A Simple Real-Coded Extended Compact Genetic Algorithm

Luca Fossati Pier Luca Lanzi Kumara Sastry David E. Goldberg Osvaldo Gomez

***Abstract*— This paper presents a simple real-coded estimation of distribution algorithm (EDA) design using χ-ary extended compact genetic algorithm (χECGA) and discretization methods. Specifically, the real-valued decision variables are mapped to discrete symbols of user-specified cardinality using discretization methods. The χECGA is then used to build the probabilistic model and to sample a new population based on the probabilistic model. The effect of alphabet cardinality and the selection pressure on the scalability of the real-coded ECGA (rECGA) method is investigated. The results show that the population size required by rECGA—to successfully solve a class of additively-separable problems—scales sub-quadratically with problem size and the number of function evaluations scales sub-cubically with problem size. The proposed rECGA is simple, making it amenable for further empirical and theoretical analysis. Moreover, the probabilistic models built in the proposed real-coded ECGA are readily interpretable and can be easily visualized. The proposed algorithm and the results presented in this paper are first step towards conducting a systematic analysis of real-coded EDAs and towards developing a design theory for development of scalable and robust real-coded EDAs.**

I. Introduction

Over the last two decades significant progress has been made in the design and the development of scalable genetic algorithms that solve hard problems quickly, reliably, and accurately [1], [2]. One such class of scalable genetic algorithms are the so called estimation of distribution algorithms (EDAs) [3], [4], [5], [6], which replace the traditional variation operators of genetic algorithms by building and sampling probabilistic models. EDAs have successfully solved boundedly-difficult single-level and hierarchal problems, oftentimes requiring only sub-quadratic number of function evaluations [4]. Despite their demonstrated scalability, most EDAs operate on binary variables, and their success has not been extensively carried over to other encodings such as permutation, program and real-codes. Moreover, existing real-coded EDAs are fairly complex, rendering them intractable for theoretical, analytical, or even semi-empirical analysis.

Therefore, the purpose of this paper is to develop a simple real-coded estimation of distribution algorithm that can be used to investigate—both theoretically and empirically—the strengths and weaknesses of different design decisions. We extend one of the simpler binary-coded EDAs called extended compact genetic algorithm (ECGA) [7]. Specifically, we map the real-coded variables into symbols of user-specified cardinalities using discretization methods. Then we use χECGA [8] to build and sample the models. The sampled population of discrete symbols is mapped back into real-valued decision variables. Unlike other real-coded EDAs, with the proposed real-coded ECGA, the probabilistic model—which is a partition of the decision variables into non-overlapping clusters—can be easily visualized and analyzed. Moreover, similar to the binary ECGA the probability model yields a direct mapping of linkage groups and also readily identifies key variable interactions. In this paper, we investigate the effects of alphabet cardinality and selection pressure on the scalability of the proposed real-coded ECGA. The proposed work is a first step towards systematically analyzing the strengths and weaknesses of existing different design decisions that can potentially lead to a design theory for the development of successful real-coded EDAs.

This paper is organized as follows. In the following section, we briefly discus related real-coded EDA designs. Section III gives an outline of the extended compact genetic algorithm followed by implementation details of the proposed real-coded ECGA in section IV. The population-sizing and scalability results are discussed in section V followed by summary and conclusions.

II. Related Work

We now provide a brief overview of the works that are relevant to this paper; for more references, we refer the reader to the two recent books [5], [6]. In [9], Pelikan et al. introduced a real-coded PMBGA working in the continuous domain; they used marginal histograms to model promising solutions. Both the marginal models used, fixed width and fixed height histograms, performed fairly well on test functions with no or weak interaction among the variables, but failed to recognize linkage among variables in functions with a medium level of linkage. An evolution of this approach is presented in [10] where the focus is on the linkage identification in real-coded GAs. In [10], the authors applied the SPX operator [11] as recombination operator and tried to identify linkage information by observing the distribution of the individuals in the population; in particular they examined the correlation coefficient matrix of parameter values of the individuals in the population.

Pelikan et al. [12], [13] combined the Bayesian Optimization Algorithm for recombination with evolutionary strategies (ES) for mutation. In [12], [13], a real population is first discretized, then BOA is applied to recombine individuals in the discrete population and the new population is mapped back to the real domain so that adaptive mutation from ES can be applied to obtain the next real population. Three discretization strategies were presented: Fixed-Width Histograms,

Luca Fossati is with the Dipartimento di Elettronica e Informazione, Politecnico di Milano, (email: fossati@elet.polimi.it). Pier Luca Lanzi is with the Illinois Genetic Algorithm Laboratory at University of Illinois at Urbana-Champaign, Urbana, Illinois, USA (email: lanzi@illigal.ge.uiuc.edu). He is also with the Dipartimento di Elettronica e Informazione, Politecnico di Milano, (email: pierluca.lanzi@polimi.it). Kumara Sastry is with the University of Illinois at Urbana-Champaign, Urbana, Illinois, USA (email: kumara@illigal.ge.uiuc.edu). David E. Goldberg is with the University of Illinois at Urbana-Champaign, Urbana, Illinois, USA (email: deg@uiuc.edu).

1-4244-1340-0/07/$25.00 ©2007 IEEE

Fixed-Height Histograms and k-Means Clustering. The approach was tested on three functions and the reported results showed good scalability. Ahn [14] developed a real-coded Bayesian Optimization Algorithm, rBOA, which constructs the Bayesian factorization graph using finite mixture models. All the relevant substructures are extracted from the graph and each substructure is fit and sampled independently.

Recently, Chen et al. [15] developed a real-valued version of the ECGA (rECGA) by combining a binary version of ECGA together with Split-On-Demand (SoD) discretization strategies. This is an adaptive discretization technique that takes into account the distribution of the current population in creating discrete intervals. rECGA was tested on several problems and the accuracy of the computed solutions was shown.

III. Extended Compact Genetic Algorithm (eCGA)

The extended compact genetic algorithm (ECGA) [16], [17] is an estimation of distribution algorithm (EDA) that replaces traditional variation operators of genetic and evolutionary algorithms by building a probabilistic model of promising solutions and sampling the model to generate new candidate solutions. χ-ary ECGA is an extension of the ECGA to handle integer variables with differing cardinalities [18], [19]. The typical steps of χECGA can be outlined as follows:

1) Initialization: The population is usually initialized with random individuals. However, other initialization procedures can also be used in a straightforward manner.
2) Evaluation: The fitness or the quality-measure of the individuals are computed.
3) Selection: Like traditional genetic algorithms, EDAs are selectionist schemes, because only a subset of better individuals is permitted to influence the subsequent generation of candidate solutions. Different selection schemes used elsewhere in genetic and evolutionary algorithms—tournament selection, truncation selection, proportionate selection, etc.—may be adopted for this purpose, but a key idea is that a "survival-of-the-fittest" mechanism is used to bias the generation of new individuals.
4) Probabilistic model estimation: Unlike traditional GAs, however, EDAs assume a particular probabilistic model of the data, or a class of allowable models. A class-selection metric and a class-search mechanism is used to search for an optimum probabilistic model that represents the selected individuals.

Model representation: The probability distribution used in ECGA is a class of probability models known as marginal product models (MPMs). MPMs partition genes into mutually independent groups and specifies marginal probabilities for each linkage group. For example, the following MPM, `[1,3][2][4]`, for a four-bit problem represents that the 1st and 3rd genes are linked and 2nd and 4th genes are independent. An MPM must also specify probabilities for each substructure. For the above example, the MPM consists of the marginal probabilities: $\{p(x_1 = 0, x_3 = 0), p(x_1 = 0, x_3 = 1), p(x_1 = 1, x_3 = 0), p(x_1 = 1, x_3 = 1), p(x_2 = 0), p(x_2 = 1), p(x_4 = 0), p(x_4 = 1)\}$, where x_i is the value of the ith gene.

Class-Selection metric: To distinguish between better model instances from worse ones, ECGA uses a minimum description length (MDL) metric [20]. The key concept behind MDL models is that all things being equal, simpler models are better than more complex ones. The MDL metric used in ECGA is a sum of two components:

- **Model complexity** which quantifies the model representation size in terms of number of bits required to store all the marginal probabilities. Let, a given problem of size ℓ with alphabet cardinality χ, have m partitions with k_i genes in the ith partition, such that $\sum_{i=1}^{m} k_i = \ell$. Then each partition i requires $\chi^k - 1$ independent frequencies to completely define its marginal distribution. Furthermore, each frequency is of size $\log_2(n)$, where n is the population size. Therefore, the model complexity (or the model representation size), C_m, is given by

$$C_m = \log_2(n) \sum_{i=1}^{m} \left(\chi^{k_i} - 1\right). \tag{1}$$

- **Compressed population complexity**, which quantifies the data compression in terms of the entropy of the marginal distribution over all partitions.

$$C_p = n \sum_{i=1}^{m} \sum_{j=1}^{\chi^{k_i}} -p_{ij} \log_2 \left(p_{ij}\right), \tag{2}$$

  where p_{ij} is the frequency of the jth gene sequence of the genes belonging to the ith partition. In other words, $p_{ij} = N_{ij}/n$, where N_{ij} is the number of chromosomes in the population (after selection) possessing bit-sequence $j \in [1, \chi^{k_i}]$ [1] for i^{th} partition.

Class-Search method: In ECGA, both the structure and the parameters of the model are searched and optimized to best fit the data. While the probabilities are learned based on the variable instantiations in the population of selected individuals, a greedy-search heuristic is used to find an optimal or near-optimal probabilistic model. The search method starts by treating each decision variable as independent. The model-search method continues by merging two partitions that yields greatest improvement in the model-metric score. The subset merges are continued until no more improvement in the metric value is possible.

[1]Note that a BB of length k has χ^k possible sequences where the first sequence denotes be $00\cdots0$ and the last sequence $(\chi-1)(\chi-1)\cdots(\chi-1)$

5) Offspring creation: In ECGA, new individuals are created by sampling the probabilistic model. The offspring population is generated by randomly generating subsets from the current individuals according to the probabilities of the subsets as calculated in the probabilistic model.
6) Replacement: Many replacement schemes generally used in genetic and evolutionary computation—generational replacement, elitist replacement, niching, etc.—can be used in EDAs, but the key idea is to replace some or all the parents with some or all the offspring.
7) Repeat steps 2–6 until one or more termination criteria are met.

Analytical models have been developed for predicting the population-sizing and the scalability of ECGA [21]. The models predict that the population size required to solve a problem with m building blocks of size k with a failure rate of $\alpha = 1/m$ is given by

$$n \propto \chi^k \left(\frac{\sigma_{BB}}{d}\right) m \log m, \tag{3}$$

and the number of function evaluations is given by

$$n_{fe} \propto \left(\frac{\sigma_{BB}}{d}\right) \sqrt{k} \cdot \chi^k m^{1.5} \log m. \tag{4}$$

IV. Real-Coded ECGA

Real-coded estimation of distribution algorithms (EDAs) are complex and difficult to analyze. Our goal was to develop the simplest real-coded EDA possible. For this reason, we considered the most elementary discretization algorithm (i.e., equal-width discretization) and χECGA [8], possibly one of the simplest non-binary EDA, we combined them together so as to obtain a very simple real-coded ECGA.

Our rECGA works as follows, at each generation, given a population of n real-coded individuals of length l, tournament selection is applied with a rate S. The values of each gene j in the selected population are discretized into k intervals $I_{j,k}$, $1 \leq j \leq l$. Each discretization basically maps the gene j into a virtual alphabet [22] $\Sigma_j = \{s_{j,1} \ldots s_{j,k}\}$ whose symbols identify the intervals $I_{j,1}, \ldots I_{j,k}$. To keep the approach simple, among the many discretization algorithms available [23], we chose the simplest one, that is equal-width discretization. This partitions a range $[a, b]$ into k intervals of the same size [23]. The χECGA steps are applied on the discrete population: first the population model is built using a greedy MPM search; then a new discrete population is generated. The discrete population generated by the χECGA is mapped back into a real population by replacing, for each gene j, a symbol $s_{j,i}$ with a real value uniformly generated in the interval $I_{j,i}$. Finally, Restrict Tournament Replacement (RTR) [4], [24] is applied between the original real population (i.e., before tournament selection was applied) and the new real population obtained by "undiscretizing" the population obtained by the χ-ary ECGA. The process stops either when a certain number of generations have been reached or when the χECGA model has converged. The pseudo-code of our rECGA is shown as Algorithm IV. The algorithm is very simple and it basically adds only three steps to the typical χECGA: the discretization step (11), the undiscretization step (15), the restricted tournament selection step (16).

Algorithm 1 Real-Coded ECGA

1: **procedure** RECGA(k)
2: **var** k is the number of intervals
3: **var** rp is the real population
4: **var** dp is the discrete population
5: **var** $I_{i,j}$ is the j-th interval for gene i
6: $rp \leftarrow$ random();
7: Generate a random population rp
8: Evaluate the fitness in rp
9: **while** stop criterion not true **do**
10: Undergo tournament selection at a rate S
11: Discretize rp into dp using k and generate $I_{i,j}$
12: Model dp using a greedy MPM search
13: If the model has converged, stop
14: Generate a new dp_{+1} using the model
15: Generate a new rp_{+1} from dp_{+1} using $I_{i,j}$
16: $rp \leftarrow$ ApplyRTR(rp_{+1},rp)
17: Evaluate the fitness in rp
18: **end while**
19: **end procedure**

V. Experimental Results

We tested our simple real-coded ECGA on the real deceptive function f_{RDP},

$$f_{RDP}(\mathbf{y}) = \sum_{i=0}^{m-1} f_{trap}\left(y_{2i}, y_{2i+1}\right)$$

where $y_i \in [0, 1]$, m is the subproblem size, and $f_{trap}(\cdot, \cdot)$ is the two-dimensional trap (Figure 1),

$$f_{trap}\left(y_j, y_{j+1}\right) = \begin{cases} 1 & \text{if } y_j, y_{i+1} \geq 0.8 \\ 0.8 - \sqrt{\frac{y_j^2 + y_{j+1}^2}{2}} & \text{otherwise} \end{cases}$$

We performed a scalability analysis by applying the typical bisection procedure used in [25], [4] to determine the smallest population size and the smallest number of evaluations which guarantee the convergence.

At first, we analyzed our rECGA using the basic equal-width discretization with different number of intervals. Figure 2 reports (a) the number of evaluations and (b) the population size as functions of the problem size. The discretization based on four intervals is the more demanding in terms of number of evaluations and in population size. As the number of intervals increases, the number of evaluations and the population size decrease. The best results are obtained for a discretization with 5 intervals. Then, as the number of intervals increases, the number of evaluations and the population size required increase. These results show a clear trade-off between the number of intervals used and the

Fig. 1. Bidimensional description of the basis function of the real trap deceptive function.

number of evaluations required to converge. With fewer intervals, when the discretization is coarser, the alphabet used by χECGA is smaller so that the problem is potentially simpler for χECGA. However, since fewer intervals convey less information about the domain, overall rECGA needs more evaluations and more individuals to converge. As the number of interval increases, the identification of interesting areas in the problem domain is easier. The best partitioning is the one with five intervals. Five is in fact the smaller number of partitions that separates the interval [0.8-1.0] where the trap reach the optimum. Then, as the number of interval increases the problem becomes more demanding since the alphabet used by the underlying χECGA is larger. When we analyze the data in Figure 2 fitting them with a polynomial we find that the number of function evaluations (Figure 2a) grows sub-cubically while the population size grows sub-quadratically.

The trade-off between number of intervals and problem difficulty is more evident when we plot the same data as a function of the number of intervals (Figure 3). As Figure 3a shows, the number of evaluations decreases as the number of interval increases until five intervals are considered. With five intervals, when one interval actually coincides with the area of the trap function where the maximum is located, rECGA requires the least amount of computational resources possible. As we use a discretization with more than five intervals, the problem becomes more difficult since the number of symbols that χECGA has to deal with, becomes higher and higher.

In the second experiment, we analyzed how the performance of our simple rECGA is influenced by the tournament size. For this purpose, we considered the discretization with four intervals and applied rECGA with a tournament size of 16, 8, 4, and 2. Figure 4 reports (a) the number of evaluations and (b) the population size as a function of the problem size for different values of the tournament size S. As the tournament size increases, it becomes easier and easier for

Fig. 2. Real-Coded ECGA applied to the real deceptive function using equal-width discretization with different number of intervals: (a) number of evaluations, (b) population size.

rECGA to converge. As it should be expected, the difference in the number of evaluations and in the population size, is smaller in simpler problems and larger in more difficult problems. We repeated the same set of experiments for the 5 intervals discretization (the most favorable discretization for this problem) and for the 10 intervals discretization. The results are shown in Figure 5 and Figure 6 respectively.

VI. Summary and Conclusions

In this paper we present the design of a simple estimation of distribution algorithm that operate on real-coded variables. The proposed algorithm, called real-coded extended compact genetic algorithm (rECGA), uses discretization methods and χ-ary extended compact genetic algorithm. In rECGA, we map the real-coded variables into integer symbols of user-specified cardinalities using discretization methods. Then we use χECGA [8] to build and sample a probabilistic model of promising candidate solutions.

The sampled population of discrete symbols is mapped back into real-valued decision variables. As a first-step to-

Fig. 3. Real-Coded ECGA applied to the real deceptive function using equal-width discretization with different number of intervals: (a) number of evaluations and (b) population size as a function of the number of intervals.

Fig. 4. Real-Coded ECGA applied to the real deceptive function using 4 intervals and different values of tournament size S: (a) number of evaluations and (b) population size as a function of the problem size.

wards the analysis of the proposed algorithm, we investigated the effects of alphabet cardinality and selection pressure on the scalability of rECGA. The empirical results show that for a class of boundedly-difficult, additively-separable problems, the population size required by rECGA scales sub-quadratically with the number of decision variables. The results also show that the number of function evaluations scales sub-cubically with the problem size.

The proposed rECGA is simple, making it amenable for further empirical and theoretical analysis. Unlike other real-coded EDAs, the probabilistic model of rECGA—which is a partition of the decision variables into non-overlapping clusters—can be easily visualized and analyzed. Moreover, similar to the binary ECGA the probability model yields a direct mapping of linkage groups and also readily identifies key variable interactions. The proposed work is a first step towards systematically analyzing the strengths and weaknesses of existing different design decisions and towards developing a design theory for the development of scalable and robust real-coded EDAs.

Acknowledgments

In this research Luca and Pier Luca were partially funded by the European Community's Sixth Framework Programme, hArtes project (www.hartes.org).

This work was also sponsored by the Air Force Office of Scientific Research, Air Force Materiel Command, USAF, under grant FA9550-06-1-0096, the National Science Foundation under ITR grant DMR-03-25939 at Materials Computation Center, UIUC. The U.S. Government is authorized to reproduce and distribute reprints for government purposes notwithstanding any copyright notation thereon.

The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the Air Force Office of Scientific Research, the National Science Foundation, or the U.S. Government.

References

[1] D. E. Goldberg, "The race, the hurdle, and the sweet spot: Lessons from genetic algorithms for the automation of design innovation and

Fig. 5. Real-Coded ECGA applied to the real deceptive function using 5 intervals and different values of tournament size S: (a) number of evaluations and (b) population size as a function of the problem size.

Fig. 6. Real-Coded ECGA applied to the real deceptive function using 10 intervals and different values of tournament size S: (a) number of evaluations and (b) population size as a function of the problem size.

creativity," in Evolutionary Design by Computers, P. Bentley, Ed. San Mateo, CA: Morgan Kaufmann, 1999, ch. 4, pp. 105–118.

[2] ——, Design of innovation: Lessons from and for competent genetic algorithms. Boston, MA: Kluwer Academic Publishers, 2002.

[3] P. Larrañaga and J. A. Lozano, Eds., Estimation of distribution algorithms. Boston, MA: Kluwer Academic Publishers, 2002.

[4] M. Pelikan, Hierarchical Bayesian Optimization Algorithm (Toward a New Generation of Evolutionary Algorithms). Springer Berlin / Heidelberg, 2005.

[5] M. Pelikan, K. Sastry, and E. Cantú-Paz, Scalable Optimization via Probabilistic Modeling: From Algorithms to Applications (Studies in Computational Intelligence). Secaucus, NJ, USA: Springer-Verlag New York, Inc., 2006.

[6] J. A. Lozano, P. Larranaga, I. Inza, and E. Bengoetxea, Towards a New Evolutionary Computation: Advances on Estimation of Distribution Algorithms (Studies in Fuzziness and Soft Computing). Secaucus, NJ, USA: Springer-Verlag New York, Inc., 2006.

[7] G. Harik, "Linkage Learning via Probabilistic Modeling in the ECGA," 1999. [Online]. Available: citeseer.ist.psu.edu/harik99linkage.html

[8] L. de la Ossa, K. Sastry, and F. G. Lobo, "chi-ary Extended Compactd Genetic Algorithm in C++," IlliGAL, Tech. Rep., March, 2006.

[9] S. Tsutsui, M. Pelikan, and D. E. Goldberg, "Evolutionary algorithm using marginal histogram in continuous domain," in Optimization by Building and Using Probabilistic Models (OBUPM) 2001, San Francisco, California, USA, 7 2001, pp. 230–233. [Online]. Available: citeseer.ist.psu.edu/article/tsutsui01evolutionary.html

[10] S. Tsutsui and D. E. Goldberg, "Simplex crossover and linkage identification: Single-stage evolution vs. multi-stage evolution," 2002. [Online]. Available: citeseer.ist.psu.edu/tsutsui02simplex.html

[11] T. Higuchi, S. Tsutsui, and M. Yamamura, "Theoretical analysis of simplex crossover for real-coded genetic algorithms," in PPSN VI: Proceedings of the 6th International Conference on Parallel Problem Solving from Nature. London, UK: Springer-Verlag, 2000, pp. 365–374.

[12] M. Pelikan, D. E. Goldberg, and S. Tsutsui, "Combining the strengths of bayesian optimization algorithm and adaptive evolution strategies," pp. 512–519, 2002.

[13] ——, "Getting the best of both worlds: discrete and continuous genetic and evolutionary algorithms in concert," Inf. Sci., vol. 156, no. 3-4, pp. 147–171, 2003.

[14] C. W. Ahn, "Theory, design, and application of efficient genetic and evolutionary algorithms," Ph.D. dissertation, Gwangju Institute of Science and Technology, Korea, 2005.

[15] C.-H. Chen, W.-N. Liu, and Y.-P. Chen, "Adaptive discretization for probabilistic model building genetic algorithms," in GECCO '06: Proceedings of the 8th annual conference on Genetic and evolutionary computation. New York, NY, USA: ACM Press, 2006, pp. 1103–1110.

[16] G. Harik, "Linkage learning via probabilistic modeling in the ECGA," University of Illinois at Urbana-Champaign, Urbana, IL, IlliGAL Report No. 99010, January 1999.

[17] G. R. Harik, F. G. Lobo, and K. Sastry, "Linkage learning via

probabilistic modeling in the ECGA," in Scalable Optimization via Probabilistic Modeling: From Algorithms to Applications, M. Pelikan, K. Sastry, and E. Cantú-Paz, Eds. Berlin: Springer, 2006, ch. 3, pp. 39–61, (Also IlliGAL Report No. 99010).

[18] K. Sastry and D. E. Goldberg, "Probabilistic model building and competent genetic programming," in Genetic Programming Theory and Practise, R. L. Riolo and B. Worzel, Eds. Kluwer, 2003, ch. 13, pp. 205–220.

[19] L. de la Ossa, K. Sastry, and F. G. Lobo, "χ-ary extended compact genetic algorithm in C++," University of Illinois at Urbana-Champaign, Urbana, IL, IlliGAL Report No. 2006013, March 2006.

[20] J. J. Rissanen, "Modelling by shortest data description," Automatica, vol. 14, pp. 465–471, 1978.

[21] K. Sastry and D. E. Goldberg, "Designing competent mutation operators via probabilistic model building of neighborhoods," Proceedings of the 2004 Genetic and Evolutionary Computation Conference, vol. 2, pp. 114–125, 2004, also IlliGAL Report No. 2004006.

[22] D. E. Goldberg, "Real-coded genetic algorithms, virtual alphabets, and blocking," Complex Systems, vol. 5, pp. 139–167, 1991.

[23] E. Cantú-Paz, "Supervised and unsupervised discretization methods for evolutionary algorithms," Genetic and Evolutionary Computation Conference, January 24, 2001.

[24] G. R. Harik, "Finding multimodal solutions using restricted tournament selection," in Proceedings of the 6th International Conference on Genetic Algorithms. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1995, pp. 24–31.

[25] K. Sastry, "Evaluation-relaxation schemes for genetic and evolutionary algorithms," Master's thesis, University of Illinois at Urbana-Champaign, Urbana, IL, 2001, (Also IlliGAL Report No. 2002004).