PN Code Acquisition Using Boolean Satisfiability Techniques

Fadi A. Aloul
Department of Computer Science & Engineering
American University of Sharjah, UAE
faloul@aus.edu

Mohamed El-Tarhuni
Department of Electrical Engineering
American University of Sharjah, UAE
mtarhuni@aus.edu

***Abstract* — In mobile radio communication systems using spread spectrum technology, an accurate estimate of the signal propagation delay is needed in order to recover the transmitted data. This process is usually done using a pseudo noise (PN) code acquisition algorithm to search for the correct propagation delay within a window of possible delays. In this paper, we propose a new approach to solving the PN code acquisition problem using advanced Boolean Satisfiability (SAT) techniques. SAT solvers use intelligent search algorithms that can traverse the search space and *efficiently* prune parts that contain no solutions. These solvers have recently been used to solve many challenging problems in Engineering and Computer Science. In this paper, we show how to formulate the PN code acquisition problem as a SAT instance and evaluate the use of advanced SAT techniques in solving the problem. Our approach is verified by simulation and presented results indicate that the proposed system achieves a detection probability for the correct delay of almost 100% for all practical cases.**

I. INTRODUCTION

Direct sequence spread spectrum (DS-SS) has recently been proposed for use in third generation mobile communications systems such as CDMA2000 and WDCMA. In DS-SS systems, a wideband pseudo noise (PN) code is used to spread the spectrum of the data at the transmitter before it is transmitted over the radio channel. This bandwidth spreading results in several features such as resistance to multipath, accurate ranging, and multi-user communications. Due to signal reflection, refraction, and scattering in the radio channel, the transmitted signal goes through several propagation paths before reaching the receiver. It is imperative that the receiver collects the energy received through all these paths in order to have good system performance. This is typically done through what is known as a RAKE receiver which consists of several correlators, called fingers, each assigned to collect the energy of one of the multipath components.

The PN code synchronization process is usually developed over two stages: PN code acquisition and PN code tracking [20]. The acquisition stage is responsible for the coarse timing adjustment such that the local de-spreading code is within one or one-half chip from the received signal code. PN code tracking is then used to refine the timing and to maintain alignment to within a fraction of a chip (e.g. one-eighth of a chip). This paper focuses on the PN code acquisition problem.

PN code acquisition can be accomplished by searching the time delay uncertainty range for the correct multipath delays. The uncertainty range represents the possible delays that the signal may have and is related to the channel memory. The delay range is usually specified as cells that are one-chip or one-half of a chip apart. The search of these cells, i.e. finding the cells that have strong energy and hence multipath components, can be either done in a serial or parallel fashion [19, 22, 23, 24].

In serial search, one cell at a time is tested by measuring the signal energy at that cell. If the energy exceeds a preset threshold then the cell is declared as a multipath cell while if the energy is below the threshold then it is declared as a no multipath cell. The search advances to the next cell and the process is continued until all cells in the uncertainty range are tested. The other search strategy uses parallel search where the energies of all cells are calculated simultaneously using a parallel circuits and cells with energy above the threshold are declared as multipath cells. Apparently, serial search is slower compared to parallel search at it takes longer time to search all the cells and find the delays. On the other hand, serial search has a much lower reduced complexity (both hardware and processing).

A common drawback of the existing schemes is that in searching for the correct cells they don't utilize the inherent structure of the PN code. These schemes need to search all possible cells in the search window, which could be as large as the length of the PN code, in order to find the correct cells. For example, for a PN code with a length of 2047 chips (generated by 11-stage shift register) the serial and parallel search schemes need to test 2047 cells if the search step is one chip or twice of that if the search step is one-half of a chip. This testing may need to be repeated many times if the multipath components were not detected at the first trial due to noise. In this paper, we propose a PN code acquisition scheme that exploits the structure of the PN code to reduce the number of decisions needed to find correct cells. The proposed scheme is based on using Boolean Satisfiability (SAT) solving to perform intelligent search of the uncertainty region and hence reduce the number of decisions needed to find the correct cells significantly. This is done by searching only PN code phases that result in minimum difference (minimum distance) between the PN code in the received signal and a locally generated PN code according to the SAT formulation (to be explained later).

Recently, Boolean Satisfiability (SAT) have been shown to be very successful in solving complex problems in various Engineering and Computer Science applications. Such applications include: Formal Verification [5], FPGA routing [17], Power Optimization [3], etc. SAT has also been extended to a variety of applications in Artificial Intelligence including other well known NP-complete problems such as graph colorability, vertex cover, hamiltonian path, and independent sets [8]. Despite SAT being an NP-Complete problem [7], many researchers have developed powerful SAT solvers that are able of handling problems consisting of thousands of variables and millions of

constraints [2, 15, 16]. Briefly defined, the SAT problem consists of a set of Boolean variables and a set of constraints expressed in product-of-sum form. The goal is to identify an assignment to the variables that would satisfy all constraints or prove that no such assignment exists.

Even though in recent years we have seen a surge in the application of SAT techniques to assist in finding solutions to various Engineering problems, very few researchers reported on the use of SAT-based techniques in mobile communication related research.

The formulation of the PN acquisition as a SAT instance to develop the search strategy is described in details in this paper. Simulation results indicate that the proposed approach is *complete* and is guaranteed to identify the optimal solution (correct cells), if one exists.

The reminder of this paper is organized as follows. Sections II and III present the signal model and an overview of SAT, respectively. Section IV describes the proposed scheme and shows how to formulate the PN code acquisition problem as a SAT instance. Simulation results are presented and discussed in Section V. Finally, the paper is concluded in Section VI.

II. SYSTEM DESCRIPTION

In this section, we provide a short discussion on the signal model used for establishing PN code acquisition. We then discuss the acquisition process and its objectives. Finally, we introduce the Boolean SAT solving technique and provide simple examples to familiarize the reader with its use.

2.1 Signal model

A direct-sequence code division multiple access (DS-CDMA) system is investigated in this paper. The signal model used is similar to that used in CDMA-2000 system where a separate pilot signal is code multiplexed with the data (traffic) channel for each user to allow for PN code acquisition and tracking as well as channel estimation. The transmitted signal from the desired user is given by:

$$s_1(t) = \sqrt{P_1}\sum_i (d_{1i}W + \sqrt{G_p}) \sum_{k=0}^{N-1} c_{1k}\, g(t - iT_b - kT_c) \quad (1)$$

where P_1 is the transmitted power, d_{1i} is the i^{th} information bit (± 1), W is the Walsh code used to separate the pilot channel from the traffic channel, G_p is the pilot channel power gain relative to the traffic channel, c_{1k} is the spreading pseudo random (PN) code of the desired user of ± 1, N is the PN code length which is the same as the number of chips per bit, i.e. $N = T_b / T_c$, T_b is the bit duration, T_c is the chip duration, and $g(t)$ is the chip pulse shape.

We assume that the channel model used is Additive White Gaussian Noise (AWGN). This model was used to simplify the presentation of the proposed SAT algorithm and the results can be extended to a more practical channel models used in mobile radio systems with multipath fading. The received signal is given by

$$u(t) = s(t - \tau) + n(t) \quad (2)$$

Where τ is the channel delay that we would like to estimate and $n(t)$ is an additive white Gaussian noise (AWGN) with zero mean and two-sided power spectral density $N_0/2$ that models the effect of the receiver noise. It is assumed that the number of users *(M)* is relatively large such that the interference can be modeled as part of the AWGN model.

2.2 PN Code Acquisition Process

To maximize the signal-to-noise ratio, the received baseband signal is first applied to a chip-matched filter to produce the following signal

$$z[k] = \int_{(k-1)T_c}^{kT_c} u(t)g(t)dt \quad (3)$$

In conventional acquisition schemes, the output of the chip-matched filter is correlated with a locally generated PN code with different offsets that cover the delay uncertainty region (possible the whole PN code period) as follows

$$Y[i] = \sum_{k=0}^{N-1} z[k]c_1[k-i], \qquad i = 0, 1, \ldots, N-1 \quad (4)$$

where the index *i* indicates the delay offset. The correlation results in (4) are used to estimate the energy at different delay offsets and a decision is made on the multipath delays based on the highest energy values. It is also common to use a preset threshold where only energy values that exceed the threshold are declared as correct multipath components while others are ignored.

Based on the outcome of the decision process, we can have one of the following events:

- **Detection:** This event occurs when the energy value exceeds the threshold and the estimated delay matches one of the actual delays of the multipath components in the received signal. We would like to maximize the detection probability to improve the performance of the RAKE receiver in detecting the data signal.
- **False Alarm:** Occurs when the energy value exceeds the threshold but the estimated delay did not match any of the actual delays of the multipath components. We would like to minimize the false alarm probability since the RAKE receiver would be using a signal that has no useful energy to detect the data.
- **Miss:** Occurs when the energy value is below the threshold but the delay offset has a correct multipath component. We would like to minimize such event since the RAKE receiver will not get all useful energy in detecting the data.

We would like to remark that it is not possible to achieve the goals for the three events mentioned above simultaneously and a trade-off is usually needed. Another important objective is to minimize the resources needed to accomplish PN code acquisition including both hardware and processing dimensions.

III. BOOLEAN SATISFIABILITY

The last few years have seen significant advances in Boolean satisfiability (SAT) solving. These advances have lead to the successful deployment of SAT solvers in a wide range of problems in Engineering and Computer Science. Given a set of Boolean variables and a set of constraints expressed in product-of-sum form, the goal is to find a variable assignment that satisfies all constraints or prove that no such assignment exists. The term "Satisfiability" emerges from that fact that we are asked to find a satisfying assignment, while the term "Boolean" comes from the fact that such assignment consists of only *true* or *false* variable states.

The SAT problem is usually expressed in conjunctive normal form (CNF). A CNF formula φ on n binary variables $x_1, \ldots, x_n$ is the conjunction (AND) of m clauses $\omega_1, \ldots, \omega_m$ each of which is a disjunction (OR) of one or more literals, where a literal is the occurrence of a variable or its complement. A formula φ maps to a unique n-variable Boolean function $f(x_1, \ldots, x_n)$ [14]. Clearly, a function f can be represented by many equivalent CNF formulas. We will refer to a CNF formula as a *clause database* and use "formula" and "CNF formula" interchangeably.

A variable x is said to be *assigned* when its logical value is set to 0 or 1 and *unassigned* otherwise. A literal l is a *true* (*false*) literal if it evaluates to 1 (0) under the current assignment to its associated variable, and a *free literal* if its associated variable is *unassigned*. A clause is said to be *satisfied* if at least one of its literals is true, *unsatisfied* if all of its literals are set to false, *unit* if all but a single literal are set to false, and *unresolved* otherwise. A formula is said to be satisfied if all its clauses are satisfied, and unsatisfied if at least one of its clauses is unsatisfied. In general, the SAT problem is defined as follows: Given a Boolean formula in CNF, find an assignment of variables that satisfies the formula or prove that no such assignment exists.

In the following example, the CNF formula:

$$\varphi = (a \vee b) \cdot (\bar{b} \vee c) \cdot (\bar{a} \vee c) \quad (5)$$

consists of 3 variables, 3 clauses, and 6 literals. The assignment $\{a = 1, b = 0, c = 0\}$ violates the third clause and unsatisfies φ, whereas the assignment $\{a = 1, b = 0, c = 1\}$ satisfies φ. Note that a problem with n variables will have 2^n possible assignments to test. The above example with 3 variables has 8 possible assignments.

Despite the SAT problem being NP-Complete [7], there have been dramatic improvements in SAT solver technology over the past decade. This has lead to the development of several powerful SAT algorithms that are capable of solving problems consisting of thousands of variables and millions of constraints. Such solvers include: GRASP [15], zChaff [16], Berkmin [13], and MiniSAT [12]. In the next three sections, we describe the basic SAT search algorithm, recent extensions to the SAT solver input, and the use of hardware with SAT.

3.1 Backtrack Search

Most modern complete SAT algorithms can be classified as enhancements to the basic Davis-Logemann-Loveland (DLL) backtrack search approach [10]. The DLL procedure performs a search process that traverses the space of 2^n variable assignments until a satisfying assignment is found (the formula is satisfiable), or all combinations have been exhausted (the formula is unsatisfiable). It maintains a *decision tree* to keep track of variable assignments and can be viewed as consisting of three main engines: (1) *Decision* engine that makes *elective* assignments to the variables, (2) *Deduction* engine that determines the consequences of these assignments, typically yielding additional *forced* assignments to, i.e. implications of, other variables, and (3) *Diagnosis* engine that handles the occurrence of conflicts, i.e. assignments that cause the formula to become unsatisfiable, and backtracks appropriately. An example of a decision tree is shown in Figure 1.

$$f(a, b, c, d) = (a \vee b \vee c) \cdot (a \vee b \vee \bar{c}) \cdot (\bar{a} \vee c \vee d) \cdot (\bar{a} \vee c \vee \bar{d}) \cdot (\bar{b} \vee \bar{c} \vee d) \cdot (\bar{b} \vee \bar{c} \vee d)$$

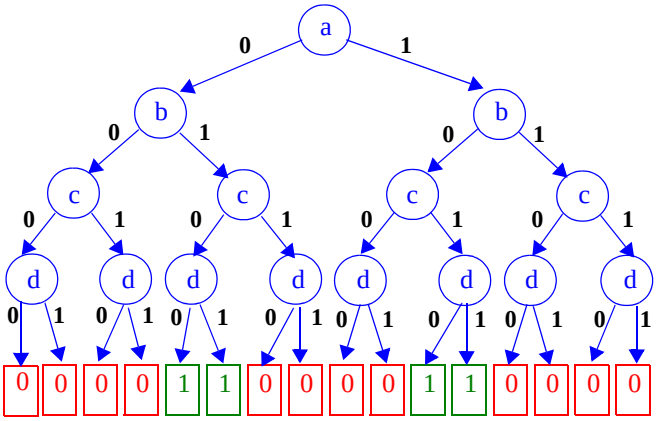


Fig. 1. An example of a satisfiable SAT instance showing its corresponding decision tree.

Recent studies have proposed the use of the *conflict analysis* procedure in the diagnosis engine [15]. The idea is whenever a conflict is detected, the procedure analyzes the variable assignments that cause one or more clauses to become unsatisfied. Such analysis can identify a small subset of variables whose current assignments can be blamed for the conflict. These assignments are turned into a *conflict-induced clause* and augmented with the clause database to avoid regenerating the same conflict in future parts of the search process. In essence, the procedure performs a form of learning from the encountered conflicts. Today, conflict analysis is implemented in almost all SAT solvers [13, 15, 16].

3.2 More Expressive Input

Restricting the input of SAT solvers to CNF formulas can restrict their usage in various domains. Therefore, researchers have focused on extending SAT solvers to handle stronger input representations. Specifically, SAT solvers [2, 6, 11, 12, 18] have recently been extended to handle pseudo-Boolean (PB) constraints which are linear inequalities with integer coefficients that can be expressed in the normalized form [2] of:

$$a_1 x_1 + a_2 x_2 + \ldots + a_n x_n \geq b \quad (6)$$

where $a_i, b \in Z^+$ and x_i are literals of Boolean variables. Note that any CNF clause can be viewed as a PB constraint, e.g. clause $(a \vee b \vee c)$ is equivalent to $(a + b + c \geq 1)$.

PB constraints can, in some cases, replace an exponential number of CNF constraints. They have been found to be very efficient in expressing "counting constraints" [2]. Furthermore, PB extends SAT solvers to handle *optimization* problems as opposed to only *decision* problems. Subject to a given set of CNF and PB constraints, one can request the minimization (or maximization) of an objective function which consists of a linear combination of the problem's variables.

$$\sum_{i=1}^{n} a_i x_i \tag{7}$$

This feature has introduced many new applications to the SAT domain. Recent studies have also shown that SAT-based optimization solvers can in fact compete with the best generic integer linear programming (ILP) solvers [2, 6].

3.3 Hardware-Based SAT Solvers

Note that SAT solvers can be implemented in hardware. Several studies proposed the use of FPGA reconfigurable systems to solve SAT problems [1, 25]. Hardware solvers could be a standalone or as an accelerator where the problem is partitioned between the hardware solver and the attached computer using software. Many different architecture were proposed to solve SAT problems in hardware. Linearly connected set of finite state machines, control unit, and deduction logic was proposed in [25]. The authors in [25] implemented their algorithm on Xilinx XC4028 FPGA. While in [1], the authors proposed a technique for modeling any boolean expression. Their objective is to set the function output to 1. A backtrack algorithm is used to propagate the output back to the input and finding an assignment of the inputs to satisfy a logical 1 at the output.

The authors in [9] proposed an architecture for evaluating clauses in parallel. In their architecture, the clauses are separated into a number of groups and the deduction is performed in parallel. Then the results are merged together to allow the assignment to the variables.

A software/hardware solver for SAT was introduced in [21]. In their approach, they minimized the hardware compilation time which greatly reduced the total time to solve the problem. They also implemented their solver on an FPGA.

IV. SAT MODEL FOR PN CODE ACQUISITION

A hard decision is made on the matched filter output in (3) to get a binary sequence of data, z_b, that represent an estimate of the PN code in the received signal as follows

$$z_b[k] = \begin{cases} 1 & z[k] \geq 0 \\ 0 & z[k] < 0 \end{cases} \tag{8}$$

Although hard decisions are in general not sufficient statistics for estimating the delay, but in the context of the developed SAT model for PN acquisition it would be enough to provide an estimate of the received PN code and hence allows for the SAT search to be implemented as will be discussed later.

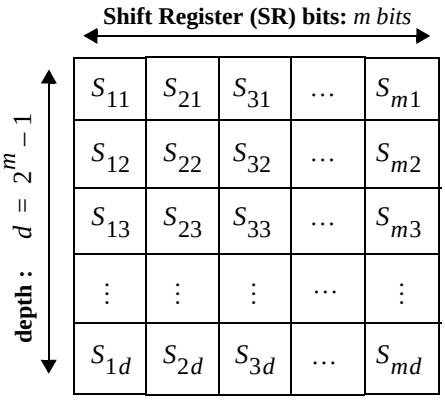


Fig. 2. Sample layout of Shift Register bits.

In this paper we are interested in using advanced SAT solvers to solve the PN code acquisition problem. To illustrate our approach, lets assume a system consisting on *n* data bits and *m* Shift Register (SR) bits. The depth *d* is equal to $(2^m - 1)$ levels as shown in Figure 2. The objective is to (1) find all solutions that satisfy the presented constraints, (2) for each solution we compute the number of bits in which the matched filter output z_b is different from the locally generated PN code, and (3) select the solution that has the smallest difference (error) as the shift register state the corresponds to the delay offset estimate

Three sets of *variables* are defined for the problem:

- A Boolean variable C_i is defined for each data bit *i*. That is a total of *n* variables. A value of 1 (0) for each variable indicates that the corresponding bit is a 1 (0) in the original source.
- A Boolean variable Q_i is defined for each chip as the difference between the C_i and the PN code chip. That is a total of *n* variables.
- A Boolean variable S_{ij} is defined for each SR bit *i* at each level *j*. That is a total of $m \times (2^m - 1)$ variables.

The total number of needed Boolean variables is equal to $2n + m(2^m - 1)$.

The following set of *constraints* are generated:

- **Data Bit Constraints:** For each data bit *i*, its corresponding C_i bit is set to 0 or 1 depending on the feeded data. This can be expressed using a single PB constraint per data bit as follows:

  $$C_i = v \quad ; v \in 0, 1 \text{ based on input value; } i = 1, \ldots n \tag{9}$$

  Thats a total of *n* PB constraints.
- **Initial State Constraints:** The initial SR bits should have at least one bit assigned to 1. This can be expressed using a *single* PB constraint as follows:

Logical Constraint	CNF Constraint
$(x = y)$	$(\bar{x} \vee y) \cdot (x \vee \bar{y})$
$(x = y \oplus z)$	$(\bar{x} \vee y \vee z) \cdot (x \vee \bar{y} \vee z) \cdot$ $(x \vee y \vee \bar{z}) \cdot (\bar{x} \vee \bar{y} \vee \bar{z})$

TABLE 1. Expressing logical constraints using CNF constraints.

$$\left(\sum_{i=1}^{m} S_{i1} \right) > 0 \qquad (10)$$

- **Shifting Constraints:** The shifting within the shift register relation, e.g. $S_{22} = S_{11}, S_{32} = S_{21}, \ldots$, is expressed using the following equality constraint per SR bit:

$$(S_{il} = S_{(i-1)(l-1)}) \quad ; \; l = 2, \ldots, d; \; i = 2, \ldots, m \qquad (11)$$

This results in a total of $(m-1)(2^m - 2)$ equality constraints. Each equality constraint of format $(x = y)$ can be expressed using two CNF constraints as shown in Table 1.

- **Feedback Constraints:** The PN code feedback relation, e.g. $S_{12} = S_{p1} \oplus S_{q1}$, is expressed using the following XOR constraint per initial SR bit:

$$[S_{1l} = S_{p(l-1)} \oplus \ldots \oplus S_{q(l-1)}] \quad ; \; l = 2, \ldots, d \qquad (12)$$

where $p, q \in \{1, \ldots, n\}$ according to the feedback connection of the PN code generator. This results in a total of $(2^m - 2)$ XOR constraints. Each XOR constraint of format $(x = y \oplus z)$ is expressed using four CNF constraints as shown in Table 1.

- **Difference Constraints:** The Q relation between the data and SR bits is expressed as follows:

$$[Q_i = S_{(m)(((i+d-1) mod(d))+1)} \oplus C_i] \; ; \; i = 1, \ldots, n \qquad (13)$$

This results in n XOR constraints. As mentioned earlier each XOR constraint can be expressed using four CNF constraints.

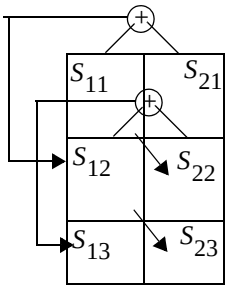
- **Optimization Function:** The final optimization goal is to minimize the sum of *Qs*. This is expressed using the following PB optimization objective:

$$Min\left(\sum_{i=1}^{n} Q_i \right) \qquad (14)$$

4.1 Illustrative Example

To further illustrate the formulation in SAT input, lets consider the example in Figure 3. The system consists of 8 data bits and 2 SR bits. Hence, the depth d is 3. The SAT problem generates a total of $2 \times 8 + 2(2^2 - 1) = 22$ Boolean variables. The figure displays the needed constraints.

Data Input Bits: 0, 1, 0, 0, 1, 0, 0, 0



Constraints:

$C_1 = 0$ $C_5 = 1$

$C_2 = 1$ $C_6 = 0$

$C_3 = 0$ $C_7 = 0$

$C_4 = 0$ $C_8 = 0$

$S_{22} = S_{11}$

$S_{23} = S_{12}$

$S_{12} = S_{11} \oplus S_{21}$

$S_{13} = S_{12} \oplus S_{22}$

$S_{11} + S_{21} > 0$

$min(Q_1 + Q_2 + \ldots + Q_8)$

$Q_1 = S_{21} \oplus C_1$

$Q_2 = S_{22} \oplus C_2$

$Q_3 = S_{23} \oplus C_3$

$Q_4 = S_{21} \oplus C_4$

$Q_5 = S_{22} \oplus C_5$

$Q_6 = S_{23} \oplus C_6$

$Q_7 = S_{21} \oplus C_7$

$Q_8 = S_{22} \oplus C_8$

Fig. 3. An example of a network with 8 data bits and 2 SR bits.

V. SIMULATION RESULTS

In this paper, we simulated a spread spectrum system with a PN code of length 2047 (11-stage shift register) with a single path channel. Other users in the system (interference) have been modeled as AWGN and result in increasing the noise level in the received signal. To reduce the simulation time, we choose to model the effect of the noise as causing random errors in the received PN code. In contrast, we can run the simulations for a specific signal-to-noise ratio (SNR) and use the detected chips as the input to the SAT PN code acquisition algorithm. We would like to remark that there is no difference in the expected results.

Initially, we started with a received signal that has zero error in the PN code received (all chips are correct in equation (5)). Although this is unrealistic since it means we have extremely high signal-to-noise ratio, but it serves as a validation of the functionality of the proposed SAT-based PN code acquisition scheme. Then, we investigated different scenarios where more realistic levels of signal-to-noise ratio are used that result in a different percentage of errors in the received PN code. For example, we will include results with 10-20% errors (high SNR), 30-40% errors (medium SNR) and 50% errors (low SNR). To clarify this point, with 10% errors in the received PN code, we have introduced about 204 errors at random locations in the sequence of 2047 PN code. The simulation is repeated 500 times for the same number of errors but with randomly generated locations and the length of the received vector is assumed to be five PN code periods (total of 10235 chips).

The SAT-based algorithm searched for the possible states that match the received signal with the PN code and the state that

Error Rate (%)	Minimum Distance	Probability of Detection	Number of Decisions
0	0	100	13
10	1023	100	2089
20	2047	100	2103
30	3070	100	2109
40	4094	100	2114
50	4943	0	2116

TABLE 2. SAT-base PN code acquisition simulation results.

results in minimum difference is used to find the delay estimate. Table 2 shows the minimum distance found at different error percentages. The detection probability for the different scenarios is also shown in Table 2. As we can see all the cases show a very high detection probability meaning that the SAT-based system was able to determine the correct delay offset in all 500 experiments. The only case where it failed was when the errors were so significant (50% error rate) and that resembles a very low SNR that is not typically expected in real systems.

The second measure of performance used to quantify the advantage of the SAT-based system over conventional serial or parallel search strategies is the number of decisions made in order to find the correct delay. As Table 2 illustrates, the SAT-based system was able to find the delay after about 2000 decisions for most of the cases and with only 13 decisions when the error rate was 0%. Note that the number of variables in the SAT-algorithm is about 42987 and for a brute force searcher it would require 2^{42987} decisions to find the optimum solution. This illustrates the advantage of the SAT-algorithm as the reduction is quite extensive and indicated the feasibility of implementing such algorithm in practice.

VI. CONCLUSIONS

In this paper, a new algorithm for PN code acquisition for mobile radio systems using Boolean satisfiability (SAT) techniques is presented. The algorithm uses the structure of the PN code to find the propagation delay and hence synchronize the transmitted PN code with a locally generated code. The paper demonstrates how to model the PN code acquisition problem as a SAT problem. Simulation results showed that the proposed scheme was successful in providing correct delay estimates without any error for most practical cases. The computational complexity is also shown to be very small compared to brute force search.

REFERENCES

[1] M. Abramovici, and D. Saab "Satisfiability on Reconfigurable Hardware," in *Proc. of the Int'l Workshop on Field Programmable Logic and Application*, 448-456, 1997.

[2] F. Aloul, A. Ramani, I. Markov, and K. Sakallah, "Generic ILP Versus Specialized 0-1 ILP: An Update," in *Proc. of Int'l Conference on Computer-Aided Design* (ICCAD), 450-457, 2002.

[3] F. Aloul, S. Hassoun, K. Sakallah, and D. Blaauw, "Robust SAT-Based Search Algorithm for Leakage Power Reduction," in *Proc. of* PATMOS, 167-177, 2002.

[4] B. Al-Rawi and F. Aloul, PBS4 SAT Solver, 2005. Available at: *http://www.aloul.net/Tools/pbs4*

[5] A. Biere, A. Cimatti, E. Clarke, M. Fujita, and Y. Zhu, "Symbolic Model Checking using SAT procedures instead of BDDs," in *Proc. of Design Automation Conference* (DAC), 317-320, 1999.

[6] D. Chai and A. Kuehlmann, "A Fast Pseudo-Boolean Constraint Solver," in *Proc. of* DAC, 830-835, 2003.

[7] S. A. Cook, "The Complexity of Theorem Proving Procedures," in *Proc. of the Symp. on the Theory of Computing*, 151-158, 2004.

[8] N. Creignou, S. Kanna, and M. Sudan, "Complexity Classifications of Boolean Constraint Satisfaction Problems," *SIAM Press*, 2001.

[9] A. Dandalis, and V. K. Prasanna, "Run-time Performance Optimization of an FPGA Based Deduction Engine for SAT Solvers," in *ACM TODAES*, 7(4), 547-562, October 2002.

[10] M. Davis, G. Longman, and D. Loveland "A Machine Program for Theorem Proving," in *J. of the ACM*, 5(7), 394-397, 1962.

[11] H. Dixon and M. Ginsberg, "Inference Methods for a Pseudo-Boolean Satisfiability Solver," in *Proc. of National Conference on Artificial Intelligence* (AAAI), 635-640, 2002.

[12] N. Een and N. Sorensson, "An Extensible SAT-solver," in *Proc. of the SAT*, 502-508, 2003.

[13] E. Goldberg and Y. Novikov, "BerkMin: A Fast and Robust SAT-solver," in *Proc. of Design Automation and Test in Europe Conference* (DATE), 142-149, 2002.

[14] J. Hayes, "Introduction to Digital Logic Design," *Addison-Wesley*, 1993.

[15] J. Marques-Silva and K. Sakallah "GRASP: A search algorithm for propositional satisfiability," in *IEEE Trans. on Computers*, 48(5), 506-521, 1999.

[16] K. Moskewicz, C. Madigan, Y. Zho, and S. malik "Chaff: Engineering an efficient SAT solver," in *Proc. of Design Automation Conference* (DAC), 503-535, 2001.

[17] G. Nam, F. A. Aloul, K. A. Sakallah, and R. Rutenbar, "A Comparative Study of Two Boolean Formulations of FPGA Detailed Routing Constraints," in *Proc. of International Symposium on Physical Design* (ISPD), 222-227, 2001.

[18] H. Sheini and K. Sakallah, "Pueblo: A Modern Pseudo-Boolean SAT Solver," in *Proc. of Design, Automation and Test in Europe Conference* (DATE), vol. 2, 684-685, 2005.

[19] O.-S. Shin and K.B. E. Lee, "Utilization of Multipaths for spread-pectrum Code Acquisition in Frequency-selective Rayleigh Fading Channels," in *IEEE Trans. Comm.*, v. 49, 734-743, Apr. 2001.

[20] M. Simon, J. Omura, R. Scholtz and B. Levitt, "Spread Spectrum Communications," *McGraw-Hill Inc.*, New York, 1994.

[21] I. Skliarova and A. Ferrari "A Software/Reconfigurable Hardware SAT Solver," in *IEEE Trans. on Very Large Scale Systems*, 12(4), 408-419, April 2004.

[22] W. Suwansantisuk and Z. Win, "Multipath Aided Rapid Acquisition: Optimal Search Strategies," in *IEEE Trans. Info. Theory*, v. 53, 174-193, January 2007.

[23] W. Suwansantisuk, Z. Win, and L. Shepp, "On the Performance of Wide-Bandwidth Signal Acquisition in Dense Multipath Channels," in *IEEE Trans. Veh. Tech.*, v. 54, 1584-1594, Sep. 2005.

[24] L.-L. Yang and L. Hanzo, "Serial Acquisition Performance of Single-Carrier and Multicarrier DS-CDMA over Nakagami-m fading Channels," in *IEEE Trans. Wireless Comm.*, v. 1, 692-702, 2002.

[25] P. Zhong, M. Martonosi, P. Ashar, and S. Malik, "Using Reconfigurable Computing to Accelerate Boolean Satisfiability," in *IEEE Trans. on Computer Aided Design of Integrated Circuits and Systems*, 18(6), 861-868, 1999.