2011 Eighth International Conference on Information Technology: New Generations

Flow Decomposition in Complex Systems

David Luper
Department of Computer Science
University of Georgia
Athens, GA, USA
luper.david@gmail.com

John Schramski
Faculty of Engineering
University of Georgia
Athens, GA, USA
jschrams@uga.edu

Caner Kazanci
Department of Mathematics
University of Georgia
Athens, GA, USA
caner@uga.edu

Hamid R. Arabnia
Department of Computer Science
University of Georgia
Athens, GA, USA
hra@cs.uga.edu

***Abstract*—Complex systems can be represented as weighted digraphs. Cycles play an important role in complex systems because they define relationships consisting of unique groupings of nodes. A grouping of connected nodes contains rich contextual meaning because of the relationships defined by its connecting edges. Cycle bases are a description of the set of all independent cycles within a graph. The work herein outlines a computational methodology to decompose the total throughflow of a complex system into a set of coefficients over its cycle bases. A coefficient is computed for each cycle representing the cycle's contribution to the total system throughflow. This vector of coefficients provides information for data mining and information clustering applications to analyze the system. The proposed methodology provides a powerful framework for analyzing symbolic data by assigning magnitude values to the contextual meaning within groupings of symbols.**

Data Mining; Graph Mining; Information Clustering; Network Analysis; Sequence Mining; Systems Analysis

I. INTRODUCTION

Complex systems can be modeled as a set of interconnected compartments with directed, weighted edges (currency flows; e.g., mass, energy, dollars, information). This type of graph construct appears across a wide variety of scientific disciplines [3][4][5][6][7][8] including economics, computer science, ecology, biology and sociology. Models help one understand systems that are too complex for deterministic behavior to be recognized, such as a person's movement (i.e. tracking a person's GPS data)[18][19], a food web in an ecosystem [17], rhythm patterns within music [20] or financial volatility within economic systems [21]. Systems analysis historically involves the evaluation of graph structure and function through the calculation of such metrics as nodal connectedness or compartment and total throughflows. This methodology can be helpful, but lacks the ability to analyze interaction between groupings of connected nodes. We consider a more complete method of analysis, which includes a node's sphere of influence within node groupings defined by cycles. This approach can serve to contextualize the behavior of a specific node and provides a more global, complete understanding of its role within the entire set of nodes and their connectivity. The goal of this research is to quantify a measure of flow over these cycle related groupings of interrelated nodes in a graph. This goal requires both structural and functional decomposition of graphs. Kavitha et al. [14] summarize the methodology to structurally decompose a graph to obtain its set of cycles. Graph function refers to the flow along edges within a graph. This work analyzes a distribution of pathways through a graph with a context free grammar to obtain functional decomposition. Interpreting a graph pathway in this manner allows each time step t in a pathway to be labeled as part of a particular cycle, and a histogram can be built to represent the number of times each cycle has appeared in a distribution of pathways. This represents the flow magnitude for each cycle, or the total throughflow in the graph portioned out to each of its structural components. The functional decomposition is the main focus of the work herein.

The structure of this paper is a follows. First, basic concepts in graph theory and regular languages will be reviewed. Then the data model and computational techniques for deriving the flow magnitude values will be discussed. Finally, the limitations of this methodology as well as some preliminary results will be presented with discussion.

II. Graph Theory

Complex systems can be represented as directed weighted graphs. Graph theory is a well-established field, and representing a system this way affords many rich constructs and methodologies. The following definitions are taken from Corman et al. [1].

"A directed graph G is a pair (V, E) where V is a set of nodes or vertices and E is a set of binary relations on V called edges. A graph is connected if a path exists from every node to every other node, and a directed graph is said to be weakly connected if replacing directed edges with undirected edges yields a connected graph. An adjacency matrix representation of graph G(V,E) is a |V| x |V| matrix A = (a_{ij}) such that a_{ij} = 1 if (i, j) is an edge in G and a_{ij} = 0 otherwise. A path is a sequence of vertices or nodes of length k $< v_0, v_1, \dots, v_k >$ originating at u and ending at u' such that u = v_0 and u' = v_k and (v_{i-1}, v_i) ∈ E for i = 1, 2, 3, . . ., k. A simple path is a path where each vertex is unique. A cycle is a path where $v_0 = v_k$."

III. Context Free Grammars

Context free grammars are a collection of substitution rules governing the definition of a regular language Sipser [13]. These rules allow grammatical units (groupings of literals) to be nested arbitrarily deep but not to overlap. Formally defined, a context free grammar is a 4 tuple G = (V, ∑, R, S) where V is a finite set of variables, ∑ is a finite set of terminal symbols, R is a list of substitution rules, and S is a start symbol. Starting at S, substitution rules are applied to produce a parse tree where the leaves of the tree are terminal symbols and the internal nodes of the tree are variables. A parse tree holds the information on which grammatical units comprise a particular string. There can be multiple parse trees for a given string due to ambiguity in the grammar. In certain grammars it is not possible to eliminate ambiguity, and these grammars are labeled inherently ambiguous. When arranging a context free grammar it can be necessary to formulate substitution rules in a simplified way called Chomsky normal form (CNF) [13]. A grammar is in CNF if every substitution rule is either of the form A → B C or A → a, where A, B, and C are any variables, except B and C are not the start state, and a is any terminal.

IV. Decomposition of Complex Systems

In the most general sense, a system is a grouping of interconnected components that forms a whole entity. Fig. 1 depicts an example of an ecological directed graph compartmental system created to model an oyster reef habitat. A complex system is more a notion of intuition rather than definable characteristics due to the fact that there is no agreed upon definition of system complexity. A complex system is typically comprised of building blocks (e.g. atoms, molecules, cells, etc. more generally denoted agents), which can be complex systems themselves (giving rise to hierarchical levels of interaction), that work with each other to exude actions, traits, behaviors and organization not observable from studying the building blocks in isolation [16]. Interaction between compartments is typically illustrated through edges in a weighted di-graph. Extending these relationships beyond a single edge defines relationships with greater contextual meaning. The set of all cycles in the system exhaustively defines a set of unique, self-sustaining relationships. As a unit of flow moves through a system, the pathway it follows can be described in terms of these defined relationships. This valuable source of system information tracks the frequency of occurrence for these unique relationships.

Magnitude values (i.e. frequency of occurrence) for each cycle in a graph are calculated using a data simulation technique called Network Particle Tracking (NPT) [9], which simulates network activity over a specified range of time in the network's life. The data NPT creates is a list of particle pathways traversed through a system. Each of these pathways are treated as a string generated from a context free grammar defined by the set of cycles within the system. Using this grammar each unique cycle within the system is a grammatical unit

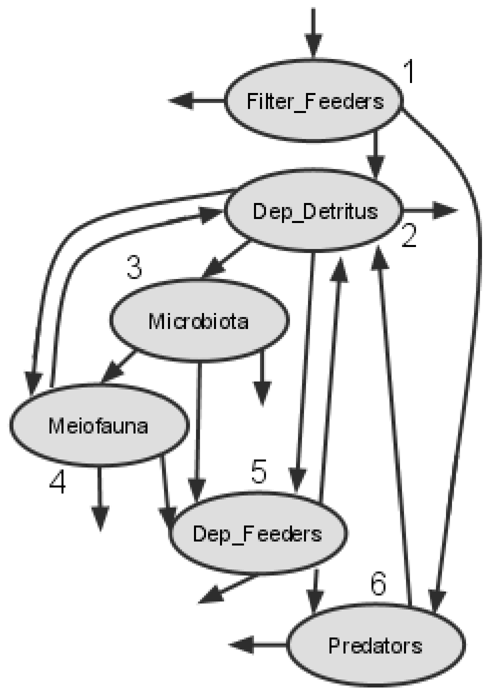

Figure 1. An ecosystem model depicting an oyster reef habitat.

within the context free language of the decomposed directed graph. Each particle pathway in the data distribution is parsed into the grammatical units from which it is constructed. A histogram is kept representing each time the different cycles have been used. The resulting histogram defines the magnitude values.

Constructing a context free grammar from the set of cycles in a graph is central to the proposed methodology. A context free grammar is a 4 tuple, as defined earlier, G = (V, ∑, R, S). The variables (V) and the terminal symbols (∑) for a grammar in the proposed methodology are all compartments in the graph. It is important to note that in systems research there is a concept of an environment that inputs into the system and receives output from the system. The environment can be added to any graph, and it is treated as a special node in the graph where pathways must start and end. The substitution rules for a grammar (R) are obtained from the cycles in a graph. Two distinct classes of cycles exist in a grammar, those that contain the environment and those that do not. The cycles that do not contain the environment represent internal cycling loops in the system. Any pathway through the system can be interpreted using exactly one cycle containing the environment (as paths must start and stop there) and one or more internal cycling loops. Any variable in the grammar can be replaced by any grammatical unit where that variable is the first element in the unit, or by its corresponding literal. An example of rule derivation is depicted in Fig. 2.

Each cycle not containing the environment represents multiple grammatical units because there are multiple orderings the compartments of the cycle can appear. For example if cycle A consists of three compartments 1, 2 and 3, this cycle could appear in the orderings 1-2-3, 3-1-2 and 2-3-1. Internal cycles represent the number of grammatical units equal to the length of the cycle. These different grammatical units are made by assigning a new index to each node in the cycle equal to its current index plus one, mod the length of the cycle. This is applied to find every possible ordering of cycle compartments for every internal cycle. The pseudo code for this can be seen in Fig. 3. Each cycle containing the environment represents a single grammatical unit because the environment is the starting and stopping point for each pathway. This determines the order these cycles must appear. Finally, the environment is designated the start symbol (S).

Once the grammar is identified, a graph pathway can be parsed into the grammatical units (i.e. cycles) it traversed. This process can be understood as interpreting a pathway through the graph. Fig.4 illustrates an interpreted pathway through the system depicted in Fig. 1.

To compute a parse tree, the Cocke-Younger-Kasami (CYK) algorithm is used [22][23][24]. The CYK dynamic programming algorithm considers every possible subsequence of the sequence of words presented in a string and compares these subsequences against the grammar to see if they are interpretable. The results are stored in a table that can be traced back through to construct all parse trees associated with the input string. The run time for this algorithm is $O(n^3 * |G|)$ where n is the length of the input string and |G| is the size of the grammar [24]. Grammars presented to this algroithm need to be structured in the more simplified CNF [23]. As mentioned earlier, a grammar is said to be in CNF if every substitution rule is either of the form A → B C or A → a, where A, B and C are any variables, except B and C are not the start state and a is any terminal. Any context free grammar can be transformed into a CNF grammar expressing the same language [13]. This transformation can lead to considerable bloat in the size of the grammar. In the worst case this can increase grammar size from g^2 to 2^{2g}. For implementation of the proposed methodology each rule in the original grammar requires a number of rules in the CNF grammar equal to one minus the length of the right hand side of the rule. The transformation is achieved by introducing new symbols into the grammar. A rule in the original context free grammar representing a cycle ABC

Variable:	X
Grammatical Unit:	X Y Z
Substitution Rule:	X → X Y Z X

Figure 2. Example of a rule substitution where a cycle is substituted for a variable.

```
foreach internal cycle:
    for i in range(1,len(cycle)):
        new cycle[len(cycle)];
        for j in range(1,len(cycle)+1):
            x = j%len(cycle);
            new cycle[x] = cycle[j–1];
        cycle = new cycle;
```

Figure 3. Pseudo code for obtaining all possible grammar elements from an internal cycling loop in the decomposed set of cycles.

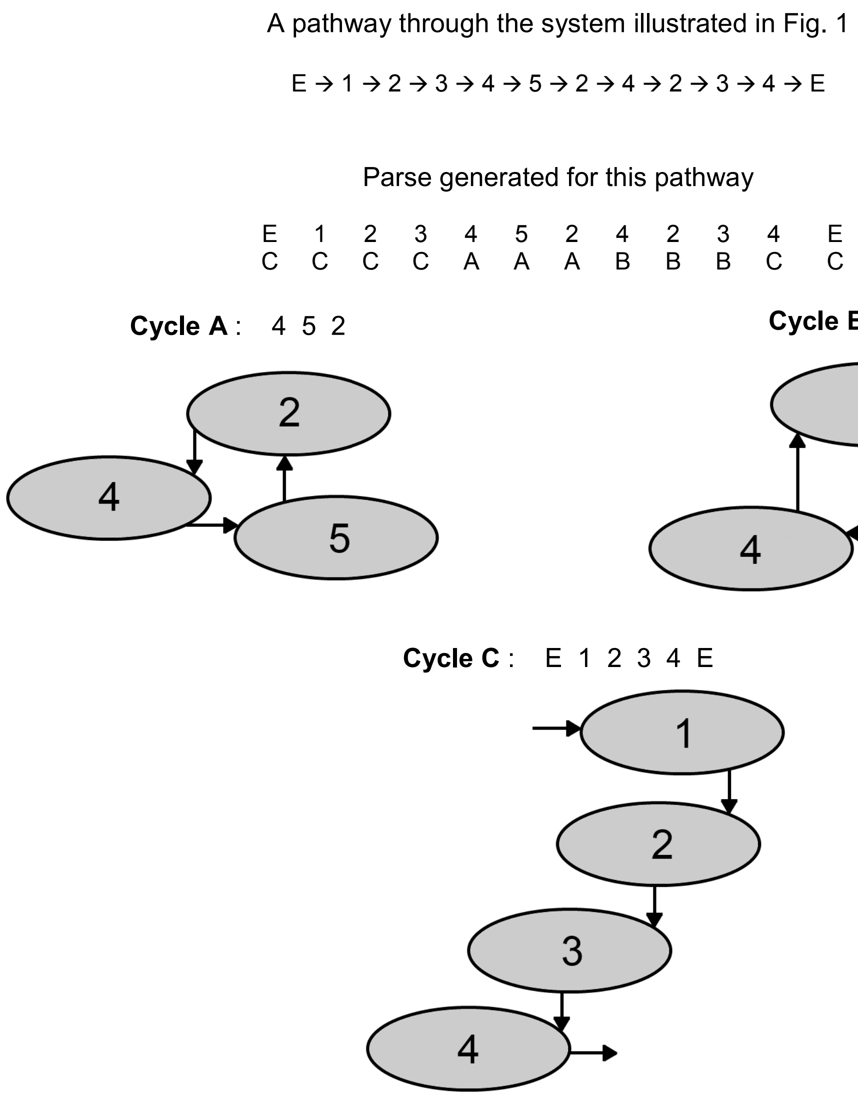
A pathway through the system illustrated in Fig. 1
E → 1 → 2 → 3 → 4 → 5 → 2 → 4 → 2 → 3 → 4 → E
Parse generated for this pathway
E 1 2 3 4 5 2 4 2 3 4 E
C C C C A A A B B B C C
Cycle A : 4 5 2
Cycle B : 4 2 3
Cycle C : E 1 2 3 4 E
2
4
5
2
3
4
1
2
3
4

Figure 4. An example of an interpreted pathway through the ecosystem model in Fig. 1. The pathway is comprised of two internal cycles (A and B) and one cycle containing the environment (C). Each time step in the pathway is assigned to a specific cycle as depicted in the parse layout.

would appear as A → ABCA. This rule is transformed into three rules A → AX, X → BY, Y →CA representing the same cycle where X and Y are symbols not previously in the grammar. The number of rules to be expected would be equivalent to $\sum_{i=0}^{n}\sum_{j=0}^{m} len(cycle[i][j]) - 1$ where n is the number of cycles in the set of cycles over the graph and m is the number of orderings for that cycle. After the transformation, the CYK algorithm is used to analyze a distribution of pathways simulated by NPT. The distribution of pathways can be of arbitrary size, but the accuracy of the magnitude values increase as the number of analyzed pathways increase. To calculate the magnitude values a histogram is maintained over the entire distribution of data tracking the occurrences of each cycle.

V. Discussion

The histogram calculated by the proposed methodology represents the occurrence frequency (i.e. magnitude) of a unit of flow completely passing through each cycle. The frequency of occurrence could be seen as a method for ranking importance within a complex system, but other uses for these magnitude coefficients exist. This methodology takes symbolic sequences of data produced from a model and computes numeric magnitude coefficients reflecting position within an n dimensional feature space. This can be seen as a symbolic transformation routine that would provide valuable results for machine learning and computational intelligence methodologies. These histograms would allow insight into the position of a model within a feature space, conveying what state a model was in when the data being analyzed was produced. Mapping models into a feature space would allow distance between structurally similar models to be calculated. Furthermore, it would be possible to preform comparative analysis on specific groupings of coefficients between two models giving potential for such things as finding a subset of cycles which account for a maximal flow difference between two models.

Ambiguity in a particular grammar means that certain input strings can be generated by multiple parse trees. It is possible to interpret some pathways through graphs using different sets of cycles (i.e. some sets of cycles, when added together generate the same sub-graph). Two different methods exist to deal with multiple interpretations. One alternative is to find all interpretations of a pathway and weight each addition to the histogram by the total number of interpretations. The resulting histogram utilizes all interpretations and reflects their combined occurrence by assigning them a weighting parameter. This results in a histogram of averaged values as they occurred in simulation. The second alternative uses one of the interpretations. This decreases the runtime of the algorithm as the trace-back over the parse trees need only be completed for one parse tree, not exhaustively for all parse trees. In order for this alternative to be useful, the ignored interpretations must be addressed. This is accomplished through a difference vector matrix. For example, when an ambiguity occurs in the grammar some grouping of cycles can be replaced with a different grouping of cycles to interpret the same set of literals. The difference vector matrix stores every linearly independent ambiguity in the grammar as an equality constraint. These equality constraints (together with the inequality constraint that every element in the histogram has to be greater than 0) bound an n dimensional hyper plane in the m dimensional feature space where $n < m$. This n dimensional hyper plane bounds all of the valid histograms from the analyzed graph. The difference vector matrix is defined by the structure of a graph and not by the flow values. This means that two structurally equivalent graphs would share the same set of equality constraints regardless of the flow values over their respective edges. The current method used to find the difference vector matrix is an exhaustive search over all possible combinations of cycles.

The proposed methodology has runtime and memory constraints. The runtime is bounded by the runtime of the CYK algorithm used for interpreting the cycles in a given pathway. This runtime is $O(n^3 * |G|)$ where n is the length of the input string and $|G|$ is the size of the grammar. Therefore, the most significant factor affecting this runtime is the length of the input string. This means that graphs producing shorter pathways require less computation. As the internal cycling within a graph increases the runtime increases exponentially. The size of the grammar also affects the runtime to a lesser degree, and the size of the grammar depends on both the size and connectedness of the graph. Methods of addressing these memory and runtime constraints are currently being researched. Strategic graph contraction and distributed computing models are being researched to overcome these constraints for large, heavily connected graphs with lots of internal cycling.

This methodology was used on three ecological network models together representing varying sizes and complexities including the oyster reef [25], the Georgia salt marsh [26] and the Neuse river basin model [27]. Magnitude coefficients for each of the cycles in both the oyster reef and the Georgia salt marsh models were computed. The Neuse river model posed a greater challenge due to the size and connectedness of the model in combination with its heavy proclivity for internal cycling. Pathways for this model routinely had lengths of greater than 40 nodes with the occasional pathway reaching lengths in the one and two hundreds of nodes. Memory issues were the main problem with this model as the current implementation of this methodology uses RAM and not disk memory while computing the CYK algorithm. If disk memory were to be used the memory constraint could be overcome which would leave a very substantial runtime to deal with. As mentioned before research is being directed into dealing with these problems.

VI. Conclusion

The work herein proposes a methodology for complex systems analysis using graph theory and context free grammars to produce magnitude coefficients for compartment groupings within system models. The methodology uses a structurally decomposed, directed, weighted graph (i.e. the set of all cycles in the graph) to build a context free grammar that allows for the interpretation of any pathway through the graph. The context free grammar is used in conjunction with the CYK algorithm to interpret a distribution of pathways simulated over a graph as combinations of individual cycles. The frequency of occurrence of each cycle over the distribution of pathways is calculated and presented as a histogram of magnitude coefficients. The histogram holds a wealth of information about the state of a system, and it maps the symbolic distribution of sequences produced from the graph into a numerical feature space. Uses for the histogram range from information clustering to machine learning or any number of methods that rely on numerical feature spaces. Ambiguity within the context free grammars produced by this methodology was discussed as well as ways this ambiguity can be dealt with to maintain the integrity of the results. The ambiguity can be potentially ignored using an averaging technique when interpreting pathways, or it can be defined by a set of equality and inequality constraints over the feature space the histogram is mapped into. More research needs to be directed towards finding all of these constraints as currently an exhaustive search over all possible combinations of cycles is needed.

This methodology can analyze throughflow in a complex system as it relates to nodal groupings within the system. This allows for nodal throughflow analysis with greater contextual nuance.

REFERENCES

[1] Thomas Cormen, Thomas, Charles Leiserson, and Ronald Rivest. 2001. *Introduction to Algorithms.* Second Edition, Cambridge, MA: MIT Press.

[2] R. Jiang, Z. Tu, T. Chen, and F. Sun, "Network motif identification in stochastic networks," *Proc. Nat. Acad. Sci. U.S.A.*, vol. 103, no. 25, pp. 9404–9409, Jun. 2006.

[3] P. Ammann, D. Wijesekera, S. Kaushik, "Scalable, Graph-Based Network Vulnerability Analysis," in *Proceedings of CCS 2002: 9th ACM Conference on Computer and Communications Security*, Washington, DC, November 2002.

[4] Kazanci, C., EcoNet: A new software for ecological modeling, simulation and network analysis, Ecol. Model., vol. 208, no. 1, pp. 3--8, 2007.

[5] Batagelj, V. and Mrvar, A. (1998). Pajek { Program for large network analysis. Connections, 21(2):47{57. Project home page at http://vlado.fmf.uni-lj.si/pub/networks/pajek/.

[6] Smith, David A., and Douglas R. White. 1992. "Structure and Dynamics of the Global Economy: Network Analysis of International Trade, 1965-1980." *Social Forces 70:857-93.*

[7] Wellman B, Salaff J, Dimitrova D, Garton L, Gulia M, Haythronwaite C. 1996. Computer networks as social networks: collaborativework, telework and virtual community. *Annu. Rev. Sociol.* 22:213–38

[8] Thibert B, Bredesen DE, del Rio G. Improved prediction of critical residues for protein function based on network and phylogenetic analyses. BMC Bioinformatics 2005;6:213.

[9] Kazanci, C. and Matamba, L. and Tollner, E. W., Cycling in ecosystems: An individual based approach, Ecol. Model., vol. 220, no. 21, pp. 2908--2914, 2009.

[10] James K. Dame, 2007, Evaluation of ecological network analysis: Validation of output, Ecological Modelling; Vol. 210 Issue 3, p327-338, 12p

[11] Tollner, E. W. and Schramski, J. R. and Kazanci, C. and Patten, B. C., Implications of network particle tracking (NPT) for ecological model interpretation, Ecological Modelling, vol. 220, no. 16, pp. 1904--1912, 2009.

[12] Brand,M.: Fast online svd revisions for lightweight recommender systems. Proceedings of the 3rd SIAM International Conference on Data Mining. (2003)

[13] M. Sipser. *Introduction to the Theory of Computation*. PWS Publishing, 1997.

[14] T. Kavitha, C. Liebchen, K. Mehlhorn, D. rios Michail, R. Rizzi, Cycle bases in graphs: Characterization, algorithms, complexity, and applications, Computer Science Review 3 (2009), 199–243.

[15] P. Hingston, Using Finite State Automata for Sequence Mining, In: Proceedings of the 25th Australasian Computer Science Conference, Melbourne, Australia, 105 – 110, 2001.

[16] Deisboeck, T. S. and Kresh, J. Y., Complex systems science in biomedicine. New York, Springer, 2006.

[17] Rudolf Philippe Rohr, Heike Scherer, Patrik Kehrli, Christian Mazza and Louis-Félix Bersie, Modeling Food Webs: Exploring Unexplained Structure Using Latent Traits, The American Naturalist, Vol. 176, No. 2, August 2010, (pp. 170-177

[18] David Luper, Muthukumaran Chandrasekaran, Khaled Rasheed, and Hamid R. Arabnia, Path Normalcy Analysis Using Nearest Neighbor Outlier Detection, ICAI 2008, 2008, p. 776-783, Proc. of International Conference on Information and Knowledge Engineering (IKE'08; ISBN #: 1-60132-075-2), Las Vegas, USA, July 14-17, 2008 pp. 776-783.

[19] David Luper, Ron McClendon, and Hamid R. Arabnia, Positional Forecasting From Logged Training Data Using Probabilistic Neural Networks, Proc. of International Conference on Information and Knowledge Engineering (IKE'09; ISBN # for set: 1-60132-116-3), Las Vegas, USA, July13-26, 2009, pp. 179-189.

[20] David Temperley, Modeling Common - Practice Rhythm, Music Perception: An Interdisciplinary Journal, Vol. 27, No.5, June 2010, p. 355 – 376

[21] Marcelo C. Medeiros and Alvaro Veiga, Modeling Multiple Regimes in Financial Volatility with a Flexible Coefficient GARCH(1, 1) Model, Econometric Theory, Vol. 25, No. 1, Feb. 2009, p 117 – 161

[22] John Cocke, Jacob T. Schwartz (1970). Programming languages and their compilers: Preliminary notes. Technical report, Courant Institute of Mathematical Sciences, New York University.

[23] Tadao Kasami (1965). An efficient recognition and syntax-analysis algorithm for context-free languages. Scientific report AFCRL-65-758, Air Force Cambridge Research Lab, Bedford, MA.

[24] Danial H. Younger (1976). Recognition and parsing of context-free languages in time n^3. Information and Control 10(2): 189-208.

[25] Dame R.F. & Patten B.C. (1981) Analysis of energy flow in an intertidal oyster reef. Marine Ecology Progress Seires, 5, 115-124

[26] Teal, J.M. 1962. Energy flow in the salt marsh system of Georgia. Ecology 43: 614-624

[27] Schramski J.R., Gattie, D.K., Patten, B.C., Borret, S.R., Fath, B.D., (2006). Indirect effects and distributed control in ecosystems: distributed control in the environ networks of a seven-compartment model of nitrogen flow in the Neuse River Estuary. USA-steady-state analysis. Ecol. Medel. 194, 189-201.