_1 z_2 z_3 z_4 z_5 z_6 z_7 z_8 z_9 z_{10} z_{11} z_{12} z_{13} z_{14} z_{15} z_{16} z_{17} z_{18} z_{19} z_{20} z_{21} z_{22}
h'_0 h'_1 h'_2 h'_3 h'_4 h'_5 h'_6 h'_7 h'_8 h'_9 h'_{10} h'_{11} h'_{12}
σ_0 σ_1 σ_2 σ_3 σ_4 σ_5 σ_6 σ_7 σ_8 σ_9 σ_{10} σ_{11} σ_{12}
h''_0 h''_1 h''_2

Fig. 8. Example of hierarchical model for GPS/SBAS codes.

is separated. In this case, the required memory, it is about 1/2 in case of 2 sub-TGs and 1/10 in case of 10 sub-TGs, because the initial RGM was split in sub-TGs with about the same dimensions. Furthermore, all the models have the same complexity.

V. Hierarchical Model

In this section, we propose a distinct approach to acquire Gold codes using hierarchical models (HMs) built by their two generating m-sequences. In other words, introducing a set of *hidden variables* (see [7], [8], and [15]) opportunely defined, it is possible to generate two graphical models serially connected. So, the result is a hierarchical TG on which an iMPA can be run. The following example can clarify this concept.

Considering the primitive polynomials in (5) (GPS/SBAS LFSR generators), (4) can be expressed as

$$\begin{aligned} P(D) = P_{c'} \cdot D^{10} + P_{c'} \cdot D^{9} + P_{c'} \cdot D^{8} + P_{c'} \cdot D^{6} \\ + P_{c'} \cdot D^{3} + P_{c'} \cdot D^{2} + P_{c'} \end{aligned}$$

which leads to the system of equations

$$\sigma_{k-10} \triangleq z_k \oplus z_{k-3} \oplus z_{k-10} \tag{10a}$$

$$\begin{aligned} 0 = \sigma_k \oplus \sigma_{k-2} \oplus \sigma_{k-3} \oplus \sigma_{k-6} \\ \oplus \sigma_{k-8} \oplus \sigma_{k-9} \oplus \sigma_{k-10} \end{aligned} \tag{10b}$$

where the (10a) defines a generic hidden variable σ_k. It is evident that the system (10) identifies a concatenated structure because (10b) (*slave*) directly depends on (10a) (*master*). Furthermore, (10a) provides a first set of local constraints in function of the index k that are useful to generate a first preliminary model, $\mathbf{H}'$. A second set of constraints is got by the (10b), outputting the second graphical model $\mathbf{H}''$. These two models are related by their hidden variables. An example of this hierarchical TG is displayed in Fig. 8, in case of 23 observations. Specifically, a generic h'_i corresponds to a $\mathbf{H}'$ check node, while a generic h''_j is a check nodes of the $\mathbf{H}''$ matrix.

We also remark that this method can be applied to a generic set of Gold codes and the example in Fig. 8 is not the only way to realize a HM for GPS/SBAS sequences. Indeed, many different HMs can be constructed manipulating their generating m-sequence polynomials and inverting the master and slave polynomials. Of course, grouping together more HMs, graphical models with redundancy are obtained. Furthermore, the message updating algorithm for HMs is described in the appendix.

Some preliminary results, obtained using this technique, are displayed in Fig. 9. Specifically, HM$_1$ and HM$_2$ are both generated by $P_{c'}$ and $P_{c''}$, from (5), by inverting master and slave. The performance is better in the case of $P_{c''}$-master and $P_{c'}$-slave (HM$_2$). Furthermore, manipulating $P_{c''}(D)$ yields

$$\begin{aligned} P^{esp}_{c''}(D) &= P_{c''}(D) \cdot \left[D^3 + D^2 + 1\right] \\ &= D^{13} + D^4 + 1 \end{aligned}$$

where $P^{esp}_{c''}$-master and $P_{c'}$-slave are used to generate HM$_3$ and HM$_4$. In particular, in the case of HM$_4$, the $\mathbf{H}''$ is constructed as YRGM$_6$. So, from results in Fig. 9, it is evident that more sparse polynomials can generate models (HM$_3$) that provide better performance with fewer iterations than more dense models (HM$_1$ and HM$_2$) and, adding more redundancy (HM$_4$), this performance can be improved. Therefore, these results confirm the previous section conclusions.

VI. Conclusions

This paper focused on rapid acquisition of LFSR sequences with dense generating polynomials and Gold codes, using iMPAs. Exploiting the theorem reported in [11], we showed that these two cases are closely related, because every set of Gold sequences can be described by a high-order LFSR (typically dense) generator. So, in order to detect these SS sequences, iMPAs are run on RGMs generated by high degree equivalent polynomials that are very sparse. These polynomials can be algebraically computed by manipulation of dense primitive polynomials of LFSR sequences. Simulations results demonstrate that RGMs^{esp} offer benefits in terms of performance at low-SNR and low complexity respect graphical models based on dense polynomials.

Furthermore, addressing the problem of large memory requirements, a different activation schedule and modified message passing rules were proposed using the min-sum algorithm with a damping factor ([17] and [18]). This approach yields significant memory savings without a change in computational complexity as compared to the initial iMPAs on RGMs. This modification can provide good performance but requires care in selecting the sub-model partition and the damping factor.

In order to acquire Gold codes, HMs are also proposed. Our preliminary results demonstrate the richness of the available HMs for Gold codes, but the performance is not as good as

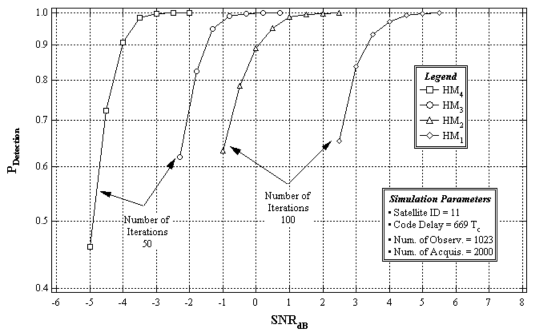


Fig. 9. Hierarchical model performance.

that obtained with redundant TG models. Thus, an interesting future direction is to further explore these HMs in search for better performance/complexity compromises.

Our results, take along with thoses in [1], [2], demonstrate that iMPAs can yield low-complexity, full parallel search for rapid PN search that approximates the ML method. Many topics still remain to be investigated, such as: the SS acquisition with the joint coarse estimation of code timing, and carrier phase and frequency, the improvement of HM or RGM^{esp} techniques, the search of other graphical models that could introduce more benefits, and more detailed hardware implementation considerations.

VII. Appendix

In order to describe the message updating proposed for HMs, a generic path of these graphs is extracted and shown in Fig. 10. Let be $\mathbf{z}$ an observation vector of M elements, the *soft-in* information (Δsi_i), in negative log-domain, is defined (see also [15]) as $\Delta si_i \triangleq -\log\left[\frac{\Pr(z_i|x_i=1)}{\Pr(z_i|x_i=0)}\right] = z_i,\ 0 \leqslant i \leqslant M-1$. Here all the main steps of our algorithm are reported.

Step 0 - Initialization ($n = 1$):

$$\Delta so_i \triangleq so_i[1] - so_i[0] = \Delta si_i$$

all other messages are zero.

Step 1 - Iteration n^{th}, $1 \leqslant n \leqslant I_{max}$:

$$\Delta\mu'_{i,j} = \Delta so_i - \Delta\eta'_{j,i}, \qquad \forall i{:}\ z_i \rightarrow h'_j$$

$$\Delta\alpha_j = \prod_{\forall i: z_i \rightarrow h'_j} \left[\mathcal{S}(\Delta\mu'_{i,j})\right] \cdot \min_{\forall i: z_i \rightarrow h'_j} \left(\left|\Delta\mu'_{i,j}\right|\right), \qquad \forall j{:}\ h'_j \rightarrow \sigma_j$$

$$\Delta\mu''_{j,k} = \Delta\alpha_j + \Delta\beta_j - \Delta\eta''_{k,j}, \qquad \forall j{:}\ \sigma_j \rightarrow h''_k$$

$$\Delta\eta''_{k,j} = \prod_{\substack{\forall l: \sigma_l \rightarrow h''_k \\ l \neq j}} \left[\mathcal{S}(\Delta\mu''_{l,k})\right] \cdot \min_{\substack{\forall l: \sigma_l \rightarrow h''_k \\ l \neq j}} \left(\left|\Delta\mu''_{l,k}\right|\right), \qquad \forall k{:}\ h''_k \rightarrow \sigma_j$$

$$\Delta\beta_j = \sum_{\forall k: h''_k \rightarrow \sigma_j} \Delta\eta''_{k,j} = S_{\beta_j} \cdot M_{\beta_j}, \qquad \forall j{:}\ \sigma_j \rightarrow h'_j$$

$$\Delta\eta'_{j,i} = S_{\beta_j} \cdot \prod_{\substack{\forall m: \\ z_m \rightarrow h'_j \\ m \neq i}} \mathcal{S}(\Delta\mu'_{m,j}) \cdot \min_{\substack{\forall m: \\ z_m \rightarrow h'_j \\ m \neq i}} \left[M_{\beta_j}, \left|\Delta\mu'_{m,j}\right|\right], \qquad \forall j{:}\ h'_j \rightarrow z_i$$

$$\Delta so_i = \Delta si_i + \sum_{\forall j: h'_j \rightarrow z_i} \left(\Delta\eta'_{j,i}\right), \qquad \forall i \in [0, M-1]$$

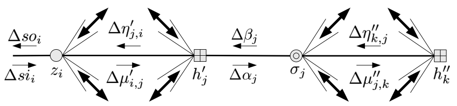


Fig. 10. One hierarchical model path.

where $\mathcal{S}(\centerdot) = \text{sgn}(\centerdot)$, $S_{\beta_j} = \mathcal{S}(\Delta\beta_j)$, and $M_{\beta_j} = |\Delta\beta_j|$.

The hard decision is made on the *soft-out* information (Δso_i). If the estimated vector verifies all the parity checks, the algorithm will end, otherwise $n = n + 1$ and the **Step 1** will restart. The algorithm definitely ends when the last iteration is performed ($n = I_{max}$).

References

[1] K. M. Chugg and M. Zhu, "A new approach to rapid pn code acquisition using iterative message passing techniques," *IEEE Journal on Selected Areas in Communications*, vol. 23, no. 5, pp. 884–897, May 2005.

[2] O. W. Yeung and K. M. Chugg, "An iterative algorithm and low complexity hardware architecture for fast acquisition of long pn codes in uwb systems," *Springer J. VLSI and Signal Processing (Special Issue on UWB Systems)*, vol. 43, April 2006.

[3] M. K. Simon, J. K. Omura, R. A. Scholtz, and B. K. Levitt, *Spread Spectrum Communications Handbook*. McGraw-Hill TELECOM, 2002.

[4] J. G. Proakis, *Digital Communications*, 4th ed., S. W. Director, Ed. McGraw-Hill, 2001.

[5] R. L. Pickholtz, D. L. Schilling, and L. B. Milstein, "Theory of spread spectrum communications - a tutorial," *IEEE Transactions on Communications*, vol. COM-30, no. 5, pp. 855–884, May 1982.

[6] R. M. Tanner, "A recursive approach to low complexity codes," *IEEE Transactions on Information Theory*, vol. 27, no. 5, pp. 533–547, September 1981.

[7] N. Wiberg, "Codes and decoding on general graphs," Ph.D. dissertation, Linköping University, S-581 83 Linköping, Sweden, 1996.

[8] F. R. Kschichang, B. J. Frey, and H. A. Loeliger, "Factor graph and the sum-product algorithm," *IEEE Transactions on Information Theory*, vol. 47, no. 2, pp. 498–519, February 2001.

[9] B. Vigoda, J. Dauwels, M. Frey, N. Gershenfeld, T. Koch, H. A. Loeliger, and P. Merkli, "Synchronization of pseudorandom signals by forward-only message passing with application to electronic circuits," *IEEE Transactions on Information Theory*, vol. 52, no. 8, pp. 3843–3852, August 2006.

[10] L. Yang and L. Hanzo, "Acquisition of m-sequences using recursive soft sequential estimation," *IEEE Transactions on Communications*, vol. 52, no. 2, pp. 199–204, February 2004.

[11] R. Gold, "Optimal binary sequences for spread spectrum multiplexing," *IEEE Trans. Information Theory (Correspondence)*, vol. IT-13, pp. 619–621, October 1967.

[12] M. Aparicio and *et al.*, *Global Positioning Systems: Theory and Applications*, B. W. Parkinson and J. J. Spilker, Eds. 370 L'Enfant Promenade, SW, Washington, DC 20024-2518: American Institute of Aeronautics and Astronautics, Inc., 1996, vol. I and II.

[13] R. G. Gallager, "Low-density parity-check codes," *IRE Transactions on Information Theory*, pp. 21–28, January 1962.

[14] S. Benedetto, D. Divsalar, G. Montorsi, and F. Pollara, "Soft-input soft-output modules for the construction and distributed iterative decoding of code networks," *European Transactions on Telecommunications*, vol. 9, no. 2, March-April 1998.

[15] K. M. Chugg, A. Anastasopoulos, and X. Chen, *Iterative Detection: Adaptivity, Complexity Reduction, and Applications*. Norwell, Massachusetts 02061: Kluwer Academic Publishers, 2001.

[16] J. Jiang and K. R. Narayanan, "Iterative soft decision decoding of Reed Solomon codes," *IEEE Communications Letters*, vol. 8, no. 4, pp. 244–246, April 2004.

[17] T. R. Halford and K. M. Chugg, "Random redundant soft-in soft-out decoding of linear block codes," in *Proc. IEEE International Symp. on Information Theory*, Seattle, WA, July 2006.

[18] J. Chen and M. P. C. Fossorier, "Near optimum universal belief propagation based decoding of low-density parity-check codes," *IEEE Transactions on Communications*, vol. 50, no. 3, pp. 406–414, March 2002.