2003

Ordering Genetic Algorithm Genomes with Reconstructability Analysis

Stephen Shervais
Eastern Washington University

Martin Zwick
Portland State University, zwick@pdx.edu

Citation Details

Shervais, Stephen, and Martin Zwick†. "Ordering genetic algorithm genomes with reconstructability analysis." International Journal of General Systems 32.5 (2003): 491-502.

ORDERING GENETIC ALGORITHM GENOMES WITH RECONSTRUCTABILITY ANALYSIS

Stephen Shervais
Eastern Washington University
College of Business and Public Administration
Cheney, WA 99004
sshervais@ewu.edu

Martin Zwick
Portland State University
Systems Science Ph.D. Program
Portland, OR 97201
zwickm@pdx.edu

KEYWORDS

Reconstructability analysis, genetic algorithms, transposition, crossover, optimization, OCCAM

ABSTRACT

The building block hypothesis implies that genetic algorithm effectiveness is influenced by the relative location of epistatic genes on the chromosome. We find that this influence exists, but depends on the generation in which it is measured. Early in the search process it may be more effective to have epistatic genes widely separated. Late in the search process, effectiveness is improved when they are close together. The early search effect is weak but still statistically significant; the late search effect is much stronger and plainly visible. We demonstrate both effects with a set of simple problems, and show that information-theoretic reconstructability analysis can be used to decide on optimal gene ordering.

I. INTRODUCTION

Holland's schema theorem and the building block hypothesis suggest that the performance of a genetic algorithm (GA) will be influenced by the relative location on the chromosome of genes exhibiting epistasis, i.e., genes having a strong interaction effect in their effect on fitness. At first glance, the theory implies that epistatic genes should be placed close together. Genes that are close are less likely to be separated by the crossover operator, and alleles that have high fitness can constitute a building block for further evolution. An alternative hypothesis, however, might be that epistatic genes should be located distant from one another on the chromosome, precisely because they then are more likely to be separated by the crossover operator, since this will allow the GA to explore a wider region of the search space. Epistasis may thus imply the existence of an optimal gene order for a GA but does not tell us what it is, if it is a single order, or even if it is constant.

Shervais, S. and Zwick, M. (2003). "Ordering Genetic Algorithm Genomes With Reconstructability Analysis." International Journal of General Systems, Vol. 32(5), pp. 491-502

This paper extends the study first presented by Zwick and Shervais in 2002. Here, after describing the schema and building block hypotheses and their relevance to epistasis (Section II), we further demonstrate the existence of a gene order effect in a set of simple problems, and show that the effect depends on the stage of the search process (Section III). We then show that the methodology of reconstructability analysis can be used to discover preferred gene orders from data obtained by sampling the fitness function (Section IV). Finally, we discuss the results of these preliminary experiments and point to areas for future exploration (Section V).

II. SCHEMA THEOREM, BUILDING BLOCKS, AND EPISTASIS

The schema theorem was first proposed by Holland (1975) as a description of how adaptive systems "persistently test and incorporate structural properties associated with better performance (p. 66)." Although there is now some doubt as to how well it describes the dynamics of the GA search process (Thornton, 1997; Mitchell, 1995), it is still useful as a conceptual device, and we use it that way here. According to the schema theorem, GAs work by parallel testing of multiple combinations of bit strings made up of the available alleles. In the typical binary chromosome, the alleles may be represented as 1, 0, and * (don't care). Thus, 110***11 is a schema (call it S1) of defining length eight and cardinality five. Note that S1 contains a shorter schema (S2), 110*****, with a defining length and cardinality of three, and a third schema (S3), ******11, with a defining length and cardinality of two. In fact, a schema of defining length eight has 3^8 possible schemata embedded in it, but we here discuss just these three.

If strings containing S2 have a higher-than-average fitness, they will be preferentially selected, and S2 will act as a *building block* that can be assembled with other building blocks to create longer schemata and higher fitness bitstrings. Since the ratio of the defining length to the cardinality is low, S2 is not likely to be broken up by the crossover operator. The same argument applies to S3. Now consider S1. If bitstrings containing this schema have a higher than average fitness, they will be preferentially selected as well. However, since the defining length of S1 is large relative to its cardinality, it also stands a higher chance of being broken up during crossover. If S2 and S3 are both important to the fitness of S1, we would be better off changing the representation so that S2 and S3 are close together. In other words, if S1 has high fitness, it would be more likely to survive recombination if we had some good reason to move the 11 alleles over to be adjacent to the 110 alleles, i.e., to recode the genome so that this schema was 11011***. Note that this will be the case even if the fitness contribution of the

S2 and S3 schemas are independent of each other, but it will be especially true if the fitnesses of S2 and S3 interact in some way, that is, are epistatic. This paper confines its arguments to epistatic genes, which are discussed further, below.

Although the usefulness of short building blocks has long been understood, only a few researchers have addressed the issue of how changing gene order might facilitate reaching enhanced fitness. Barbara McClintock is credited with discovering the importance of gene transposition in nature (McClintock, 1987). Transposition is thus available as a possible genetic operator available for use by GA researchers. Simoes and Costa (1999) examined the usefulness of McClintock's transposons as a replacement for the crossover operator. In their work, randomly selected runs of bitstrings were moved about on the chromosome. While the study displayed the effectiveness of the transposition operator, the experimenters reported no data that would allow identification of the most effective bitstring orders..

Goldberg, et al. (1993) developed the "fast messy" GA, which, among other things, allows the GA to evolve gene locations on the chromosome. They did this by coding stretches of the chromosome with a gene identifier, which specified the gene which that part of the chromosome represents. A given gene might start out over-expressed in a chromosome, because its identification code appears at two different locations. The program selects the first instance of the gene and ignores the rest. Alternatively, a gene might be under-expressed if it does not appear in the bitstring at all. The program then applies a default template to supply the missing gene values. As evolution proceeds, and the length of the GA is allowed to change from long to short and back to long again, those bitstrings with efficient gene orders will be preferentially selected. The difficulty with this approach is that the search for an efficient gene order proceeds in parallel with the search for an optimal solution.

In a much different approach, Beasley et al. (1993) used *a priori* knowledge to code interactive genes into *sub-problems*, which are subject to separate evolutionary processes and are recombined each generation. This requires that some exogenous process identify the sub-problems.

The impact of one gene on the fitness contribution of another is called *epistasis.* (In other contexts, this might be called an "interaction effect" or might be said to exemplify "synergy.") In the schema S1 discussed above, assume that the high fitness of S1 derives from an epistatic interaction between S2 and S3, and not merely from the separate high fitnesses of these two schemas. This would be all the more reason for S2 and S3 to be adjacent to one another and constitute a compact building block. The matter might be more complex, however. While the tight coupling of high epistatic genes into building blocks might seem at first glance to be an unalloyed good, further reflection shows the advantage of repositioning genes on our illustrative chromosome accrues only *after* the good 110 and 11 alleles first occur on the genome, after which preservation of these alleles as a building block becomes advantageous. A different, indeed *opposite*, argument might apply to the process of searching for high fitness schemata. During the early generations, the GA is still searching for good combinations of alleles, and crossover is the primary tool for searching out novel combinations. If the 110 and 11 alleles exist on two different parental chromosomes, they are more likely to be recombined as the result of crossover if the genes are distant from one another.

We thus have an argument for epistatic genes being close together, and a second argument for their being far apart.

III. DEMONSTRATING GENE ORDER EFFECTS IN A GENETIC ALGORITHM

In Zwick and Shervais (2002) we demonstrate the possibility of a gene order effect by using an extremely simple fitness function, namely the function (to be maximized) specified by the equation

$$F = \min(A/B, B/A) * C,$$

where A, B, and C take on values between 0 and 3.0. The minimization operation constrains the min(A/B, B/A) expression to values less than or equal to 1.0 and fitness, F, to the range 0.0 to 3.0. The epistatic nature of the problem arises from the fact that the AB term is maximized (at 1.0) only if A and B are equal. The C variable has no impact on the AB term, and contributes to overall fitness in simple proportion to its value.

For the current paper, we looked at additional problems of slightly greater complexity:

Maximize $F_1 = \min(A/B, B/A) + \min(C/D, D/C)$ (1+)

Maximize $F_2 = \min(A/B, B/A) * \min(C/D, D/C)$ (1*)

Minimize $F_3 = 100(A^2-B)^2 + (1-A)^2 + 100(C^2-D)2 + (1-C)^2$, (2+)

Minimize $F_4 = 100(A^2-B)^2 + (1-A)^2 * 100(C^2-D)^2 + (1-C)^2$ (2*).

The two variants of equation (1), which we will abbreviate as the AB+CD and AB*CD problems, are a simple extension of the original problem wherein the variable pair AB and the variable C were to be optimized independently of each other. These are maximization problems, since minimization only requires that A and C go to zero. The problem presented in equation (2) is a set of linked DeJong F2 functions (for a summary of GA benchmark functions, see Digalakis and Margaritis,

2000) , The DeJong F2 function is also known as Rosenbrock's valley and the Banana Function. The last name comes because the two-dimensional minimum lies in a narrow, banana-shaped fitness valley. Our experiments in this series linked two of the F2 functions with the addition operator and with the multiplication operator, so that variables AB optimized the first function, and CD optimized the second. again requiring that variable pairs AB and CD be optimized independently of each other. The two problems are abbreviated as F2+F2 and F2*F2, and this time the GA is seeking the minimum. The equations are numbered to indicate plainly that in (1+) and (2+), the independence of A, B and C, D is additive, while in (1*) and (2*) it is multiplicative. The common denominator of these two different types of independence is the *separability* of the fitness function.

Focusing exclusively on the imperative of retaining (as opposed to finding) good building blocks, one would expect that, for all of the above fitness functions, a chromosome where A and B were side by side in one place and C and D were side by side in another, would allow the GA to perform more effectively than one where the variables were separated with A and B separated by some combination of C and D. Table 1. shows the twelve different ways (ignoring reflections, e.g. ACBD includes also DBCA) of ordering the four variables on the chromosome. There are four orders where both pairs are *adjacent*. The four orders where no pairs are adjacent, and the four with one pair adjacent and one pair separate, have been designated as *separated*. The question is, which is more efficient, the chromosomes with epistatic genes being adjacent, or the chromosomes where they are separated.

TABLE 1. POSSIBLE GENE ORDERINGS FOR FOUR VARIABLES.

Adjacent	*Separated*	
ABCD	ACBD	ACDB
ABDC	CADB	CABD
BACD	BCAD	BCDA
BADC	CBDA	CBAD

In the orderings labeled *Adjacent*, both pairs of epistatic genes are next to each other. In the *Separated orderings*, there is at least one gene pair that is not adjacent to one another. Note that each of the above orders includes also its inverse, e.g., ABCD includes also DCBA and ACBD includes also DBCA.

Our hypotheses are:

Hypothesis 0: Gene position has no impact on the effectiveness of the GA search, that is, there is no difference in average effectiveness between the adjacent and separated orders.

Hypothesis I: For the early phases of GA search, the separated orders will be more effective.

Hypothesis II: For the late phases of GA search the adjacent orders will be more effective.

Shervais, S. and Zwick, M. (2003). "Ordering Genetic Algorithm Genomes With Reconstructability Analysis." International Journal of General Systems, Vol. 32(5), pp. 491-502

The Genetic Algorithm we used employed standard binary encoding, with 8-bit genes. The GA parameters for all experiments included: breeding population 30, generations 30, mutation rate 0.01, crossover rate 1.0, and repetitions 100, with a new random seed for each repetition. Crossover was single point, and occurred at any place on the chromosome.

Results of GA runs with the different orders follow. First we show these results graphically, and the graphs visually confirm Hypothesis II very clearly, but support Hypothesis I only weakly. We then report statistical tests which show that the support for Hypothesis I, however weak, is usually statistically significant.

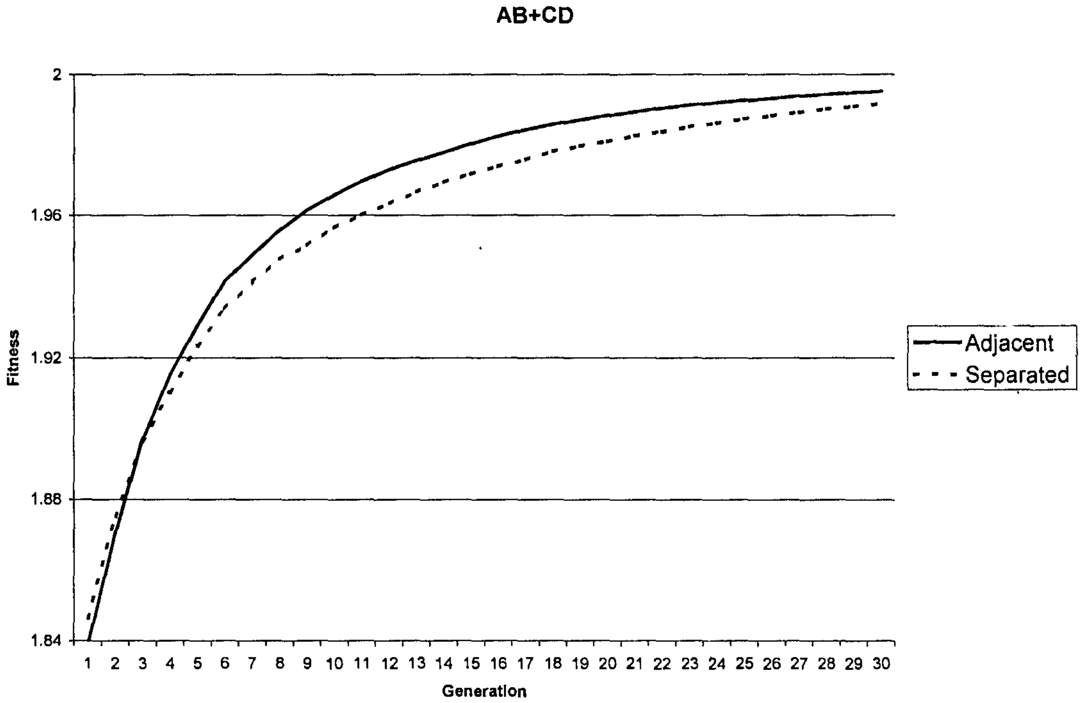

FIGURE 1. EFFECTIVENESS OF THE GA ON THE AB+CD PROBLEM: Effectiveness of adjacent orders, where linked genes AB and CD are adjacent, is compared to effectiveness of separated orders, where no linked genes are adjacent, for equation (1+).

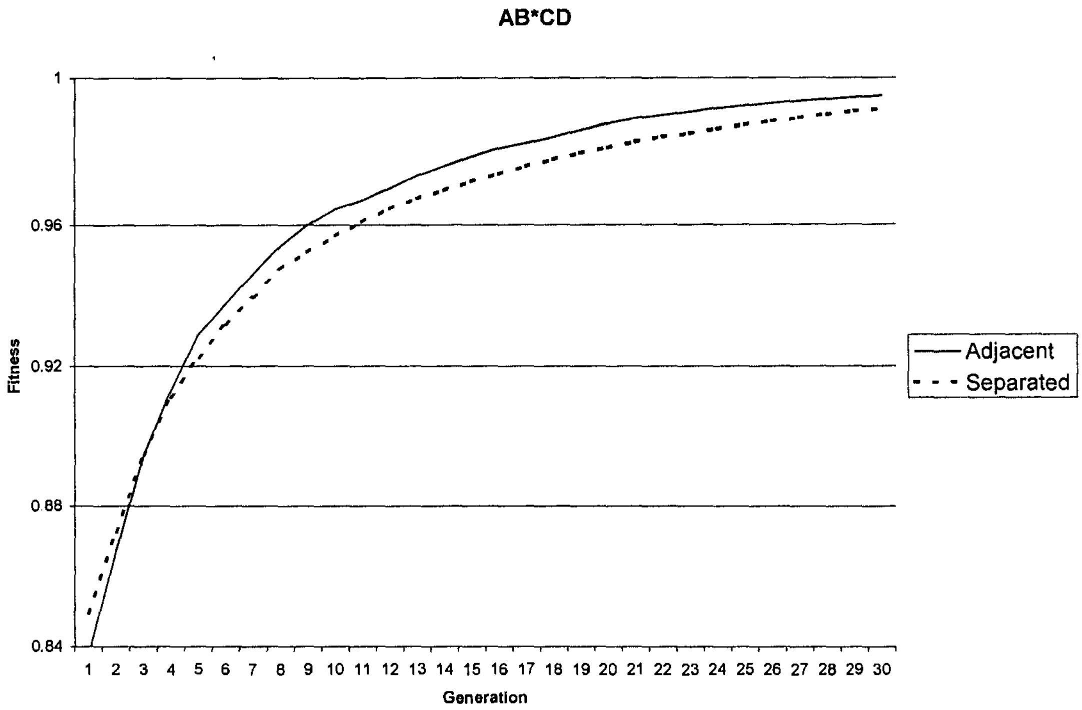

FIGURE 2. GA EFFECTIVENESS ON THE AB*CD PROBLEM. Effectiveness of adjacent orders, where linked genes AB and CD are adjacent, is compared to effectiveness of separated orders, where no linked genes are adjacent, for equation (1*).

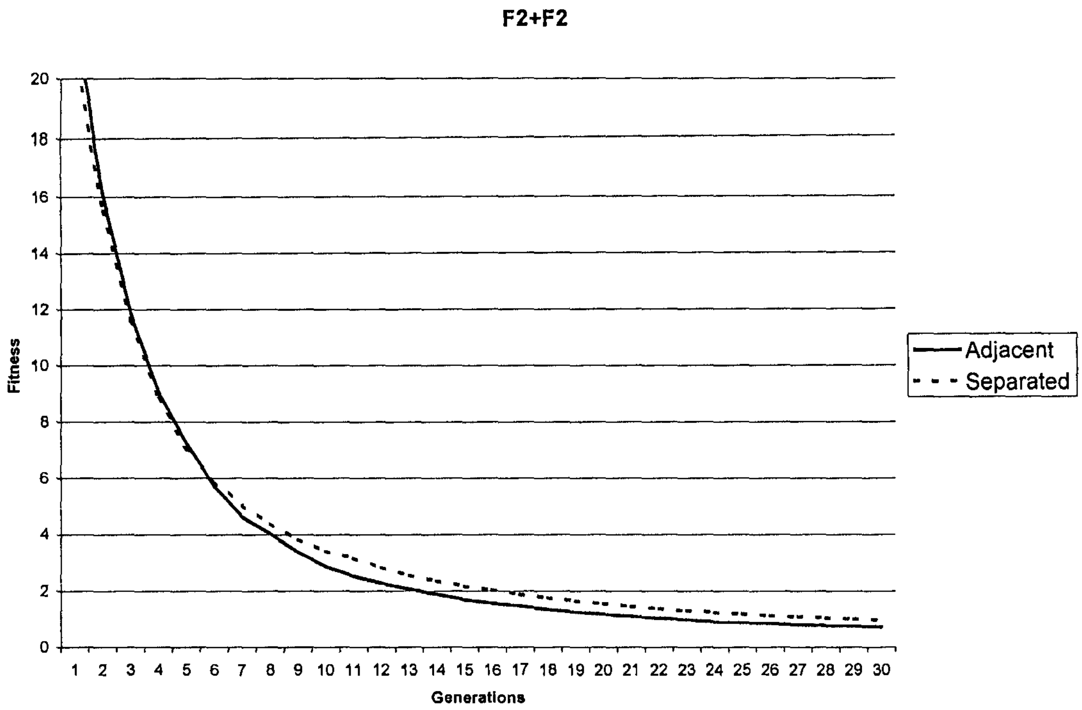

FIGURE 3. PERFORMANCE OF THE GA ON THE F2+F2 PROBLEM. Effectiveness of adjacent orders, where linked genes AB and CD are adjacent, is compared to effectiveness of separated orders, where no linked genes are adjacent, for equation (2+).

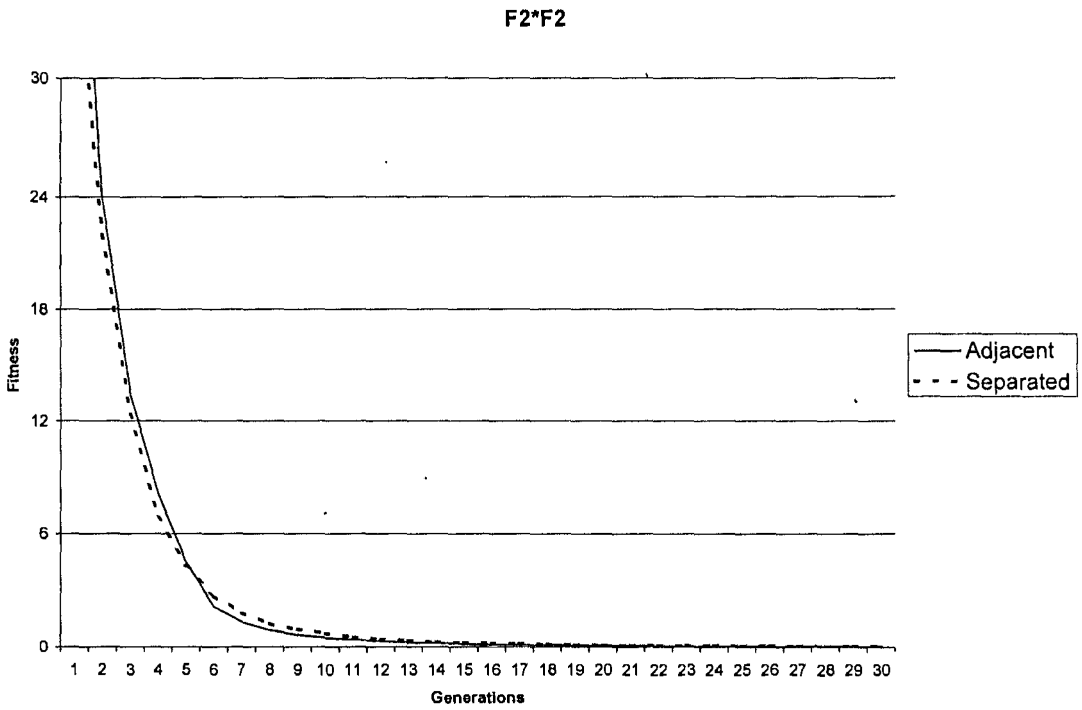

FIGURE 4. PERFORMANCE OF THE GA ON THE F2*F2 PROBLEM. Effectiveness of adjacent orders, where linked genes AB and CD are adjacent, is compared to effectiveness of separated orders, where no linked genes are adjacent, for equation (2*).

The figures all display similar characteristics. With such simple problems, the GA approaches the solutions quite rapidly. Visually, the separated orders appear to perform better than the adjacent orders in the very early generations. The performance curves cross somewhere around generations three to five, and from then on, the adjacent orders show a better performance than the separated orders. Since we are predicting the direction of the difference, we can test our hypotheses using a single-tailed t-test to compare the means. The details of this performance difference for generation 1 are shown in Table 2, and those for generation 30 are shown in Table 3.

TABLE 2. GENE ORDER PERFORMANCE AT GENERATION 1

	AB+CD	AB*CD	F2+F2	F2*F2
Mean Adjacent (n = 400)	1.83	0.843	22.28	43.47
Mean Separated (n=800)	1.84	0.845	20.97	37.48
Std Dev Adjacent	0.086	0.077	16.87	77.61
Std Dev Separated	0.081	0.08	17.73	53.14
t-value	-1.55	-0.45	1.23	1.57
p-value	0.0606	0.3247	0.1096	0.058

Performance of adjacent vs. separated gene orders at the start of the experimental run. In every case, the separated orders perform better than the adjacent orders (recall that the left two functions are being maximized while the right two functions are being minimized). This performance is statistically significant at a better than 0.10 level in two cases, almost reaches that level of significance in one case, and fails the significance test in the fourth.

TABLE 3. GENE ORDER PERFORMANCE AT GENERATION 30.

	AB+CD	AB*CD	F2+F2	F2*F2
Mean Adjacent	2.00	1.00	0.71	0.032
Mean Separated	1.99	0.99	1.02	0.045
Std Dev Adjacent	0.005	0.004	0.81	0.10
Std Dev Separated	0.007	0.008	0.97	0.11
t-value	8.61	8.12	-4.41	-2.31
p-value	<0.0001	<0.0001	<0.0001	0.0105

Performance of adjacent vs. separated gene orders at the end of the experimental run. In every case, the separated orders were more efficient than the adjacent orders, significant at a 0.01 level or better.

The experiments described above provide good support for Hypothesis II, and weak support for Hypothesis I. In generation 1, in every case, the separated orders perform better than the adjacent orders. This performance is statistically significant at a better than 0.10 level in two cases, almost reaches that level of significance in one case, and fails the significance test in the fourth. The most we can say is that there is weak support for Hypothesis I. Late in the search process, the separated orders outperform the adjacent orders in all experiments, and at generation 30 this difference is statistically significant; in three cases at the 0.0001 level, and in one case at the 0.01 level, providing strong support for Hypothesis II.

Shervais, S. and Zwick, M. (2003). "Ordering Genetic Algorithm Genomes With Reconstructability Analysis." International Journal of General Systems, Vol. 32(5), pp. 491-502

IV. DETECTING OPTIMAL GENE ORDER BY RECONSTRUCTABILITY ANALYSIS

If gene order matters, it should be useful to find out what the optimum gene order is. In this section we show that this determination is achievable, at least for the cases examined here, using the methods of reconstructability analysis..

Reconstructability Analysis

Reconstructability analysis (RA) derives from Ashby (1964), and was developed by Broekstra, Cavallo, Cellier, Conant, Jones, Klir, Krippendorff, and others; an extensive bibliography is available in (Klir, 1986), and a compact summary of RA is available in (Zwick, 2001a; Zwick, 2002). RA resembles log-linear (LL) methods (Bishop et al, 1978; Knoke & Burke, 1980), used widely in the social sciences, and where RA and LL methodologies overlap they are equivalent (Knoke & Burke, 1980; Krippendorff, 1986). In RA (Klir, 1985), a probability or frequency distribution or a set-theoretic relation is decomposed (compressed, simplified) into component distributions or relations. The most common approach is typically the decomposition of frequency or probability distributions, where RA does statistical analysis. RA can model problems both where "independent variables" (inputs) and "dependent variables" (outputs) are distinguished (*directed* systems) and where this distinction is not made (*neutral* systems).

In the present case, we have a directed system, with inputs A, B, C and D, and output the fitness value, F. Consider an observed frequency distribution g(A, B, C, D, F) which we write more simply as ABCDF. RA decomposes such distributions into sets of projections such as ABCD:ABF:CDF and these models are assessed for statistical significance, usually with the Chi-square distribution. Taken together, the linked projections constitute a model of the data which is less complex (has fewer degrees of freedom) than the data. By maximum-entropy (uncertainty) composition of these projections, the model yields a *calculated* distribution $ABCDF_{ABCD:ABF:CDF}$ (the subscripts show the model used) which may differ from the observed ABCDF. The difference (error) represents loss of information in the model. By definition, the data itself, ABCDF, which asserts that A,B,C,D predict F, has 100% information (0% error). The "independence model," ABCD:F, which asserts that nothing predicts F, has 0% information.

One can distinguish between different classes of RA models Here we consider two classes: "disjoint models" and "chain models" (Krippendorff, 1986). Models are defined as a set of components, a "non-predicting" component which involves only the inputs (IVs), and one or more "predicting components" involving the DV. For simplicity, the non-predicting component (in the present case, ABCD) will from now on be omitted from the model specification. In disjoint models, which Krippendorf calls *regression models,* the inputs in different predicting components do not overlap, as in ABF:CDF, while in chain models, every component overlaps the next component in one variable, as in ABF:BCF:CDF. The disjoint model ABF:CDF means that A and B together predict F, and C and D also together predict F, and that these two predicting components are merged via the maximum entropy formalism of RA. The chain model ABF:BCF:CDF means that A and B predict F, as do B and C, as do C and D, and that the three predicting components are merged via the maximum entropy formalism.

We will use these disjoint and chain models to decompose the fitness function, and the information content of the various models will reveal the relative merits of adjacent versus separated orders of the GA chromosome. To connect model specification and gene order, note that the sequence of the components in the model specification is arbitrary and that the sequence of variables in a component is also arbitrary. Thus the disjoint model, ABF:CDF, is consistent with gene orders ABCD, ABDC, BACD, and BADC (and their inverses), i.e., with all four adjacent orders. This disjoint model is also inconsistent with all eight separated orders. By contrast the disjoint model ABCF:DF is consistent *both* with ABCD and DABC, and thus cannot be used to evaluate the relative merits of adjacent vs. separated orders. Chain models are more specific: every chain model is consistent with only one gene order (and its inverse). For example, the chain model ABF:BCF:CDF is consistent only with ABCD (and DCBA).

RA Calculations

Calculations were made using the RA software programs developed at Portland State University, now integrated into the package OCCAM (for the principle of parsimony and as an acronym for "Organizational Complexity Computation And Modeling"). The earliest of these programs was developed by Zwick and Hosseini (Hosseini, Harmon, & Zwick, 1986); a review of RA methodology is offered in (Zwick, 2001a; Zwick, 2002); a list of recent RA papers in the PSU group is given in (Zwick, 2001b). A description of the OCCAM architecture is given in (Willett & Zwick, 2002).

For this paper, the reconstructability analysis was conducted on datasets generated by random sampling of the search space. We uniformly sampled the continuous values of the four variables A, B, C, and D, over an appropriate range, and used the resulting sets of independent variables to calculate the dependent variable, F(A,B,C,D). Approximately 7200 unique combinations of A, B, C, D, and F were created. All five variables were then each binned into four bins. The datasets were analyzed by the *OCCAM* software package and the results are shown in Table 4.

TABLE 4. RECONSTRUCTABILITY ANALYSIS OF THE FOUR FITNESS FUNCTIONS

AB+CD			ABxCD			F2+F2			F2xF2		
MODEL		INF.	MODEL		INF.	MODEL		INF.	MODEL		INF.
DISJOINT			**DISJOINT**			**DISJOINT**			**DISJOINT**		
ABF:CDF	a	0.672	ABF:CDF	a	0.631	AF:BCDF		0.845	ABF:CDF	a	0.736
ACDF:BF		0.540	ABDF:CF		0.556	ABDF:CF		0.809	ABDF:CF		0.700
ABDF:CF		0.527	ABCF:DF		0.551	ABF:CDF	a	0.786	AF:BCDF		0.662
AF:BCDF		0.526	AF:BCDF		0.528	ACDF:BF		0.768	ABCF:DF		0.625
ABCF:DF		0.518	ACDF:BF		0.526	ABCF:DF		0.732	ACDF:BF		0.605
AF:BF:CDF		0.416	ABF:CF:DF		0.424	AF:BF:CDF		0.710	ABF:CF:DF		0.555
ABF:CF:DF		0.407	AF:BF:CDF		0.401	ABF:CF:DF		0.683	AF:BF:CDF		0.521
ADF:BCF	s	0.255	ACF:BDF	s	0.290	ADF:BCF	s	0.671	ADF:BCF	s	0.441
ACF:BDF	s	0.243	ADF:BCF	s	0.284	ACF:BDF	s	0.654	ACF:BDF	s	0.438
ADF:BF:CF		0.233	AF:BDF:CF		0.268	ADF:BF:CF		0.653	AF:BDF:CF		0.421
AF:BDF:CF		0.229	ACF:BF:DF		0.263	AF:BDF:CF		0.637	ADF:BF:CF		0.418
AF:BCF:DF		0.228	AF:BCF:DF		0.262	AF:BCF:DF		0.636	AF:BCF:DF		0.417
ACF:BF:DF		0.221	ADF:BF:CF		0.262	ACF:BF:DF		0.629	ACF:BF:DF		0.407
AF:BF:CF:DF		0.205	AF:BF:CF:DF		0.237	AF:BF:CF:DF		0.613	AF:BF:CF:DF		0.389
CHAIN			**CHAIN**			**CHAIN**			**CHAIN**		
ABF:BCF:CDF	a	0.695	ABF:BCF:CDF	a	0.662	ABF:ADF:CDF	a	0.811	ABF:ADF:CDF	a	0.778
ABF:ADF:CDF	a	0.693	ABF:BDF:CDF	a	0.658	ABF:BCF:CDF	a	0.809	ABF:BDF:CDF	a	0.757
ABF:ACF:CDF	a	0.687	ABF:ACF:CDF	a	0.657	ABF:BDF:CDF	a	0.806	ABF:BCF:CDF	a	0.756
ABF:BDF:CDF	a	0.686	ABF:ADF:CDF	a	0.656	ABF:ACF:CDF	a	0.802	ABF:ACF:CDF	a	0.754
ADF:BCF:CDF	s	0.462	ABF:ACF:BDF	s	0.475	ADF:BCF:CDF	s	0.758	ABF:ADF:BCF	s	0.605
ACF:BDF:CDF	s	0.452	ABF:ADF:BCF	s	0.469	ACF:BDF:CDF	s	0.751	ABF:ACF:BDF	s	0.604
ABF:ADF:BCF	s	0.449	ACF:BDF:CDF	s	0.449	ABF:ADF:BCF	s	0.743	ADF:BCF:CDF	s	0.585
ABF:ACF:BDF	s	0.444	ADF:BCF:CDF	s	0.448	ABF:ACF:BDF	s	0.720	ACF:BDF:CDF	s	0.570
ADF:BCF:BDF	s	0.281	ACF:ADF:BDF	s	0.319	ACF:ADF:BCF	s	0.699	ADF:BCF:BDF	s	0.474
ACF:ADF:BDF	s	0.276	ADF:BCF:BDF	s	0.316	ACF:ADF:BDF	s	0.695	ACF:ADF:BDF	s	0.472
ACF:ADF:BCF	s	0.273	ACF:BCF:BDF	s	0.315	ADF:BCF:BDF	s	0.692	ACF:ADF:BCF	s	0.467
ACF:BCF:BDF	s	0.266	ACF:ADF:BCF	s	0.310	ACF:BCF:BDF	s	0.679	ACF:BCF:BDF	s	0.466

Shown are the information content (INF) for all disjoint models which include all inputs and for all chain models. Next to the models, “a” is written if the model is consistent only with an adjacent order, and “s” is written if the model is consistent only with a separated order. If neither “a” nor “s” is indicated, the model is compatible with both adjacent and separated orders, and thus cannot be used to discriminate between the two. The table omits (for simplicity) the ABCD component (present in all these models) from the name of the model, e.g., disjoint model ABF:CDF is really model ABCD:ABF:CDF

The OCCAM computations are essentially as follows. For each sample we have an observation of a four-input, one-output directed system (A_i, B_j, C_k, D_l, F_m). The information in a given model is calculated as

INF (test model),= 1 - [T(test model)/T(independence model)] (3),

where I is information content, T is transmission, the independence model is ABCD:F, and the test model is a disjoint or chain model, as indicated in Table 4. The transmission calculation is

T(test model) = $\sum_{ijklm}$ p_{ijklm} log (p_{ijklm} / q(test model)$_{ijklm}$) (4)

where $p_{ijklm} = f_{obs}(A_i, B_j, C_k, D_l, F_m)/N$,

q(test model)$_{ijklm}$ = $f_{calc(test\ model)}(A_i, B_j, C_k, D_l, F_m)/N$,

N = sample size,

f_{obs} = observed frequency, and

f_{calc}(test model) = calculated frequency.

The f_{calc} for a test model is obtained by iteratively applying the iterative proportional fitting (IPF) algorithm using the projections specified by the model. For example, the f_{calc} for disjoint model ABCD:ABF:CDF is obtained by iteratively imposing, via IPF, the ABCD, ABF, and CDF f_{obs} projections on an initially uniform f_{calc} distribution.

Table 4 shows that, for the fitness functions used in this study, RA can unambiguously ascertain how these fitness functions depend on the four input variables and clearly identifies the AB and CD building blocks of these dependencies. For all four fitness functions, the best chain models favor the adjacent orders. For three out of four fitness functions, the model ABF:CDF was the best disjoint model, again showing that the adjacent rather than the separated orders reflect the functional dependence of fitness on the inputs; for the fourth fitness function, ABF:CDF was not the best disjoint model but was plainly superior to other disjoint models which favor a separated order. In summary, knowing that a GA will perform better with adjacent rather than separated orders, and given sampled data on the fitness function, RA can tell us how to optimally order the input variables on the GA chromosome, at least for the fitness functions examined in this study.

V. DISCUSSION

For the simple test problems shown, constructed to depend on the interaction of epistatic genes, the adjacent orders (those that kept epistatic genes together) ultimately found better solutions faster than did orders that separated the epistatic genes. The gene order effect was small, but in more complex problems, it may become more substantial. The relative effectiveness of the two sets of orders changed throughout the experiments. At the beginning -- roughly the first five generations -- the separated orders pęrformed as well or better than the good ones. For the next twenty generations the adjacent orders performed better. In the end game, when both were approaching the solution asymptotically, the separated orders slowly caught up, but were still behind the performance of the adjacent orders at the end.

This behavior of the GA is comprehensible. Specifically, one expects that in the early phases of a GA run, epistatically linked genes should best be located far from one another, since at the beginning of the search it may be useful for all genes to be mixed as much as possible by the crossover operator. If two epistatic genes are side-by-side from the beginning, then crossover would have less chance of improving them, and the GA would have to depend upon a good initialization and fortunate mutations to create the best gene pair possible. If, on the other hand, two epistatic genes start out well separated, the crossover operation might more easily assemble a larger selection of allele patterns in the two genes. The early-separated effect is exhibited in all the experiments (Table 2), but the effect is small. One might speculate that its impact is problem-dependent.

Reconstructability analysis allows one to find the models that retain high information about the data. Using both disjoint and chain models, one can obtain an indication of how to order the variables on the GA chromosome. Both chain and disjoint models suggested that the adjacent orders would be better than the separated orders. Moreover the use of disjoint models might be a way to solve Beasley et al.'s problem of *a priori* identification of subproblems for expansive coding. The top model, ABCDF, includes interactions among all variables, but Table 4 shows that ABF:CDF has 70-80% of the information, so solving the ABF and CDF subproblems separately and merging the answers would probably give a good result. Using RA to decompose optimization problems into subproblems might of course also be useful for optimization methods other than the GA.

Future Research

The primary reason for obtaining an optimum gene order is to improve the effectiveness of the GA: either to allow it to find a good solution faster, or to find a better solution in the same time. The examples considered here are too simple for RA to yield a gain in GA efficiency in any practical sense. This paper is just a demonstration of concept. It remains for future work to show a practical benefit from RA preprocessing. This may arise in GA analyses of more complex problems which require much longer runs, where RA analysis of a sample of the fitness function might be repaid by superior GA performance. It may even be possible to restructure the genome of a GA based on data that the GA itself generates. This might speed up processing in a particular GA run, if the optimum order can be detected early enough for the GA to gain an advantage from gene reordering. It may also be possible to generalize the structure of a GA so that it is optimized for a wide range of problems of the same specific class. To use a real-world example, one of the authors recently studied the use of a GA to solve an inventory and distribution problem (Shervais, 2000a, 2000b). One would expect that any set of such problems with the identical number and structure of nodes and stocks could use information generated by the first specific problem to be addressed. These topics are subjects for future research.

VI. BIBLIOGRAPHY

Ashby, W. R. (1964) "Constraint Analysis of Many-Dimensional Relations." *General Systems Yearbook*, 9, pp. 99-105.

Beasley, D., D. Bull, and R. Martin (1993), "Reducing Epistasis in Combinatorial Problems by Expansive Coding." In: *Proceedings of the Fifth International Conference on Genetic Algorithms*, Morgan Kaufman Publishers, San Mateo, pp. 400-407.

Bishop, Y., S. Feinberg, and P. Holland (1978) *Discrete Multivariate Analysis.* (MIT Press, Cambridge, MA).

Digalakis, J. and Margaritis, K, (2000), "An experimental study of benchmarking functions for Genetic Algorithms" in Proceedings of the IEEE International Conference On Systems, Man & Cybernetics, Nashville, TN, October 2000, pp. 3810-3815.

Goldberg, D., K. Deb, H. Kargupta, and G. Harik (1993) *Rapid, Accurate Optimization of Difficult Problems Using Fast Messy Genetic Algorithms.* IlliGAL Report No. 93004, University of Illinois.

Hosseini, J.C., R. R. Harmon, and M. Zwick (1986), "Segment Congruence Analysis Via Information Theory," in *Proceedings, International Society for General Systems Research*, Philadelphia, PA, May 1986, pp. G62 - G77.

Holland, J. (1975) *Adaptation in natural and artificial systems.* (University of Michigan Press, Ann Arbor, MI.)

Klir, G. (1985) *The Architecture of Systems Problem Solving.* (Plenum Press, New York, NY.)

Klir, G. (1986) "Reconstructability Analysis: An Offspring of Ashby's Constraint Theory", *Systems Research,* 3 (4), pp. 267-271.

Knoke, D. and P.J. Burke (1980) *Log-Linear Models.*(Quantitative Applications in the Social Sciences Monograph # 20. (Sage, Beverly Hills, CA.)

Krippendorff, K. (1986) "Information Theory: Structural Models for Qualitative Data." Quantitative Applications in the Social Sciences #62. (Sage, Beverly Hills, CA.)

McClintock, B. (1987) *The discovery and characterization of transposable elements: the collected papers of Barbara McClintock,* (Garland, New York, NY.)

Mitchell, M. (1996) *An Introduction to Genetic Algorithms.* (MIT Press, Cambridge, MA.)

Shervais, S. (2000a) "Developing Improved Inventory And Transportation Policies For Distribution Systems Using Genetic Algorithm And Neural Network Methods", in: *Proceedings of the World Conference on the Systems Sciences,* Toronto, Canada, pp. 200059-1 to 200059-17.

Shervais, S. (2000b) *Adaptive Critic Design of Control Policies for A Multi-Echelon Inventory System. Ph.D. dissertation, Portland State University. Available at:* http://www.cbpa.ewu.edu/~sshervais/Personal_Info/Papers/Dissertation.ps.zip

Simoes, A. and E. Costa (1999) "Transposition: {A} Biologically Inspired Mechanism to Use with Genetic Algorithms", In: *Proceedings of the Fourth International Conference on Neural Networks and Genetic Algorithms ({ICANNGA} 99),* Portoroz, Slovenia. *Springer Verlag, Berlin, pp. 178-186.*

Shervais, S. and Zwick, M. (2003). "Ordering Genetic Algorithm Genomes With Reconstructability Analysis." International Journal of General Systems, Vol. 32(5), pp. 491-502

Stadler, P., R. Seitz, and G.P. Wagner (2000) "Population Dependent Fourier Decomposition of Fitness Landscapes over Recombination Spaces: Evolvability of Complex Characters", *Bulletin of Mathematical Biology*, Vol. 62, No. 3, pp. 399-428.

Thornton, C. (1997) "The building block fallacy." *Complexity International* 4. http://www.csu.edu.au/ci/vol04/thornton/building.htm

Willett, K., and Zwick, M. (2002) "A Software Architecture for Reconstructability Analysis", in Proceedings of 12th International World Organization of Systems and Cybernetics and 4th International Institute for General Systems Studies Workshop, Pittsburgh, March 24-26. http://www.sysc.pdx.edu/download/papers/softarcabstract.htm

Zwick, M. (2001a) "Wholes and Parts in General Systems Methodology", in Wagner, G.,ed., *The Character Concept in Evolutionary Biology*. (Academic Press, New York) pp. 237-256. http://www.sysc.pdx.edu/faculty/Zwick/research.html#wholes

Zwick, M. (2001b) "Discrete Multivariate Modeling": http://www.sysc.pdx.edu/res_struct.html

Zwick, M. and Shervais, S. (2002) "Reconstructability Analysis Detection Of Optimal Gene Order In Genetic Algorithms", in *Proceedings of 12th International World Organization of Systems and Cybernetics and 4th International Institute for General Systems Studies Workshop*, Pittsburgh, March 24-26.

Zwick, M. (2002) "An Overview of Reconstructability Analysis", in *Proceedings of 12th International World Organization of Systems and Cybernetics and 4th International Institute for General Systems Studies Workshop*, Pittsburgh, March 24-26. http://www.sysc.pdx.edu/download/papers/ldlpitfabstract.htm

VII. BIOGRAPHIES

Martin Zwick is a Professor of Systems Science at Portland State University. Prior to taking his current position at PSU, he was a faculty member in the Department of Biophysics and Theoretical Biology at the University of Chicago, where he worked in macromolecular structure and mathematical crystallography. In the 1970's his interests shifted to systems theory and methodology. Since 1976 he has been on the faculty of the PSU Systems Science Ph.D. Program and during the years 1984-1989 he was the program head. His current research interests are in discrete multivariate modeling (reconstructability analysis), "artificial life" and theoretical/computational biology, and systems philosophy.

Steve Shervais is an Assistant Professor of Management Information Systems in the Accounting and Information Systems Department of Eastern Washington University . He was awarded his Ph.D. in Systems Science by Portland State University in 2000. Prior to his shift to academia he was an intelligence analyst in the USAF, and a database developer in industry. His main area of research is the application of systems methodologies to the solution of business problems.

Shervais, S. and Zwick, M. (2003). "Ordering Genetic Algorithm Genomes With Reconstructability Analysis." International Journal of General Systems, Vol. 32(5), pp. 491-502