Analyzing Stability of Algorithmic Systems Using Algebraic Constructs

Susmit Bagchi

Department of Informatics,
Gyeongsang National University Jinju, South Korea 660 701
susmitbagchi@yahoo.co.uk

Abstract. In general, the modeling and analysis of algorithmic systems involve discrete structural elements. However, the modeling and analysis of recursive algorithmic systems can be done in the form of differential equation following control theoretic approaches. In this paper, the modeling and analysis of generalized algorithmic systems are proposed based on heuristics along with z-domain formulation in order to determine the stability of the systems. The recursive algorithmic systems are analyzed in the form of differential equation for asymptotic analysis. The biplane structure is employed for determining the boundary of the recursions, stability and, oscillatory behaviour. This paper illustrates that biplane structural model can compute the convergence of complex recursive algorithmic systems through periodic perturbation.

Keywords: recursive algorithms, z-domain, stochastic, control theory, perturbation.

1 Introduction

The algorithm design and analysis are the fundamental aspects of any computing systems. The modeling and analysis of algorithms provide an analytical insight along with high-level and precise description of the functionalities of systems [3, 4, 6]. In general, the recursive algorithms are widely employed in many fields including computer-controlled and automated systems [10]. Traditionally, the algorithms are analyzed within the discrete time-domain paying attention to the complexity measures. However, the convergence property and the stability analysis of the algorithms are two important aspects of any algorithmic systems [10]. In case of recursive algorithms, the convergence analysis is often approximated case by case. The asymptotic behaviour of algorithms is difficult to formulate with generalization [4, 10]. The asymptotic behaviour of stochastic recursive algorithms is formulated by constructing models [10], however, such models fail to analyze the stability of the algorithm in continuous time domain throughout the execution. This paper argues that the stability analysis of any algorithm can be performed within the frequency-domain by considering the algorithms as functional building blocks having different configurations. In order to perform generalized frequency-domain analysis, the algorithms are required to be modeled and transformed following the algebraic

K. Mustofa et al. (Eds.): ICT-EurAsia 2013, LNCS 7804, pp. 81–90, 2013.

constructs. Furthermore, this paper proposes that boundary of execution of recursive algorithms can be analyzed following biplane structure and the stability of the algorithms can be observed in the presence of stochastic input by following the traces in the biplane structure bounding the algorithms. The proposed analytical models are generalized without any specific assumptions about the systems and thus, are applicable to wide array of algorithmic systems. This paper illustrates the mechanism to construct analytical model of any complex algorithmic system and methods to analyze the stability of the system under consideration. The rest of the paper is organized as follows. Section 2 describes related work. Section 3 illustrates the modeling and analysis of the algorithms in frequency-domains and their stability analysis using biplane structure. Section 4 and 5 present discussion and conclusion, respectively.

2 Related Work

The modeling of computer systems and algorithms is useful to gain an insight to the designs as well as to analyze the inherent properties of the systems [2, 3, 4, 6, 7, 10]. For example, the fusion of models of artificial neural network (ANN) and fuzzy inference systems (FIS) are employed in many complex computing systems. The individual models of the ANN and FIS are constructed and their interactions are analyzed in order to establish a set of advantages and disadvantages overcoming the complexities of these systems [1]. The other successful applications of modeling techniques to the distributed algorithms and the distributed database in view of Petri Nets are represented in [2, 6]. It is illustrated how Petri Nets can be employed to model and analyze complex distributed computing algorithms [2]. However, in case of distributed database, the concurrency control algorithms are modeled by formulating extended place/transition net (EPTN) [6]. The EPTN formalism is a derivative of the Petri Nets. In structured peer-to-peer (P2P) networks, the random-walks mechanism is used to implement searching of information in minimum time. The model of searching by random-walks in P2P network is constructed to obtain analytical expressions representing performance metrics [3]. Following the model, an equation-based adaptive search in P2P network is presented. The analysis of probabilistic as well as real-time behaviour and the correctness of execution are the challenges of systems involving wireless sensor networks (WSN). Researchers have proposed the modeling techniques of WSN to analyze the behaviour, correctness and performance of WSN by using Real-Time Maude [4]. The Real-Time Maude model provides an expressive tool to perform reachability analysis and the checking of temporal logic in WSN systems. On the other hand, the modeling and analysis of hand-off algorithms for cellular communication network are constructed by employing various modeling formalisms [5, 8]. The modeling of fast hand-off algorithms for microcellular network is derived by using the local averaging method [5]. The performance metrics of the fast hand-off algorithms and the necessary conditions of cellular structures are formulated by using the model construction. In another approach, the modeling technique is employed to evaluate the hand-off algorithms for cellular network [8]. In this case, the model is constructed based on the

estimation of Wrong Decision Probability (WDP) and the hand-off probability [8]. In the image processing systems, the modeling and analysis of signals are performed by designing the sliding window algorithms. Researchers have proposed the Windowed Synchronous Data Flow (WSDF) model to analyze the sliding window algorithms [7]. The WSDF is constructed as a static model and a WSDF-balance equation is derived.

The analysis of convergence of any algorithm is an important phenomenon [9, 10]. The convergence analysis of canonical genetic algorithms is analyzed by using modeling techniques based on homogeneous finite Markov chain [9]. The constructed model illustrates the impossibility of the convergence of canonical genetic algorithms towards global optima. The model is discussed with respect to the schema theorem. On the other hand, the modeling and analysis of generalized stochastic recursive algorithms are performed using heuristics [10]. The heuristic model explains the asymptotic behaviour of stochastic recursive algorithms. However, the model does not perform the stability analysis of the recursive algorithms in the presence of stochastic input.

3 Models of Algorithms in z-domain

The z-domain analysis is widely used to analyze the dynamics and stability of the discrete systems. The computing algorithms can be modeled in z-domain in order to construct heuristic analysis as well as stability analysis of the various algorithmic models in the view of the transfer functions.

3.1 Singular Model

In the singular model, the algorithm is considered as a transfer function with single input and single output (SISO) mechanism. The schematic representation of the singular model is presented in Fig. 1.

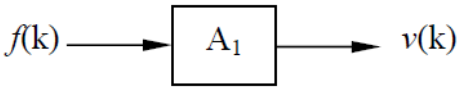

Fig. 1. Schematic representation of singular model

In SISO model, the algorithm A_1 acts as a discrete transfer function for instances k = 0, 1, 2, 3, …..N and transfers the discrete input $f(k)$ into corresponding discrete output $v(k)$. Let, a non-commutative composition of any two functions x and y is described as $(x_o y)$. Thus, the dynamics of the singular algorithmic model can be composed as, $v(k) = A_1(f(k)) = (A_{1o}f)(k)$. Let, $\alpha_1 = (A_{1o}f)$, hence in z-domain $v(z) = \Sigma_{k=0,\infty}\, \alpha_1(k).z^{-k} = \alpha_1(z)$. The algorithmic transfer function is stable if $\alpha_1(z)$ is a monotonically decreasing function for sufficiently large k.

3.2 Chained Model

In the chained model of the algorithmic system, two independent algorithms are put in series as illustrated in Fig. 2.

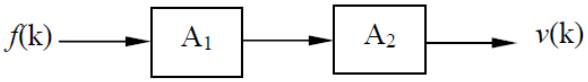

Fig. 2. Schematic representation of chained model

In the chained model, two algorithms act as independent transfer functions transforming discrete input to discrete output at every instant k. Thus, the overall transfer function of chained model can be presented as, $v(k) = (A_{2o}\alpha_1)(k) = \alpha_{21}(k)$. Hence, in the z-domain $v(z) = \alpha_{21}(z)$ and the chained algorithms are stable if $\alpha_{21}(z)$ is monotonically decreasing for sufficiently large k.

3.3 Parallel Models

In case of parallel model, two (or more) independent algorithms execute in parallel on a single set of input at every instant k and, the final output of the system is composed by combining the individual outputs of the algorithms. A 2-algorithms parallel model is illustrated in Fig. 3.

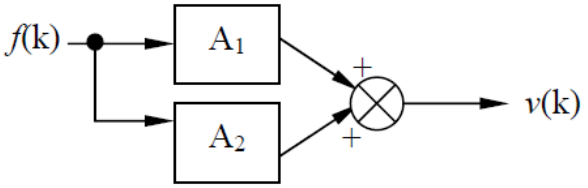

Fig. 3. Schematic representation of 2-algorithms parallel model

So, in the 2-algorithms parallel model, the output is computed as, $v(k) = (A_{1o}f)(k) + (A_{2o}f)(k) = \alpha_1(k) + \alpha_2(k)$. Hence, in the z-domain the discrete values of the output can be presented as, $v(z) = \alpha_1(z) + \alpha_2(z)$. This indicates that a parallel algorithmic system is stable if either the individual algorithmic transfer functions are monotonically decreasing or the combined transfer function of the system is converging for sufficiently large k. On the other hand, the 2-algorithms parallel model can be further extended to parallel-series model by adding another algorithm in series as illustrated in Fig. 4.

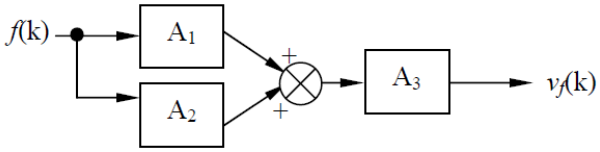

Fig. 4. Schematic representation of 2-algorithms parallel-series model

However, in this case algorithm A_3 transforms the output of parallel computation in deterministic execution at discrete instances k. Hence, the final output of the system is, $v_f(k) = A_3(\alpha_1(k) + \alpha_2(k))$. As, $v(k) = \alpha_1(k) + \alpha_2(k)$, thus $v_f(k) = \alpha_3(k)$, where $\alpha_3 = (A_3 \circ v)$ and $v_f(z) = \alpha_3(z)$. The parallel-series model is stable if $\alpha_3(z)$ is a converging function. This indicates that, $v_f(z)$ can be stable even if $v(z)$ is diverging function provided $A_3(v(z))$ is a monotonically converging function.

3.4 Recursion with Stochastic Observation

The recursive algorithms are widely used in computing systems. The fundamental aspect of any computing system involving the recursive algorithm is the existence of a feedback path as illustrated in Fig. 5. In the feedback algorithmic model, the feedback path is positive and the feedback gain can be either unity or can have any arbitrary transfer-gain.

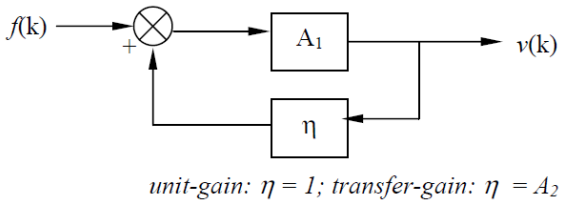

Fig. 5. Schematic representation of recursive model

In pure recursive computing model, the feedback path will have unit gain and the system will be controlled by external input $f(k)$ at $k = 0$, whereas the further input series to the system will change to $v(k)$ due to positive feedback for $k > 0$, where $f(k > 0) = 0$. The behaviour of such system can be analyzed by using two formalisms such as, heuristics and z-domain analysis techniques.

3.4.1 Heuristics Analysis

The generalized difference equation of the recursive algorithmic system is given as, $v(k) = A_1(A_2(v(k\text{-}1)) + f(k))$. In case of positive feedback with unit gain, the closed-loop difference equation controlling the system is given as,

$$v(k) = A_1(v(k\text{-}1) + f(k)) \tag{1}$$

Equation (1) represents a recursive algorithm with stochastic input $f(k)$, where A_1 is a deterministic function having a set of regularity conditions. The function $f(k)$ can be generalized by using some function y as,

$$f(k) = y(v(k-1), f(k-1), e(k)) \tag{2}$$

where, $e(k)$ is an arbitrary error in the system at the execution instant k.

The stability of whole system depends on the stability of equation (2). If $f(k)$ is exponentially stable within a small neighborhood around k after some point b ($k >> b$), then [10],

$$B(v(b-1), f(b)) = h(v(k)) + r(b) \tag{3}$$

where, $B(v(k-1), f(k)) = A_1(v(k-1) + f(k))$, $h(v(\ )) = EB(v, \overline{f}(b))$ and $r(b)$ is a random variable with zero mean. Thus, equation (1) can be represented as,

$$v(k) = B(v(k-1), f(k)) \tag{4}$$

Hence, equation (4) can be approximately evaluated between k and k+a ($a > 0$) as,

$$\begin{aligned} v(k+a) &= v(k) + \Sigma_{j = k+1,\, k+a}\, B(v(j-1), f(j)) \\ &\approx v(k) + \Sigma_{j = k+1,\, k+a}\, h(v(k)) + \Sigma_{j = k+1,\, k+a}\, r(j) \\ &\approx v(k) + a\, h(v(k)) \end{aligned} \tag{5}$$

In equation (5), the random variable is eliminated as it has the zero mean. Hence, the differential equation at point a is given by,

$$lim_{a \to 0}[v(k+a) - v(k)]/a = dv(k)/da = h(v(k)) \tag{6}$$

Thus, the asymptotic properties of the equation (1) can be further derived from equation (6) in the form of derivative for any specific recursive algorithmic system.

3.4.2 Stability in z-domain

For the stability analysis in z-domain, it is assumed that $A_1(k)$ represents the gain factor of A_1 at k-th instant of execution of the algorithm. Now, $v(k) = (\eta(v(k-1)) + f(k))A_1(k)$, where $f(k)$ is a singular external input to A_1 defined as, $f(k) = m$ if $k = 0$ and, $f(k) = 0$ otherwise. Hence, $v(k) = \eta A_1(k)v(k-1) + f(k)A_1(k)$. Initially at $k = 0$, $v(0) = mA_1(0)$. Hence, $v(k) = \eta A_1(k)v(k-1) + mA_1(0)$. If the system is purely recursive, then feedback gain is unity ($\eta = 1$) and, $v(k) = A_1(k)v(k-1) + mA_1(0)$. Thus, in the z-domain the system will be represented as, $v(z) = mA_1(0)z/(z-1) + \Sigma_{k = 2, \infty}\, A_1(k)\, v(k-1)z^{-k}$. Deriving further one can get,

$$\begin{aligned} v(z) &= mA_1(0)z/(z-1) + \{A_1(1)v(0)/z + A_1(2)\, v(1)/z^2 + \ldots\ldots..\} \\ &= mA_1(0)z/(z-1) + mA_1(0)\, \Sigma_{k=1,\infty}\, A_1(k)\, z^{-k} + \\ &\quad \Sigma_{k=2,\infty}\, \{\Pi_{j=k,k-1}\, A_1(j)\}v(k-2)\, z^{-k} \\ &= mA_1(0)z/(z-1) + mA_1(0)[A_1(z) - A_1(0)] + \Lambda_z \end{aligned} \quad (7)$$

where, $\Lambda_z = \Sigma_{k=2,\infty}\, \{\Pi_{j=k,k-1}\, A_1(j)\}v(k-2)\, z^{-k}$.

The system will be stable if Λ_z will minimize or converge for sufficiently large k.

3.5 Functional Properties

The functional properties of a generalized recursive algorithm with unit positive feedback analyze the stability of the overall system in the presence of oscillation, if any. In addition, the concept of biplane symmetry can be used to analyze the bounds of a recursive algorithmic system. The generalized recursive algorithmic model with positive transfer-gain is represented as $v(k) = A_1(A_2(v(k-1)) + f(k))$. Let, $(A_{1o}A_2) = \delta$ and $f(k)$ is a singular external input to algorithm defined as, $f(k) = m$ if $k = 0$ and, $f(k) = 0$ otherwise. Thus, the initial output value is $v(0) = A_1(m)$ and $v(k) = \delta^k(d)$, where $d = A_1(m)$. Now, if A_2 is a unit gain factor, then the system reduces to a pure recursive algorithm such that, $v(k) = A_1^{\,k}(d)$, $k > 0$. The stability and behavioral properties of the recursive algorithmic system can be further analyzed as follows.

3.5.1 Stability and Convergence

Let, $\not{f}: \mathcal{R} \rightarrow \mathcal{R}$ is a stochastic function defined on space $\mathcal{R}$ such that, $\delta(d) \in \not{f}(\mathcal{R}) \subset \mathcal{R}$ and $|\not{f}(\mathcal{R})| > 1$. Now, for $k > 0$, the $\delta^k(d) \in \not{f}^k(\mathcal{R})$ such that, either $\not{f}^k(\mathcal{R}) \cap \not{f}^{k+1}(\mathcal{R}) = \{\phi\}$ or $\not{f}^k(\mathcal{R}) \cap \not{f}^{k+1}(\mathcal{R}) \neq \{\phi\}$ depending on the dynamics. A system is bounded if $\not{f}^{k+1}(\mathcal{R}) \subseteq \not{f}^k(\mathcal{R})$. The boundary of $\delta^k(d)$ is $\Delta_k = \cap_{i=1,k}\, \not{f}^i(\mathcal{R})$. A ε-cut of $\not{f}^k(\mathcal{R})$ is defined as $\not{f}_{k\varepsilon} \subset \not{f}^k(\mathcal{R})$ such that, $\forall a \in \not{f}_{k\varepsilon}$ the following condition is satisfied: $\varepsilon \in \not{f}^k(\mathcal{R})$ and $a > \varepsilon$. An instantaneous remainder of $\not{f}^k(\mathcal{R})$ is given by, $\overline{\not{f}}_{k\varepsilon} = (\not{f}^k(\mathcal{R}) - \not{f}_{k\varepsilon})$. A system is stable at point N if the boundary $\Delta_N \neq \{\phi\}$, where $1 \leq |\Delta_N| \leq w$ and $w << N$. A converging system is a stable system at recursion level N with $|\Delta_N| = 1$.

3.5.2 Divergence in Systems

Let, in a system $\delta^{k-1}(d) \in \not{f}_{(k-1)\varepsilon}$ whereas $\delta^k(d) \in \not{f}_{k\varepsilon}$ and $\delta^{k+1}(d) \in \not{f}_{(k+1)\varepsilon}$ such that, $\delta^{k-1}(d) < \delta^k(d) < \delta^{k+1}(d)$. The system is divergent if $\not{f}_{(k-1)\varepsilon} \cap \not{f}_{k\varepsilon} \cap \not{f}_{(k+1)\varepsilon} = \{\phi\}$. A divergent system is unstable if the limit of recursion $k >> 1$.

3.5.3 Biplane Symmetries

Let, in a system for $k \geq 1$, $\not{f}^k(\mathcal{R}) = \not{f}(\mathcal{R})$ and, $\not{f}^*: \mathcal{R} \rightarrow \mathcal{R}$ such that $(\not{f}^*)^k(\mathcal{R}) = \not{f}^*(\mathcal{R})$ where, $\not{f}(\mathcal{R}) \cap \not{f}^*(\mathcal{R}) = \{\phi\}$. Furthermore, $\not{f}_{*\varepsilon}$ is the ε-cut of $\not{f}^*(\mathcal{R})$ and $\not{f}_\varepsilon$ is the ε-cut of $\not{f}(\mathcal{R})$, whereas the corresponding remainders are $\overline{\not{f}}_{*\varepsilon}$ and $\overline{\not{f}}_\varepsilon$, respectively. Let, $\delta^p(d) \in \not{f}(\mathcal{R})$ for $p = 1, 3, 5, \ldots..$ and, $\delta^q(d) \in \not{f}^*(\mathcal{R})$ for $q = p + 1$. Now, if $x_{p+j} = \delta^{p+j}(d)$ and $y_{q+j} = \delta^{q+j}(d)$, $j = 0, 2, 4, 6\ldots\ldots.$, then following set of predicates can occur in the system,

$P1 \Rightarrow [(x_p \in \overline{\not{f}_\varepsilon}) \wedge (x_{p+2} \in \not{f}_{*\varepsilon}) \wedge (x_{p+4} \in \overline{\not{f}_\varepsilon}) \wedge \ldots\ldots\ldots..]$
$P2 \Rightarrow [(x_p \in \overline{\not{f}_\varepsilon}) \wedge (x_{p+2} \in \not{f}_\varepsilon) \wedge (x_{p+4} \in \overline{\not{f}_\varepsilon}) \wedge \ldots\ldots\ldots\ldots]$
$P3 \Rightarrow [(y_q \in \overline{\not{f}_{*\varepsilon}}) \wedge (y_{q+2} \in \not{f}_{*\varepsilon}) \wedge (y_{q+4} \in \overline{\not{f}_{*\varepsilon}}) \wedge \ldots\ldots\ldots]$
$P4 \Rightarrow (x_{p+j} \in \overline{\not{f}_\varepsilon})$
$P5 \Rightarrow (x_{p+j} \in \not{f}_\varepsilon)$
$P6 \Rightarrow (y_{q+j} \in \overline{\not{f}_{*\varepsilon}})$
$P7 \Rightarrow (y_{q+j} \in \not{f}_{*\varepsilon})$

The possible combinatorial distributions of predicates in a recursive algorithmic system are, P_{13}, P_{23}, P_{46}, P_{47}, P_{56}, P_{57} where, $P_{ab} = (Pa \wedge Pb)$. If distributions P_{46} and P_{57} are valid in a recursive algorithmic system, then it is a biplane-symmetric algorithmic system. Otherwise, if the distributions P_{47} and P_{56} are valid in a system, then the system is a biplane-asymmetric system. Furthermore, if the distribution P_{23} is satisfied in a recursive algorithmic system, then the system is having dual-symmetry between biplanes $\not{f}$ and $\not{f}^*$ and the system is represented as, $[\not{f}/\not{f}^*]$. On the other hand, if the distribution P_{13} is satisfied in a recursive algorithmic system, then the system is called Bounded-Periodic-Perturbed (BPP) system represented as $(\not{f}_{*\varepsilon}|$. In this case, the system is bounded within $\not{f}$ and $\not{f}^*$ planes, however periodic perturbations occur within the domain $\not{f}_{*\varepsilon}$.

3.5.4 Oscillation in Recursive Systems

In a biplane-symmetric system if the following properties hold, then it is called the biplane-symmetric oscillatory recursive system, $\forall p, q, |x_p| = |y_q| = |x_{p+j}| = |y_{q+j}|$ and $x_p + y_q = x_{p+j} + y_{q+j} = 0$. However, in a $[\not{f}/\not{f}^*]$ system if the following conditions hold, then the system is called asymmetrically oscillating between $\not{f}$ and $\not{f}^*$ planes for values of s (s = 0, 4, 6……), $\forall p, q, |x_{p+s}| = |y_{q+s}|, |x_{p+s+2}| = |y_{q+s+2}|$ and $x_{p+s} + y_{q+s} = 0$, $x_{p+s+2} + y_{q+s+2} = 0$. If a recursive algorithmic system is oscillatory, then it is a deterministic but non-converging system.

A recursive algorithmic system is deterministic and converging if there exists a constant *C* such that, $\sum_{p = 1, N} (x_p + y_{p+1}) = \sum_{p = 1, M} (x_p + y_{p+1}) = C$, where $N \neq M$. This indicates that, a deterministic and converging recursive algorithmic system should be in damped oscillation (stable) and should contain idempotency. On the other hand, an oscillatory non-converging recursive algorithmic system is non-idempotent requiring strict consistency conditions.

4 Discussion

Traditionally, the recursive algorithmic systems are analyzed by using heuristics as well as asymptotic methods following difference equation. Often, the differential equation is formulated in place of difference equation in order to conduct analysis in continuous plane avoiding the complexity. However, the generalized z-domain analysis of algorithmic systems in a discrete plane offers an insight towards the understanding of the overall stability of the algorithmic systems.

The perturbation analysis of a system using biplane structure captures the inherent oscillation in the system. The determinism of convergence of the recursive algorithmic systems with stochastic input can be computed using the symmetry of biplane structure. As a result, the idempotent property of the recursive algorithmic systems becomes easily verifiable in case of complex systems. Thus, depending upon the idempotent property of the complex recursive algorithmic systems, the appropriate consistency conditions can be designed.

5 Conclusion

The analysis of stability and behaviour of any algorithmic systems can be accomplished by modeling such systems as a block having transfer functional properties. The z-domain analysis of algorithmic models captures the overall response trajectory and the stability of the algorithmic systems. The complex recursive algorithmic systems can be analyzed by modeling in view of z-domain and biplane structure. The heuristics and z-domain models of a generalized recursive algorithmic system with stochastic input reduce the overall system to the deferential equation presenting the dynamic behaviour of the recursive algorithm. On the other hand, the biplane structure determines the boundaries of the recursive algorithmic systems. In addition, the biplane structural model of recursive algorithmic systems serves as a tool to analyze the oscillatory nature of the recursions as well as the stability of the algorithmic systems. The biplane structural model helps to achieve periodic perturbation into the system dynamics and determining convergence conditions, which enables to design the appropriate consistency conditions.

References

1. Abraham, A.: Neuro Fuzzy Systems: Sate-of-the-Art Modeling Techniques. In: Mira, J., Prieto, A.G. (eds.) IWANN 2001. LNCS, vol. 2084, pp. 269–276. Springer, Heidelberg (2001)
2. Reisig, W.: Elements of distributed algorithms: modeling and analysis with petri nets. Springer (1998)
3. Bisnik, N., Abouzeid, A.: Modeling and analysis of random walk search algorithms in P2P networks. In: Second International Workshop on Hot Topics in Peer-to-Peer Systems (HOT-P2P 2005), pp. 95–103. IEEE (2005)
4. Olveczky, P.C., Thorvaldsen, S.: Formal modeling and analysis of wireless sensor network algorithms in Real-Time Maude. In: 20th International Symposium on Parallel and Distributed Processing (IPDPS 2006). IEEE (2006)
5. Leu, A.E., Mark, B.L.: Modeling and analysis of fast handoff algorithms for microcellular networks. In: 10th IEEE International Symposium on Modeling, Analysis and Simulation of Computer and Telecommunications Systems (MASCOTS 2002). IEEE (2002)
6. Ozsu, M.T.: Modeling and Analysis of Distributed Database Concurrency Control Algorithms Using an Extended Petri Net Formalism. IEEE Transactions on Software Engineering SE-11(10) (1985)

7. Keinert, J., Haubelt, C., Teich, J.: Modeling and Analysis of Windowed Synchronous Algorithms. In: 2006 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP 2006). IEEE (2006)
8. Chi, C., Cai, X., Hao, R., Liu, F.: Modeling and Analysis of Handover Algorithms, Global Telecommunications Conference (GLOBECOM 2007). IEEE (2007)
9. Rudolph, G.: Convergence analysis of canonical genetic algorithms. IEEE Transactions on Neural Networks 5(1) (1994)
10. Ljung, L.: Analysis of recursive stochastic algorithms. IEEE Transactions on Automatic Control 22(4) (1977)