Structure Learning and Optimisation in a Markov-network based Estimation of Distribution Algorithm

Alexander E.I. Brownlee, John A.W. McCall, Siddartha K. Shakya and Qingfu Zhang

***Abstract*—Structure learning is a crucial component of a multivariate Estimation of Distribution algorithm. It is the part which determines the interactions between variables in the probabilistic model, based on analysis of the fitness function or a population. In this paper we take three different approaches to structure learning in an EDA based on Markov networks and use measures from the information retrieval community (precision, recall and the F-measure) to assess the quality of the structures learned. We then observe the impact that structure has on the fitness modelling and optimisation capabilities of the resulting model, concluding that these results should be relevant to research in both structure learning and fitness modelling.**

I. Introduction

Estimation of Distribution Algorithms (EDAs) [1] is a well-established computing paradigm in the field of evolutionary algorithms. EDAs develop the concept of evolution found in a genetic algorithm, continuing the principals of selection and variation. They differ by replacing the traditional genetic reproduction operators of crossover and mutation with the construction and sampling of a probabilistic model. Markov networks have been proposed as the probabilistic model for a number of different EDAs [2], [3], [4], [5].

In [2], [3], [4] the variable interaction (independence) graph is learned from data using a statistical independence test. The structure is then refined to reduce its density and maximal cliques are found. Finally, a junction graph is learned in the case of MN-FDA [2], [3] or a Kikuchi Approximation is learned by MN-EDA [4] to approximate the distribution. In [6] an algorithm is proposed which uses the Linkage Detection Algorithm [7] to discover interactions when building a Boltzmann distribution of the fitness function. These approaches all result in an undirected structure which can also be used by the framework of Distribution Estimation Using Markov networks (DEUM) [8]. The fitness modelling approach of DEUM allows us to make observations as to the quality of the structure learned and its effect on the fitness modelling capability of the resulting model which should be of interest to others using undirected graphical models and the wider EDA community.

In this paper we investigate the impact of three structure learning approaches on the fitness modelling and optimisation capability of the underlying algorithm. We do this by extending the DEUM framework to include a structure learning step rather than relying on the existing knowledge of the problem structure. We use measures borrowed from the information retrieval community to assess the structures learned: specifically *precision* and *recall* combined in the *F-measure* [9]. We compare these with the fitness modelling and optimisation capabilities of the resulting models. This allows us to make observations on the structures learned by each algorithm which will be of relevance for other algorithms using an undirected structure.

The three different versions of DEUM with structure learning are:

- DEUM-LDA, a single-generation algorithm which incorporates the Linkage Detection Algorithm of Heckendorn and Wright
- DEUM-χ^2, a single-generation algorithm which incorporates an independence test structure learner using Pearson's Chi-Square statistics
- evDEUM-χ^2, a multi-generation variant of DEUM-χ^2

The results we present here relate to the algorithms' performances on optimising the Ising spin glass problem. This has previously been used as a benchmark problem for EDAs [10], [11], [12], [2], [3], [4], [5] due to interesting properties such as symmetry and a large number of plateaus. [10], [11], [12] demonstrated that EDAs using a directed model were able to successfully optimise 2D Ising. The problem exhibits an undirected network of interactions between variables and consequently EDAs using Markov networks are naturally suited to it - [2], [3], [4] demonstrated that Markov network based EDAs can learn learn and efficiently optimise the problem. Further, in [5] it was shown that by supplying the underlying lattice structure to the algorithm DEUM was also able to solve the problem.

The rest of this paper is structured as follows. In Section II we begin by discussing structure and our method for measuring the quality of learned structures. Section III discusses the DEUM framework for optimisation and fitness modelling. Sections V,VI and VII present the three approaches to incorporating structure learning into DEUM, followed by an analysis of the structure and fitness model learned by each, followed by optimisation results for each. Finally in Section VIII we draw our conclusions and look at possibilities for future work.

II. Learning the structure

The structure of the Markov Fitness Model in our previous work [13], [14], [8], [5], [15] has always been fixed and supplied prior to running the algorithm. In earlier work it was univariate - the model containing only one term for

A. Brownlee and J. McCall are with the School of Computing, Robert Gordon University, Aberdeen, UK (phone: +44 (0)1224 262472; email: sb,jm@comp.rgu.ac.uk).

S. Shakya is with the Intelligent Systems Research Centre, BT Group, Ipswich, UK (email: sid.shakya@bt.com).

Q. Zhang is with the Department of Computer Science, University of Essex, Colchester, UK (email: qzhang@essex.ac.uk).

978-1-4244-2959-2/09/$25.00 © 2009 IEEE

each variable in the problem. In [5], [15] the model was extended to incorporate terms representing bivariate and trivariate interactions among variables. The algorithm used in this work incorporates an additional step during which the structure is learned.

A. How good is the structure?

Other work has been done to analyse structures learned in EDAs. [16] included an analysis of EDA structure learning with a study of the structure learning capability of Learning Factorized Distribution Algorithm (LFDA) and [17] discussed the importance of higher-order interactions in EDAs. It is known that not all interactions which are present in a problem will necessarily be required in the model for the algorithm to rank individuals by fitness and find a global optimum. This concept is similar to the concept of unnecessary interactions [18]. Related is the idea of benign and malign interactions [19], essentially that some interactions result in a deceptive influence on fitness. This is in addition to the idea of spurious correlations [20], [21] which are false relationships in the model resulting from selection.

Here we build on these concepts by comparing the structure learned by the algorithm to what we call the *perfect model structure*. The perfect structure includes exactly those interactions which are present in the underlying fitness function. This does not include all possible interactions but does include those which influence the absolute fitness value. In the example Ising problem shown in Figure 1 these are: x_1x_2, x_2x_3, x_3x_4, x_1x_4, x_5x_6, x_6x_7, x_7x_8, x_5x_8, x_9x_{10}, $x_{10}x_{11}$, $x_{11}x_{12}$, x_9x_{12}, $x_{13}x_{14}$, $x_{14}x_{15}$, $x_{15}x_{16}$, x_{13x_16}, x_1x_5, x_2x_6, x_3x_7, x_4x_8, x_5x_9, x_6x_{10}, x_7x_{11}, x_8x_{12}, x_9x_{13}, $x_{10}x_{14}$, $x_{11}x_{15}$, $x_{12}x_{16}$, x_1x_{13}, x_2x_{14}, x_3x_{15}, x_4x_{16}. These interactions are required to perfectly fit the model to the fitness function and is only possible for predefined test problems where the interactions are explicit. In the case of Onemax there are no interactions. For the 2D Ising problem, an interaction exists wherever there is a coupling between two spin variables. In the example above, for a 16 bit 2D Ising problem the model will have 49 parameters in total including the univariate parameters and the constant.

In assessing the structures learned by the algorithm we use two measures from the information retrieval community: Precision p and Recall r [9]. These are defined in (1) and (2).

$$Precision = \frac{True\ interactions\ found}{Total\ interactions\ found} \tag{1}$$

$$Recall = \frac{True\ interactions\ found}{Total\ true\ interactions\ present} \tag{2}$$

That is, p measures how much of the learned structure comprises correctly identified interactions and r measures how many of the interactions present in the problem have been found. Both are proportions ranging from 0 to 1, with 1 being the best (if both measures are 1 then the learned structure perfectly matches the true structure). These can be combined into the single figure the F-measure (3) [9].

$$F = \frac{2 \cdot p \cdot r}{p + r} \tag{3}$$

This particular definition is known the F_1 measure in which p and r are equally weighted. As for the separate values, F-measure ranges from 0 to 1, with 1 representing p and r of 1.

III. Distribution Estimation using Markov networks

First we will describe the general framework of the DEUM algorithm, before going on to show the different approaches taken to incorporating a structure learning step.

A. General model

Previous publications on DEUM [5], [15] have described how the Markov network is used to model the distribution of energy across the set of variables in a problem. Energy follows a negative log relationship with fitness, so the Markov network can be used as a model of the fitness function which we call the Markov Fitness Model (MFM). For a solution $x = \{x_1, x_2, ..., x_n\}$, we can derive a function for each individual in the population (4). In this function the 0 and 1 bit values have been interpreted as -1 and +1 respectively.

$$-ln(f(x)) = \alpha_1 x_1 + \alpha_2 x_2 + ...\alpha_n x_n + \alpha_{12} x_1 x_2 + ... \tag{4}$$

In (4) each term represents a clique on the Markov network - this may be a 1-clique such as $\alpha_1 x_1$ or a 2-clique such as $\alpha_{12} x_1 x_2$. Additional terms may be added for higher order cliques in the same way as was done for 3-cliques in [15]; however for this study we restrict the model to 2-cliques to match the level of complexity of the 2D Ising problem. A system of equations can be formed by substituting the variable values and fitness for each individual within a population and a least squares approach used to estimate values for the α. The structure (which terms are present in the model) and the set of α values completely define the MFM. This can then be sampled to generate new individuals. Previously reported results for optimisation using DEUM have been found using the above techniques in the same broad workflow:

1. Generate random population
2. Repeat until termination criteria met:
   2.1. Compute model parameters
   2.2. Sample model to generate new population

A number of different techniques have been used to sample the MFM. In [22] the MFM was used to update a probability vector similar to that used in PBIL [23]. [13] described a technique to directly sample the marginal probabilities from a univariate MFM. [14], [5] employed a zero-temperature Metropolis method to sample the MFM and finally [5], [15] used variations of a Gibbs sampler. It was found empirically that the Gibbs sampler approach produced the best results for higher complexity problems and consequently it is only the Gibbs sampler which is used here - specifically the version

used when optimising the Ising problem in [5]. Depending on the problem and algorithm parameters we found that this approach would often find a global optimum in the first generation. This leads us to the single-step framework in Sections V and VI.

B. Fitness Prediction Correlation

In our results we also report the fitness prediction correlation C_r for each model, this measure is described in detail in [15], [24]. In contrast to the p, r and F values this measure is specific to the MFM approach but it still gives a helpful indication of the fitness information being learned by the model. Section III showed that DEUM models the fitness function directly. In addition to sampling the model to find an optimum we can also use it to predict the fitness of individuals; this ability can be exploited to measure how closely the MFM models the fitness function. This in turn acts as a predictor for the optimisation capability of the algorithm. C_r is the statistical correlation between the true fitness and the model predicted fitness of a population of randomly generated individuals. The experiments described here use Spearman's rank correlation [25]. As C_r approaches +1 the model has an increasingly strong positive correlation with the fitness function.

IV. Ising Problem and EDAs

The general Ising spin glass problem may be defined as an energy function $H(X)$ over a set of spin variables $X = X_1, X_2, ..., X_n$ and a set of coupling constants h and J as

$$H(X) = -\sum_{i \in L} h_i X_i - \sum_{i<j \in L} J_{ij} X_i X_j \qquad (5)$$

Each coupling constant $h_i \in h$ and $J_{ij} \in J$ relates to a single spin X_i and pair of spins X_i and X_j respectively. Each spin variable X_i can be either +1 or -1 and L represents a lattice of n spins.

Given a particular pair of values of the couple constants h_i and J_{ij}, the task is to find values for all X_i that minimises the energy H. In this paper we consider instances of the problem where each J_{ij} is restricted taking only the values +1 and -1, with the spins arranged in a two dimensional lattice of $n = l \times l$ spins. This is represented graphically in Figure 1.

In [11], [10], [12] it was shown that the hierarchical Bayesian Optimization Algorithm (hBOA) was able to efficiently solve the problem outperforming other algorithms. In [2], [3] the Ising problem was used as a benchmark for the Markov Network Estimation of Distribution Algorithm (MN-EDA) and the Markov Network Factorized Distribution Algorithm (MN-FDA). The latter of these papers argued that the Kikuchi approximation used to estimate the distribution used by MN-EDA gave an advantage for the 2D Ising problem as it was able to represent the bivariate dependencies as an exact factorisation. Finally [5] demonstrated that this also applied to the DEUM algorithm as it could represent the exact factorisation from the structure in the form of potential functions. DEUM was demonstrated to perform very well - solving the problem with only a single generation. In that case the structure was supplied to the algorithm; by adding a structure learning component this work will allow a fairer comparison with the other EDA on the Ising problem. In the experiments which follow we use the same four instances of Ising at each size as those used in [5], with the results aggregated into a mean for each size.

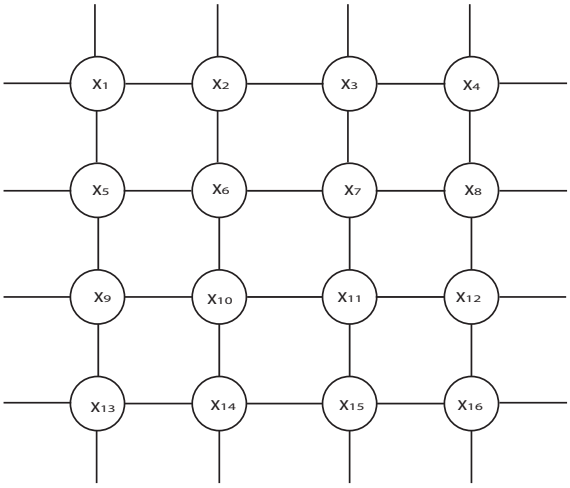

Fig. 1: 16bit 2D Ising lattice

V. DEUM-LDA

This approach adds the Linkage Detection Algorithm [7] to the DEUM framework, giving us the following workflow.

1. Run LDA on the fitness function
2. Generate random initial population π, of size [4.4N]
3. Select a subset σ, the top [1.1N] of π
4. Use σ to build MFM
5. Calculate C_r
6. Sample new population from MFM using random walk Gibbs sampler

Step 5 is not an integral part of the algorithm. It is only included in the above workflow to show the point at which we calculate the value for C_r.

The proportion of the population selected and population size used were determined by earlier experiments described in [24]. There we found that fitness modelling capability of the MFM increases greatly when the number of individuals selected is more than the number of parameters in the model (N). Once the structure is known, we then set the number to be selected to 1.1N to ensure that this is the case (we call this an over-specified system). The population size is then set to be large enough to achieve the desired selection pressure. The results in [24] revealed that fitness modelling capability improves as the selection pressure is increased. In this case we set the proportion of the population selected to 0.25 - that is, the population is four times the number of individuals required to build the MFM. This represents a good tradeoff between a useful selective pressure and an excessively large population.

The cooling rate parameter for the Gibbs sampler was 0.0005, taken from [5]. The iteration cap of the sampler was

PS	p	p-SD	r	r-SD	F	C_r	C_r-SD
16	1.00	0.00	1.00	0.00	1.00	0.887	0.091
25	1.00	0.00	1.00	0.00	1.00	0.924	0.049
36	1.00	0.00	1.00	0.00	1.00	0.932	0.034
49	1.00	0.00	1.00	0.00	1.00	0.939	0.037
64	1.00	0.00	1.00	0.00	1.00	0.949	0.023
100	1.00	0.00	1.00	0.00	1.00	0.942	0.027
256	1.00	0.00	1.00	0.00	1.00	0.945	0.018
324	1.00	0.00	1.00	0.00	1.00	0.943	0.015
400	1.00	0.00	1.00	0.00	1.00	0.940	0.022

TABLE I: DEUM-LDA Fitness Model Statistics over 30 runs

PS	C_r	LDA-FE	FE	FE-SD	IT	IT-SD	SR
16	0.887	480	210	4	163	310	100
25	0.920	1200	330	1	989	532	100
36	0.932	2520	483	25	821	1995	100
49	0.905	4704	647	3	1675	1234	100
64	0.939	8064	846	1	902	688	100
100	0.944	19800	1446	221	11989	21296	87
256	0.950	130560	4342	862	130995	118419	67
324	-	-	-	-	-	-	0
400	-	-	-	-	-	-	0

TABLE II: DEUM-LDA Optimisation Statistics over 30 runs

increased from 500 to 2000 - this improved the success rate of the algorithm and decreased the total number of iterations required to find the global optimum.

A. Fitness Model

First we will look at the model generated by the algorithm for each problem size. In Table I we see the mean precision p and recall r for the structures found over 30 runs, with corresponding standard deviations (p-SD and r-SD). This is followed by the F-measure F combining the mean precision and recall. This is shown alongside the mean fitness prediction correlation C_r for each size of the problem with its corresponding standard deviation C_r-SD.

With a precision and recall (and hence F) of 1.0, the structure discovered matches the perfect structure for the problem. This means we would expect similar optimisation results to those seen in [5].

B. Optimisation

Table II shows the results for optimising using the algorithm. The C_r value is given to show the fitness prediction power of the model in the context of optimisation. The number of fitness evaluations needed by LDA (FE-LDA) for each problem size is given next. Then the mean number of additional function evaluations (FE) and internal iterations of the Gibbs sampler (IT) are given along with their standard deviations (FE-SD and IT-SD). FE includes the evaluations needed to estimate model parameters and the evaluations needed to confirm the fitness of individuals generated by the Gibbs sampler. All of these value only include the successful runs; the success rate (SR) is given as a percentage of times the algorithm found a known global optimum over 30 independent runs.

The results were unexpectedly poor given the perfect structure learned by LDA (in [5] DEUM was able to optimise 2D Ising when supplied with the perfect structure). We increased

PS	C_r	FE-LDA	FE	FE-SD	IT	IT-SD	SR
100	0.993	19800	2409	7	8015	6825	100
256	0.993	130560	6166	48	37705	82985	100
324	0.993	209304	7786	7	18947	11962	80
400	0.992	319200	9688	86	160237	154439	100

TABLE III: DEUM-LDA Optimisation Statistics with increased population size over 30 runs

the number of individuals selected from the population to estimate the model parameters to 2N (that is, twice the number of parameters in the model). Part of our work on the fitness information content in a population [24] indicated that Ising requires a larger number of individuals than other fitness functions we have looked at to obtain a good model of fitness. The results for the rerun are given in Table III. It can be seen that this resulted in a marked improvement of the optimisation capability, even for the largest instances of the problem that we used.

One problem with this approach is that the linkage detection algorithm requires a large number of fitness evaluations to learn the structure. This could be mitigated by recycling the solutions generated by the LDA run for estimating the model parameters but this would not make a large difference. The number of possible interactions for a particular problem size n is given in (6); given that LDA requires four evaluations for each possible interaction, the number of evaluations required by LDA at a particular problem size n is given in (7). This can be reduced by caching individuals but not by a large amount and this comes with a rapidly growing space complexity.

$$Possible = n \times (n-1)/2 \tag{6}$$

$$Evals = 2 \times (n \times (n-1)) \tag{7}$$

In addition to this, without some threshold it will include any interactions which are not useful for optimisation and this will lead to a large and overly complex model which will result in the algorithm being overloaded and potentially resulting in poor performance. An effect similar to this was seen with the noisy chemotherapy problem used for benchmarking hBOA in [26]. The 2D Ising problem is in comparison very "clean" - because of the nature of the problem any interactions which LDA discovers are important for optimisation with the result that the structure found is perfect. This allows DEUM to build a model which closely matches the fitness function. For other problems this may not be possible which provides motivation for the independence test approach.

VI. DEUM-χ^2

A. The Algorithm

The workflow for this algorithm is very similar to that for DEUM incorporating LDA. The only additions are the extra selection step to choose individuals for the Chi-Square structure learning algorithm and the structure refinement step, both taken from [2], [3], [4]. The structure is learned by

PS	p	p-SD	r	r-SD	F	C_r	C_r-SD
16	0.785	0.183	0.979	0.028	0.988	0.997	0.01
25	0.749	0.213	0.968	0.029	0.98	0.992	0.015
36	0.73	0.166	0.97	0.022	0.982	0.994	0.008
49	0.754	0.101	0.963	0.019	0.976	0.99	0.012
64	0.741	0.107	0.958	0.018	0.974	0.99	0.008
100	0.695	0.116	0.948	0.018	0.966	0.985	0.01
256	0.589	0.092	0.914	0.009	0.94	0.968	0.007
324	0.584	0.083	0.908	0.011	0.935	0.964	0.008
400	0.529	0.085	0.894	0.011	0.924	0.956	0.007

TABLE IV: DEUM-χ^2 Fitness Model Statistics over 30 runs

PS	p	p-SD	r	r-SD	F	C_r	C_r-SD
16	0.8	0.037	0.703	0.052	0.854	0.573	0.163
25	0.896	0.051	0.816	0.053	0.921	0.733	0.095
36	0.954	0.02	0.89	0.027	0.976	0.869	0.05
49	0.992	0.011	0.961	0.023	0.989	0.941	0.047
64	0.995	0.007	0.984	0.009	1	0.974	0.014
100	1	0	1	0.002	1	0.992	0.004
256	1	0	1	0	1	0.993	0.002
324	1	0	1	0	1	0.993	0.002
400	1	0	1	0	1	0.993	0.001

TABLE V: DEUM-χ^2 Fitness Model Statistics with LDA-equivalent population size over 30 runs

PS	C_r	FE	FE-SD	IT	IT-SD	SR
16	0.814	216	17	646	1074	90
25	0.831	340	32	1278	2828	77
36	0.830	479	21	887	1621	60
49	0.807	722	88	6863	6948	50
64	0.816	1005	256	17435	26404	43
100	-	-	-	-	-	0
256	-	-	-	-	-	0
324	-	-	-	-	-	0
400	-	-	-	-	-	0

TABLE VI: DEUM-χ^2 Optimisation Statistics over 30 runs

performing a first-order Chi-Square independence test on each possible pairing of variables; an interaction is assumed where the test exceeds a threshold of 0.75. The number of individuals for the structure learning selection was set to the top 25%; the selection step for estimating the MFM parameters selected the top 1.1N individuals. These figures were found to represent a good balance between fitness modelling capability and function evaluations required by a series of experiments following the pattern described in [24]. The structure refinement step is the same as that used in [2], [3], [4]: a limit is imposed on the number of edges (interactions) incident to a node (variable) on the graph. For any node exceeding this limit, the edges with the lowest Chi-Square scores are removed until the limit is reached. For 2D Ising we set this limit to 4 - that is, each variable can have only four neighbours as is the case in the 2D Ising lattice. The clique finding step from those papers is not applied here as we know that the 2D Ising problem has a bivariate structure and we wish to keep the structure at this level of complexity. The population size was set to be the problem size multiplied by 100. As in the previous section, the cooling rate for the Gibbs sampler was 0.0005 and the iteration cap was set to 2000.

1. Generate random initial population π
2. Select a subset σ_1, the top 25% of π
3. Run Chi-Square edge detection algorithm to search for statistical dependencies apparent in σ_1
4. Refine structure
5. Select a subset σ_2, the top 1.1N of π
6. Use σ_2 to build MFM
7. Calculate C_r
8. Sample new population from MFM using random walk Gibbs sampler

B. Fitness Model

Again, before looking at optimisation results we will look at the model generated by the algorithm for each problem size. These are shown in Table IV; the column headings are the same is in Table I.

We can see that increasing problem size results in a decrease in both p and r, reflected in a steadily decreasing F. This indicates that with increasing problem size the algorithm finds it more difficult to correctly identify all interactions and also begins to match some false positives. This also results in a corresponding decrease in the fitness prediction power of the model, revealed in the decreasing C_r values. We can see that these fall off very quickly with a comparatively small decrease in r; this highlights the importance of finding a good model structure. The experiment was rerun with population sizes equal to the number of fitness evaluations required by LDA at each step; the results for this are given in Table V. We can see that the structures are considerably better at the larger sizes; for the smaller sizes the number of evaluations required by LDA was actually less than 100 X problem size as used in the previous experiment so the resulting structures are poorer. The increased C_r values for this algorithm on large problem sizes are a by-product of this algorithm's workflow. In DEUM-LDA the parameter estimation step selected individuals from a new population of size 4.4N rather than recycling that produced in the course of the LDA run. DEUM-χ^2 uses the structure learning population, which is very large. The high selective pressure results in a slightly better model of fitness with the same model structure, in line with the results reported in [24].

C. Optimisation Results

We know from the results in section V that the algorithm can find the global optimum when the learned structure has F equal to 1.0 relative to the true structure so the optimisation experiment was instead run on the structures learned by the Chi-Square algorithm with the smaller population - statistics for which are shown in Table IV. The motivation for this is that if the global optimum can be found using these imperfect structures then we have a significant saving in function evaluations over the LDA based algorithm.

Now we run the full optimisation algorithm incorporating the Chi-Square structure learner and report the results in Table VI. The column headings are the same as Table II.

We can see that with the decrease in the fitness prediction capability of the model there is a marked decrease in the optimisation capability of the algorithm. Indeed, it is clear

PS	C_r	FE	FE-SD	IT	IT-SD	SR
16	0.951	381	5	598	230	100
25	0.971	599	5	2122	2142	100
36	0.951	852	9	1425	543	100
49	0.940	1166	13	8192	14735	100
64	0.936	1505	16	1875	1941	100
100	0.939	2799	744	316472	525816	70
256	-	-	-	-	-	0
324	-	-	-	-	-	0
400	-	-	-	-	-	0

TABLE VII: DEUM-χ^2 Optimisation Statistics with increase population size over 30 runs

that the C_r values for the runs which proved successful (Table VI) were on average higher than those found across all runs IV. To improve on these results, the population size was again increased to 2N for the larger instances of the problem and the experiment rerun.

In contrast to the previous section, we see that the results are still poor. This can be attributed to the imperfections in the structure learned by the independence test method. The recall values of around 0.9 indicate that up to 10% of the interactions present in the perfect structure are missing; precision values of around 0.96 indicate that up to 4% of the interactions which were added to the model are not present in the perfect structure.

VII. EvDEUM-χ^2

Based on the poor results for the single-step algorithm it is worth determining whether an evolutionary approach would be able to overcome the issues with poor structure.

One important change was a reduction in the run time for the Gibbs sampler. With this algorithm there is no longer an assumption that a model with a close fit to the fitness function will be found in the first generation. This means that areas of high probability within the model will not necessarily be areas of high fitness, and running the Gibbs sampler slowly to convergence is likely to result in an individual of inferior fitness to the global optimum. The algorithm was initially run with a fixed value for cooling rate and maximum number of iterations. It was found that performance was improved by varying these parameters over the course of the evolution - reducing the cooling rate and increasing the maximum number of iterations with each generation. This allowed the algorithm to be balanced towards exploration in early generations and exploitation in later ones as the model fits more closely to the fitness function. For each generation g, the cooling rate r was calculated according to (8) and the maximum number of iterations of the Gibbs sampler I was calculated according to (9). The parameters for these were determined empirically to yield the best results.

$$r = 1 + (g^2/10) \tag{8}$$

$$I = 1/20g \tag{9}$$

We also adopted a steady-state approach for the algorithm, replacing 5% of the population each generation. This allows

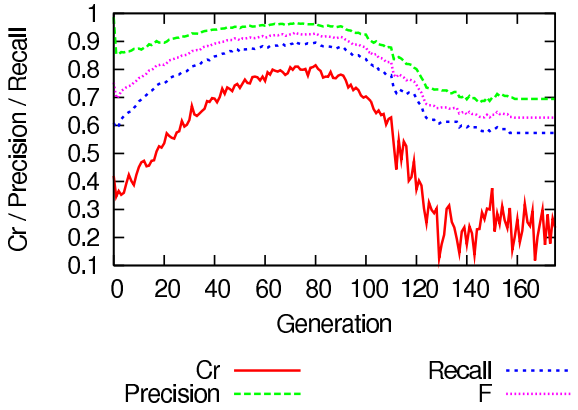

Fig. 2: Fitness Model Statistics for EvDEUM-χ^2 on 25bit 2D Ising lattice over 30 runs

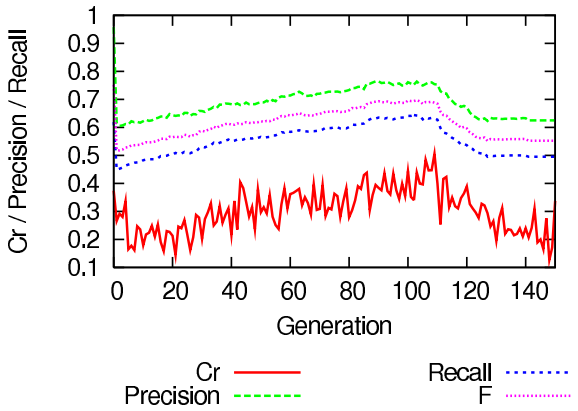

Fig. 3: Fitness Model Statistics for EvDEUM-χ^2 on 100bit 2D Ising lattice over 30 runs

us to use a large population for the structure learning component and maintain diversity for as long as possible.

1. Generate random initial population π
2. Select a subset σ_1, the top 25% of π
3. Run Chi-Square edge detection algorithm to search for statistical dependencies apparent in σ_1
4. Refine structure
5. Select a subset σ_2, the top 1.1N of π
6. Use σ_2 to build MFM
7. Calculate C_r
8. Sample R new individuals from MFM using random walk Gibbs sampler and replace poorest R individuals in π with these

A. Fitness Model

As there are now multiple models being created, there are multiple figures for p, r, F and C_r. For ease of interpretation, the values over the course of evolution for three instances of the problem (25 bit, 100 bit and 256 bit) are represented graphically in Figures 2, 3 and 4. The four measures can be shown relative to the same y-axis as they all have a range of zero to one (Strictly speaking C_r has a range of -1 to

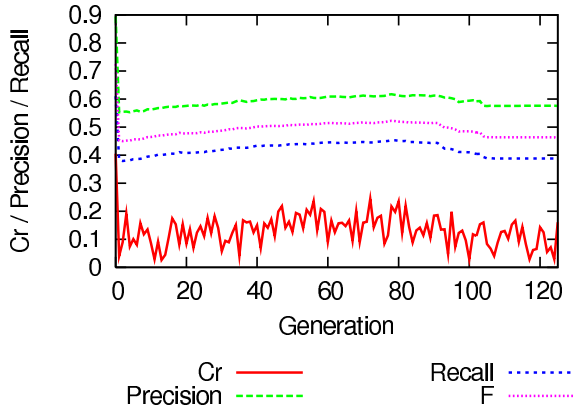

Fig. 4: Fitness Model Statistics for EvDEUM-χ^2 on 256bit 2D Ising lattice over 30 runs

+1 but in the examples here it is always positive). 25 bits was chosen for the first problem to look at rather than 16 bits because the algorithm was often able to solve the 16 bit instances of the problem in a single or very small number of generations so the chance to observe an effect over many generations was reduced.

We can see that the structure quality and consequently the fitness modelling capability of the model rise to begin with and then fall off as the population converges and diversity decreases. Further work is needed to determine the factors which affect the point at which this occurs. With increasing problem size the maximum values reached for structure quality and fitness prediction capability becomes lower, never exceeding an F of 0.5 or a C_r of 0.3 as evolution proceeds. This means the model is unlikely to be good enough to allow the algorithm to find the global optimum. It is also notable that in the first generation the precision and recall of the structure and C_r for the model are all relatively high, this then drops off immediately in the second generation.

B. Optimisation Results

The results for the optimisation capability of the algorithm are shown in Table VIII. As before, the mean number of function evaluations (FE) and internal iterations of the Gibbs sampler (IT) are given along with their standard deviations (FE-SD and IT-SD). All of these values only include the successful runs; the success rate (SR) is given as a percentage over 30 independent runs. No C_r values are present this time because that value varied over the course of the evolution.

PS	FE	FE-SD	IT	IT-SD	SR
16	2465	3254	4292	10233	100
25	21135	2861	2564001	2010019	83
36	25913	911	3226402	1154681	100
49	33321	1826	8193559	3778421	67
64	-	-	-	-	0
100	-	-	-	-	0
256	-	-	-	-	0
324	-	-	-	-	0
400	-	-	-	-	0

TABLE VIII: EvDEUM-χ^2 Optimisation Statistics over 30 runs

The results in this section reflect the poor quality of the models learned. We can see that even for small instances of the problem, the success rate is below 100% and the total number of function evaluations used by the algorithm is higher than for the single step algorithms. As the problem size increases, the perfect structure is never found and the algorithm is unable to find the global optimum. In each generation, the model build time is reduced because the structure learned has fewer interactions, means fewer terms in the model. The sampling times are greatly reduced over the single step approach as we are deliberately lowering the iteration cap on the sampler to encourage diversity in the population. Both of these benefits are lost when repeated over many generations. While the number of function evaluations is reduced in each generation (compared to the number required by the single-step structure learning algorithms), the evolutionary approach does not allow the structure to be found with reduced data.

VIII. Conclusion

In this paper we have extended the DEUM framework to incorporate a structure learning step. In our analysis of the different structure learning approaches we have introduced measures which are new to the EDA community: precision, recall and the F-measure. We believe that these are helpful terms when comparing structure learning algorithms on known benchmark problems such as Ising. They differ from existing terms which describe aspects of structure such as benign/malign and unnecessary interactions - those terms describe the influence an interaction has on fitness and whether an interaction is required by the model for optimisation. Precision, recall and the F-measure refer specifically to the number of interactions that the structure learning algorithm has found relative to the known structure and allow a precise measurement of the effectiveness of a structure learning algorithm.

We have seen that the algorithm is able to optimise successfully, but the overhead (model build and sampling time) is expensive. The MFM requires a near-perfect structure to be supplied and a large population for optimisation to be successful. An incremental approach to the model building step like that of iBOA [27] offers a potential improvement and offers potential for future work. Additionally, other ways of making use of the fitness model may give better results than the Gibbs sampler. These include guided operators in a hybrid algorithm incorporating guided operators [28] and surrogate fitness models [29], [30], [31], [32].

An interesting observation to come out of this work is the dropoff in fitness modelling capability as evolution proceeds. We attribute this to loss of diversity in the population. The problem was mitigated by the use of a steady-state approach but did still prove a hindrance over time. This relates to other work on diversity loss [33], [34], [35], [36], [37] and another avenue for future work is mutation of the probabilistic model,

niching [38] or other technique to reduce the effect of diversity loss and improve optimisation performance.

With the simple addition of a maximal clique finding algorithm it will also be possible to apply this approach to problems with higher order interactions such as SAT. It will be interesting to see if the effects described here hold true for these other problems.

REFERENCES

[1] P. Larrañaga and J. A. Lozano, *Estimation of Distribution Algorithms: A New Tool for Evolutionary Computation*. Boston: Kluwer Academic Publishers, 2002.

[2] R. Santana, "A Markov network based factorized distribution algorithm for optimization," in *Proceedings of the 14th European Conference on Machine Learning (ECML-PKDD 2003); Lecture Notes in Artificial Intelligence*, vol. 2837. Berlin: Springer-Verlag, 2003, pp. 337–348.

[3] R. Santana, "Probabilistic modeling based on undirected graphs in estimation distribution algorithms," Ph.D. dissertation, Institute of Cybernetics,Mathematics and Physics, Havana, Cuba, 2003.

[4] R. Santana, "Estimation of distribution algorithms with Kikuchi approximations," *Evol. Comp.*, vol. 13, no. 1, pp. 67–97, 2005.

[5] S. K. Shakya, J. A. W. McCall, and D. F. Brown, "Solving the Ising spin glass problem using a bivariate EDA based on Markov random fields," in *Proceedings of IEEE Congress on Evolutionary Computation*. IEEE Press, 16-21 July 2006 2006.

[6] A. H. Wright and S. Pulavarty, "On the convergence of an estimation of distribution algorithm based on linkage discovery and factorization," in *GECCO '05: Proceedings of the 2005 conference on Genetic and evolutionary computation*. ACM Press, 2005, pp. 695–702.

[7] R. B. Heckendorn and A. H. Wright, "Efficient linkage discovery by limited probing," *Evol. Comp.*, vol. 12, no. 4, pp. 517–545, 2004.

[8] S. K. Shakya, "DEUM: A framework for an estimation of distribution algorithm based on Markov random fields," Ph.D. dissertation, The Robert Gordon University, Aberdeen, UK, 2006.

[9] I. H. Witten and E. Frank, *Data Mining: Practical Machine Learning Tools and Techniques*, 2nd ed., ser. Morgan Kaufmann Series in Data Management Sys. Morgan Kaufmann, June 2005.

[10] M. Pelikan and D. Goldberg, "Hierarchical BOA solves Ising spin glasses and MAXSAT," in *Proceedings of the Genetic and Evolutionary Computation Conference 2003 (GECCO-2003)*. Springer-Verlag, 2003 2003, pp. 1271–1282.

[11] M. Pelikan, "Bayesian optimization algorithm: from single level to hierarchy," Ph.D. dissertation, University of Illinois at Urbana-Champaign, Urbana, IL, 2002.

[12] M. Pelikan, J. Ocenasek, S. Trebst, M. Troyer, and F. Alet, "Computational complexity and simulation of rare events of Ising spin glasses," in *GECCO (2)*, 2004, pp. 36–47.

[13] S. K. Shakya, J. A. W. McCall, and D. F. Brown, "Estimating the distribution in an EDA," in *Proceedings of the International Conference on Adaptive and Natural computiNG Algorithms (ICANNGA 2005)*. Springer-Verlag, 21-23 March 2005, pp. 202–205.

[14] S. K. Shakya, J. A. W. McCall, and D. F. Brown, "Incorporating a Metropolis method in a distribution estimation using Markov random field algorithm," in *Proceedings of IEEE Congress on Evolutionary Computation*. IEEE, 2-5 September 2005, pp. 2576–2583.

[15] A. E. I. Brownlee, J. A. W. McCall, and D. F. Brown, "Solving the MAXSAT problem using a multivariate EDA based on Markov networks," in *GECCO '07: Proceedings of the 2007 GECCO Conference on Genetic and Evolutionary computation*. New York, NY, USA: ACM, 2007, pp. 2423–2428.

[16] H. Mühlenbein and R. Höns, "The estimation of distributions and the minimum relative entropy principle," *Evol. Comput.*, vol. 13, no. 1, pp. 1–27, 2005.

[17] Q. Zhang, "On stability of fixed points of limit models of univariate marginal distribution algorithm and factorized distribution algorithm," *IEEE Trans. Evolutionary Computation*, vol. 8, no. 1, pp. 80–93, 2004.

[18] M. Hauschild, M. Pelikan, C. F. Lima, and K. Sastry, "Analyzing probabilistic models in hierarchical BOA on traps and spin glasses," in *GECCO 2007: Proceedings of the 9th annual conference on Genetic and Evolutionary Computation*. New York, NY, USA: ACM, 2007, pp. 523–530.

[19] L. Kallel, B. Naudts, and R. Reeves, "Properties of fitness functions and search landscapes," in *Theoretical Aspects of Evolutionary Computing*, L. Kallel, B. Naudts, and A. Rogers, Eds. Springer Verlag, 2000, pp. 177–208.

[20] H. Mühlenbein and T. Mahnig, "Evolutionary optimization using graphical models," *New Gen. Comput.*, vol. 18, no. 2, pp. 157–166, 2000.

[21] R. Santana, P. Larrañaga, and J. A. Lozano, "Challenges and open problems in discrete EDAs," Department of Computer Science and Artificial Intelligence, University of the Basque Country, Tech. Rep. EHU-KZAA-IK-1/07, October 2007. [Online]. Available: http://www.sc.ehu.es/ccwbayes/technical.htm

[22] S. K. Shakya, J. A. W. McCall, and D. F. Brown, "Updating the probability vector using MRF technique for a univariate EDA," in *Proceedings of STAIRS 2004*. IOS Press, 2004 2004, pp. 15–25.

[23] S. Baluja and R. Caruana, "Removing the genetics from the standard genetic algorithm." Morgan Kaufmann Publishers, 1995, pp. 38–46.

[24] A. Brownlee, J. McCall, Q. Zhang, and D. Brown, "Approaches to selection and their effect on fitness modeling in an estimation of distribution algorithm," in *Proceedings of the 2008 World Congress on Computational Intelligence (WCCI-2008)*, 2008.

[25] T. Lucey, *Quantatitive Techniques: An Instructional Manual*. Eastleigh, Hampshire, UK: D. P. Publications, 1984.

[26] A. E. Brownlee, M. Pelikan, J. A. McCall, and A. Petrovski, "An application of a multivariate estimation of distribution algorithm to cancer chemotherapy," in *GECCO '08: Proceedings of the 10th annual conference on Genetic and evolutionary computation*. New York, NY, USA: ACM, 2008, pp. 463–464.

[27] M. Pelikan, K. Sastry, and D. E. Goldberg, "iBOA: The incremental Bayesian optimization algorithm," in *GECCO '08: Proceedings of the 10th annual conference on Genetic and evolutionary computation*. New York, NY, USA: ACM, 2008, pp. 455–462.

[28] Q. Zhang, J. Sun, and E. Tsang, "An evolutionary algorithm with guided mutation for the maximum clique problem." *IEEE Trans.Evolutionary Computation*, vol. 9, no. 2, pp. 192–200, 2005.

[29] M. Pelikan and K. Sastry, "Fitness inheritance in the Bayesian optimization algorithm," in *GECCO (2)*, 2004, pp. 48–59.

[30] C. F. Lima, M. Pelikan, K. Sastry, M. V. Butz, D. E. Goldberg, and F. G. Lobo, "Substructural neighborhoods for local search in the Bayesian optimization algorithm," in *Parallel Problem Solving from Nature – PPSN IX, (9th PPSN'06)*, ser. Lecture Notes in Computer Science (LNCS). Reykjavik, Iceland: Springer-Verlag, Sept. 2006, vol. 4193, pp. 232–241.

[31] K. Sastry, C. Lima, and D. E. Goldberg, "Evaluation relaxation using substructural information and linear estimation," in *Proceedings of the 8th annual Conference on Genetic and Evolutionary Computation GECCO-2006*. New York, NY, USA: ACM Press, 2006, pp. 419–426.

[32] A. Orriols-Puig, E. Bernadó-Mansilla, K. Sastry, and D. E. Goldberg, "Substructural surrogates for learning decomposable classification problems: implementation and first results," in *GECCO '07: Proceedings of the 2007 GECCO conference on genetic and evolutionary computation*. New York, NY, USA: ACM, 2007, pp. 2875–2882.

[33] A. Ochoa and M. R. Soto, "Linking entropy to estimation of distribution algorithms," in *Towards a New Evolutionary Computation: Advances on Estimation of Distribution Algorithms*, J. A. Lozano, P. L. naga, I. Inza, and E. Bengoetxea, Eds. Springer-Verlag, 2006, pp. 1–38.

[34] H. Handa, "Estimation of distribution algorithms with mutation," in *Evolutionary Computation in Combinatorial Optimization*, ser. Lecture Notes in Computer Science, vol. 3448. Springer Berlin / Heidelberg, 2005, pp. 112–121.

[35] T. Mahnig and H. Mühlenbein, "Optimal mutation rate using Bayesian priors for estimation of distribution algorithms," in *SAGA '01: Proceedings of the International Symposium on Stochastic Algorithms*. London, UK: Springer-Verlag, 2001, pp. 33–48.

[36] J. Branke, C. Lode, and J. L. Shapiro, "Addressing sampling errors and diversity loss in UMDA," in *GECCO '07: Proceedings of the 9th annual conference on Genetic and evolutionary computation*. New York, NY, USA: ACM, 2007, pp. 508–515.

[37] P. Posik, "Preventing premature convergence in a simple EDA via global step size setting," in *PPSN*, 2008, pp. 549–558.

[38] W. Dong and X. Yao, "NichingEDA: Utilizing the diversity inside a population of EDAs for continuous optimization," in *IEEE World Congress on Computational Intelligence 2008 (CEC 2008)*, June 2008, pp. 1260–1267.