Self-Organized Modularization in Evolutionary Algorithms

Peter Dauscher
dauscher@informatik.uni-mainz.de
Department of Computer Science, Johannes Gutenberg-Universität Mainz, Germany

Thomas Uthmann
uthmann@informatik.uni-mainz.de
Department of Computer Science, Johannes Gutenberg-Universität Mainz, Germany

Abstract

The principle of modularization has proven to be extremely successful in the field of technical applications and particularly for Software Engineering purposes. The question to be answered within the present article is whether mechanisms can also be identified within the framework of Evolutionary Computation that cause a modularization of solutions. We will concentrate on processes, where modularization results only from the typical evolutionary operators, i.e. selection and variation by recombination and mutation (and not, e.g., from special modularization operators). This is what we call Self-Organized Modularization.

Based on a combination of two formalizations by Radcliffe and Altenberg, some quantitative measures of modularity are introduced. Particularly, we distinguish Built-in Modularity as an inherent property of a genotype and Effective Modularity, which depends on the rest of the population. These measures can easily be applied to a wide range of present Evolutionary Computation models.

It will be shown, both theoretically and by simulation, that under certain conditions, Effective Modularity (as defined within this paper) can be a selection factor. This causes Self-Organized Modularization to take place. The experimental observations emphasize the importance of Effective Modularity in comparison with Built-in Modularity. Although the experimental results have been obtained using a minimalist toy model, they can lead to a number of consequences for existing models as well as for future approaches.

Furthermore, the results suggest a complex self-amplification of highly modular equivalence classes in the case of respected relations. Since the well-known Holland schemata are just the equivalence classes of respected relations in most Simple Genetic Algorithms, this observation emphasizes the role of schemata as Building Blocks (in comparison with arbitrary subsets of the search space).

Keywords
Evolutionary Algorithms, modularity, modularization, linkage learning, genotype-phenotype-map, schemata, schema-theorem

1 Introduction

Modularization is one of the key concepts in mechanical and software engineering. However, many structures of a rather modular character (e.g. the existence of cells, tissues, organs etc.) can also be observed in biological systems.

In Evolutionary Computation research, many models deal with the question about the formation of complex or modular structures by evolutionary forces. Evolving modularity is frequently considered as one of the crucial ingredients for further research (Mitchell, 1996).
Approaches where modularity does not have to be explicitly supported (e.g., by higher fitness values or extra operators), but where it results from the normal evolutionary dynamics, are particularly interesting. This is what we refer to as Self-Organized Modularization.

However, most investigations refer to a rather specific model framework (as bit-oriented Simple Genetic Algorithms, Genetic Programming or Neuro-Genetic approaches) and it is hard (if not impossible) to compare these different frameworks with one another.

This paper, summarizing some theoretical and simulation results obtained in a more extended context (Dauscher, 2003), has two main goals:

- It provides intuitive quantitative measures for the modularity of individuals. These measures are largely independent of special types of EAs (as GA, ES, GP, LCS etc). Note that the approach introduced differs from formalisms that characterize the modularity of the search problem which are used, e.g., in the Estimation of Distribution Algorithms (EDA) or related work (cf. Mühlenbein & Mahnig, 2003; Mühlenbein & Höns, 2005).

- Using these measures, it investigates (using a simplistic experimental model) how modularity can influence evolution dynamics. In particular, it answers the question whether modularity can be a positive selection factor and thus a cause of Self-Organized Modularization.

In section 2, a general concept concerning the modularity of individuals is developed and some discrete and continuous measures (partially published by Dauscher & Uthmann, 2002) are introduced.

The close relationship between evolutionary forces and the concept of modularity can be derived theoretically for a simple special case, which is shown in section 3.

Further computer simulations are based on a toy model, described in section 4. The results presented in section 5 confirm the theoretical investigations and show some additional effects which can be of interest for future research, and which may also concern the existing approaches sketched in section 6.

2 A Formal Definition of Modularity

2.1 Modularity in Evolutionary Algorithms

Considering existing evolutionary scenarios, which take on the problem of modularization, two different extreme concepts of modularity can be distinguished:

In some approaches, several individuals constitute a composite problem solution (e.g., Darwen & Yao, 1997); individuals act as modules rather than being modular themselves. The other extreme considers models with individuals, each of which has or is developing a modular structure. These individuals do not interact, except for the competitive effects in the evolutionary scenario (e.g., Wagner, 1995; Altenberg, 1995a; Harik & Goldberg, 1996; Calabretta, Nolfi, Parisi, & Wagner, 1998). Intermediate models can be found as well (e.g., Watson & Pollack, 1999; Polani & Miikkulainen, 1999).

This paper concentrates on the dynamics of the formation of modular individuals, not on the formation of modular structures consisting of several individuals.

We use an intuitive technical example as a starting point: Most people would characterize a computer graphics adapter as a module. If we consider two out of order PCs being “recycled” in order to build one new PC, the new one is most likely to have a complete graphic adapter from one of the PCs, not part of one graphics adapter and
part of the other. Aside from the aggregation of similar functions, this recombination property can be considered as the main characteristic of a module.

We generalize this intuition within the context of Evolutionary Algorithms by considering individuals to be modular w.r.t. a certain part (or property) if this part (or property) tends to be transmitted to the offspring either as a whole or not at all.

2.2 The Radcliffe Meta-Model and Radcliffe Modularity

The problem in creating a general concept of modularity is that it has to cover a number of rather different model approaches, such as Simple Genetic Algorithms, genetically encoded Neural Networks, Genetic Programming, Learning Classifier Systems etc.

Radcliffe provides one way to tackle this difficulty with his algebra for Genetic Algorithms (Radcliffe, 1994), forthwith called the Radcliffe Meta-Model\(^1\). This approach is based on a search space \( S \) consisting of all thinkable genotypes (denoted by indexes \( i, j, k \) in the following).

This search space is partitioned by equivalence relations\(^2\) into corresponding sets of (mutually disjoint) equivalence classes \( \xi_1 \ldots \xi_K \) as shown in Fig. 1. An equivalence relation corresponds to a given kind of module (as the property “graphics adapter” in the example), the respective equivalence classes correspond to concrete modules (as, e.g. “ATI Rage Pro 128”, “NVIDIA Quadro4 980” etc.). If two elements of the search space belong to the same equivalence class, they have the respective module in common (e.g., they have the same type of graphics adapter). The search space can, of course, be partitioned by a multitude of equivalence relations (in the above mentioned example, this might be keyboards, hard disks, etc). If an equivalence relation has only one equivalence class, all elements of the search space are equivalent w.r.t. this relation. This (trivial) case will not be considered in the following discussion.

\[ \Xi_{\psi} = \{ \xi_1 \ldots \xi_4 \} \]

Figure 1: An equivalence relation \( \psi \) divides the search space \( S \) into a set \( \Xi_{\psi} \) of equivalence classes \( \xi \).

In (Radcliffe, 1994), a function \( X(j, k, \kappa) \) formalizes the crossover operator. It describes the possible offspring genotypes resulting from a mating of two individuals having genotypes \( j \) and \( k \).

\(^1\)By Meta-Model we understand a model comprising a class of several model approaches on a more abstract level (cf. Perl, Lames, & Glitsch, 2002).

\(^2\)Radcliffe (as well as this paper) uses the standard mathematics notion of an equivalence relation as a relation which is reflexive (\( a \sim a \)), symmetric (\( a \sim b \iff b \sim a \)) and transitive (\( a \sim b \land b \sim c \Rightarrow a \sim c \)), where “\( \sim \)” denotes “equivalent”.

Evolutionary Computation Volume 13, Number 3
The control parameter $\kappa \in \mathcal{K}$ is introduced to model the fact that the mating of the same two individuals can yield different offspring individuals (by random influence). The set $\mathcal{K}$ of possible values depends on the particular operator. E.g., for a one-point crossover operator in an 8 bit GA, $\mathcal{K} = \{1 \ldots 7\}$ could be chosen according to the 7 non-trivial cross-points.

Using this Meta-Model, modularity can now be formalized in a highly general way: An equivalence relation $\psi$ is said to be Radcliffe modular if and only if each offspring individual resulting of a mating is equivalent to at least one of its parents. Using Radcliffe’s formalism, this can be expressed by:

$$\forall j \in \xi, k \in \xi' \forall \kappa \in \mathcal{K} : X(j, k, \kappa) \in (\xi \cup \xi')$$

where $\xi, \xi' \in \Xi_{\psi}$ denote equivalent classes induced by the equivalence relation $\psi$.

One should note that this notion of modularity can be seen as a special case of what Radcliffe calls the Gene Transmission property (Radcliffe, 1994).

A high degree of generality for this measure is provided by the fact that the concepts of search space and of equivalence relations and classes, as well as the formalization of crossover, are largely independent of the special types of EAs (as GA, ES, GP, LCS).

Figure 2: If an equivalence relation is Radcliffe modular, the crossover of two genotypes $j, k$ of classes $\xi$ and $\xi'$, respectively, always yields a genotype of one of these classes. This can be elegantly expressed using the abbreviation $r_{i,j,k}$ introduced in section 2.3, Eq. (9).

2.3 Synthesis with the Altenberg Meta-Model

Using Radcliffe Modularity, we now have a criterion for equivalence relations which describes whether a relation can be considered to be modular or not. Unfortunately, this measure is rather rigorous and coarse. Furthermore, it does not take into account the single genotypes. A quantitative measure of modularity is needed to describe processes of Self-Organized Modularization. This measure must also be assignable to each single genotype.

Since the Radcliffe Meta-Model framework does not model the quantitative composition of the offspring of a mating, it has to be extended. This is done here using a formalism used by Altenberg as a Meta-Model for Evolutionary Algorithms in (Altenberg, 1994).

---

\(^3\)It is not necessary that all offspring individuals are equivalent to the same parent.
Here, populations are modeled as vectors (denoted here by \( x, y \) or \( z \)). Each component (e.g. \( x_i \)) of such a vector corresponds to a given genotype \( i \). The value of the component represents the relative frequency of the respective genotype in the modeled population. Since relative frequencies are used, the vectors are normalized: \( \sum_i x_i = 1 \).

The genetic operators can now be formalized as follows:

**Selection** One can understand selection to be a process generating a virtual intermediate population \( y \) from the population \( x \) according to the values of the objective function \( w_i \) indicating the “fitness” of genotype \( i \). Assuming selection proportional to the objective function, the (expected) relative frequency of genotype \( j \) in the selected population \( y \) is

\[
y_j = x_j \frac{w_j}{w}
\]  

One should note that other selection schemes such as rank selection or tournament selection, as well as mechanisms like elitism, would be much more difficult to model within this framework.

**Crossover** It is this intermediate population \( y \) on which the crossover operator acts. A further virtual intermediate population is generated by combining pairs of genotypes in \( y \) to form one or more new individuals, forming a new intermediate population \( z \).

\[
z_l = \sum_{j,k} T^{(X)}(l \leftarrow j,k) y_j y_k
\]  

\( T^{(X)}(l \leftarrow j,k) \) corresponds to the (expected) probability that a crossover among two individuals of types \( j \) and \( k \) form an individual of type \( l \). I.e., \( T^{(X)}(l \leftarrow j,k) \) is the probability distribution of the offspring genotypes given the parents of type \( j \) and \( k \). It will be called crossover transmission function.

**Mutation** After crossover process, the individuals in \( z \) may be slightly changed by mutation. This leads to the next population \( x' \).

\[
x'_i = \sum_l T^{(M)}(i \leftarrow l) z_l
\]  

where \( T^{(M)}(i \leftarrow l) \) will be called mutation transmission function.

All three operations can easily be subsumed under one term directly expressing the (expected) relative frequencies in the next generation \( x' \) in terms of the frequencies of the existing \( x \):

\[
x'_i = \sum_{j,k} T(i \leftarrow j,k) \frac{w_j w_k}{w^2} x_j x_k
\]  

with

\[
T(i \leftarrow j,k) \equiv \sum_l T^{(M)}(i \leftarrow l) T^{(X)}(l \leftarrow j,k)
\]  

which is the transmission function introduced directly in (Altenberg, 1994) (without introducing \( T^{(X)} \) and \( T^{(M)} \) separately). The formalization of variation by a transmission function is again largely independent of special types of EA. In this context, we emphasize that other theoretical papers and books also make use of similar formal descriptions. E.g., in (Vose, 1999) the term mixing is used instead of transmission function; \( T(i \leftarrow j,k) \) is denoted there as \( m_{jk}(i) \).
It is to be noted that $x'$ does not actually describe the relative frequencies of an actual next generation but only an expectation value, which differs especially if one considers small populations. When modeling a sequence of iterations, this can be problematic. However, in the further investigations just one step is considered.

Since both Meta-Models are based on the concept of a search space $S$ consisting of genotypes, a synthesis can easily be obtained by introducing some functions. The characteristic function

$$c_i(\xi) \equiv \begin{cases} 1, & \text{if } i \in \xi \\ 0, & \text{otherwise} \end{cases}$$

(7)

describes whether a genotype $i$ belongs to an equivalence class $\xi$ or not. For the sake of formal similarity, we further introduce

$$r_{ij}(\psi) \equiv \begin{cases} 1, & \text{if } i \sim \psi j \\ 0, & \text{otherwise} \end{cases}$$

(8)

indicating whether two genotypes are related by the equivalence relation $\psi$ or not.

The abbreviation

$$r_{i,jk}(\psi) \equiv \begin{cases} 1, & \text{if } (r_{ij}(\psi) = 1) \lor (r_{ik}(\psi) = 1) \\ 0, & \text{otherwise} \end{cases}$$

(9)

indicates whether a genotype $i$ is equivalent at least to one of $j$ and $k$. The usefulness of this abbreviation can be seen in rewriting the condition (1) for Radcliffe Modularity as

$$\forall j, k \in S : \quad r_{i,jk}(\psi) = 0 \quad \Rightarrow \quad T^{(X)}(i \leftarrow j, k) = 0$$

(10)

which is also illustrated in Fig. 2. Because of normalization, the condition for Radcliffe Modularity in Eq. (10) can be rewritten as

$$\forall j, k \in S : \quad \sum_i T^{(X)}(i \leftarrow j, k)r_{i,jk}(\psi) = 1$$

(11)

2.4 Quantitative Modularity Measures

Since the sum in (11) will be important for the further considerations, we will use an abbreviation

$$m_{jk}^{(X)}(\psi) \equiv \sum_i T^{(X)}(i \leftarrow j, k)r_{i,jk}(\psi)$$

(12)

we will refer to as the modularity matrix element of pure crossover. This quantitative value describes the expected proportion of offspring which share an equivalence class with at least one of the given parents (of genotypes $j$ and $k$). Radcliffe Modularity is now simply defined by rewriting Eq. (11):

$$\forall j, k \in S : \quad m_{jk}^{(X)}(\psi) = 1$$

(13)

For the further investigations of evolutionary forces driving dynamics toward more modular individuals, it is necessary to obtain a measure not only for the modularity of a combination of genotypes, but also for that of a single genotype. We will introduce two different measures in this context:
**Built-in Modularity** A rather simple measure for the modularity of a genotype $j$ is given by the average of the matrix elements $m_{jk}(\psi)$ over all $k \in S$.

$$M_j^{(X)}(\psi) = \frac{\sum_k m_{jk}^{(X)}(\psi)}{|S|}$$  \hspace{1cm} (14)

where $|S|$ denotes the size of the search space. We will call this the *Built-in modularity of pure crossover*, as it depends only on the matrix elements, all of which are fixed during the evolution process.

This measure might be interesting for theoretical investigations or “Gedankenexperimente”. However, it is hard to compute in typical practical Evolutionary Algorithms, since for each element of the (typically huge) search space a computation has to be performed. Furthermore, this measure is not likely to influence the dynamics of an Evolutionary Algorithm directly, as only a few genotypes of the search space are represented as individuals in the population.

**Effective Modularity** Because of these two arguments, it seems to be useful to introduce another modularity measure for a genotype $j$ based on the current (intermediate) population $y$ produced by the selection step: It is obtained by computing the average of all corresponding matrix elements weighted by the relative frequency $y_k$ of the crossover partner $k$ in the population $y$:

$$M_j^{(X)}(\psi, y) \equiv \sum_k m_{jk}^{(X)}(\psi) y_j$$  \hspace{1cm} (15)

We will call this measure the *Effective Modularity of pure crossover*. In contrast to the *Built-in Modularity*, this measure is quite easy to compute, since only the genotypes which are present in the population have to be considered. One should note that there is an explicit dependence on the whole population $y$. This will prove crucial for some dynamical effects.

Because of (13) and the normalization of $y$ the following equations hold if $\psi$ is *Radcliffe modular*:

$$\forall j : M_j^{(X)}(\psi) = 1 \quad \text{and} \quad \forall j : M_j^{(X)}(\psi, y) = 1$$  \hspace{1cm} (16)

**2.5 Generalization**

So far, the effects of mutation have not been considered. However, these effects can play an important role, as they can influence the transmission of parts (or properties) as a whole in a destructive way. Especially in models where mutation rates may vary (cf. Bäck & Schütz, 1996), these effects could be of interest. These effects of mutation can easily be integrated by simply replacing the function $T^{(X)}(i \leftarrow j, k)$ (modeling crossover only) by $T(i \leftarrow j, k)$ (additionally modeling the mutation effects), yielding

$$m_{jk}(\psi) \equiv \sum_i T(i \leftarrow j, k) r_{i,jk}(\psi)$$  \hspace{1cm} (17)

$$M_j^{(X)}(\psi) \equiv \frac{\sum_k m_{jk}(\psi)}{|S|}$$  \hspace{1cm} (18)

$$M_j^{(X)}(\psi, y) \equiv \sum_k m_{jk}(\psi) y_j$$  \hspace{1cm} (19)
which we call the Modularity matrix element, Built-in Modularity, and Effective Modularity omitting “... of pure crossover”.

Thus, we have developed a modularity measure dependent on each single genotype, the population and an equivalence relation which takes into account both crossover and mutation effects.

2.6 Synopsis of the Measures Introduced

In the following, we summarize the modularity measures introduced:

<table>
<thead>
<tr>
<th>Measure</th>
<th>... of pure crossover</th>
<th>including mutation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Modularity Matrix Element</td>
<td>( m_{jk}^{(X)}(\psi) )</td>
<td>( m_{jk}(\psi) )</td>
</tr>
<tr>
<td>of two genotypes ( j ) and ( k ) w.r.t. an equivalence relation ( \psi )</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Built-in Modularity</td>
<td>( \overline{M}_j^{(X)}(\psi) )</td>
<td>( M_j(\psi) )</td>
</tr>
<tr>
<td>of a genotype ( j ) w.r.t. an equivalence relation ( \psi )</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Effective Modularity</td>
<td>( M_{j,y}^{(X)}(\psi) )</td>
<td>( M_{j,y}(\psi) )</td>
</tr>
<tr>
<td>of a genotype ( j ) w.r.t. an equivalence relation ( \psi ) and a selected population ( y )</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

2.7 The Importance of Respected Equivalence Relations

An important property of these measures is revealed when the case of equivalence relations respected by the crossover operator is considered. An equivalence relation \( \psi \) is said to be respected by a crossover operator if the crossover of any two genotypes of the same equivalence class \( \xi \in \Xi_\psi \) yields a genotype of the same class \( \xi \) (as shown in Fig. 3). In bit-string oriented Simple Genetic Algorithms, the equivalence classes induced by respected equivalence relations correspond to Holland-style schemata (Holland, 1975). An equivalence relation \( \psi \) having schemata as classes is a relation defining two genotypes to be equivalent if and only if the fixed (non-joker) bits are identical. Such a relation yields a number of schemata having all the same order \( o(\psi) \), namely \( 2^{o(\psi)} \) different schemata. A relation can be expressed by a string, in which the fixed bits are represented by an “f” (e.g., a relation \( \text{ff} \) induces schemata \( \{(*00*), (*01*), (*10*), (*11*)\} \).

One can easily see that \( k \)-point crossover of two genotypes sharing a common schema always yields individuals sharing this schema. E.g., \((0100)\) and \((1101)\) share the schema \((*10*)\) and all offspring will share this schema, too (regardless of the crosspoints).

It becomes obvious that if two genotypes \( j \) and \( k \) are members of the same equivalence class (induced by a respected equivalence relation \( \psi \)), this will lead to \( m_{jk}^{(X)}(\psi) = 1 \), even if \( \psi \) is not Radcliffe modular. This property might have a strong impact on the dependence of \( M_{j,y}^{(X)}(\psi) \) on the population \( y \), which can easily be shown in an extreme example: If – for some reason – the system has converged to a population \( \tilde{y} \) only consisting of members of one equivalence class \( \xi \in \Xi_\psi \) where \( \psi \) is respected, this automatically yields \( M_{j,\tilde{y}}(\psi) = 1 \) for all \( j \) with non-zero \( y_j \).

2.8 Applicability

Because of the generality of the formalisms used, the measures introduced above can be applied to practically all kinds of Evolutionary Algorithms. The only thing one has to
Figure 3: If an equivalence relation $\psi$ is respected by a crossover operator, the crossover of two genotypes $j$ and $k$ which are members of the same equivalence class $\xi \in \Xi_\psi$ always yields again a class member.

Do is to choose an equivalence class $\psi$ subdividing the search space $S$ into a number of equivalence classes $\xi$. Basically, the relations can be chosen arbitrarily; however, depending on the approach investigated, some types of equivalence relations appear to be natural: In Simple Genetic Algorithms, for example, equivalence relations having schemata as classes are particularly interesting. In this context, it is easy to see that in the case of one-point crossover the inequality

$$m_{jk}(\psi) \geq 1 - p_c \frac{o(\psi)}{\ell} - p_m \cdot o(\psi)$$

holds, where $\psi$ is a schemata-inducing equivalence relation, $\ell$ the genome length, $p_c$ the crossover probability, $p_m$ the mutation probability of a bit, while $o(\psi)$ and $\delta(\psi)$ denote order and defining length of the induced schemata. This shows a considerable similarity to the terms used in the well-known Schema Theorem (Holland, 1975).

Furthermore, it should be noted that, even in complicated EA approaches, the Modularity matrix elements or the Effective Modularity of genotypes (w.r.t. a given population) for a given equivalence relation can be simply computed numerically. This can be accomplished by doing Monte Carlo simulations of a variation step and counting the offspring individuals in the respective equivalence classes.

3 Modularity and Evolutionary Forces

One of the most interesting questions in the context of modularity is, whether a higher modularity of a genotype $j$ can cause a positive selective pressure (compared to other genotypes of equal “fitness” but a lesser modularity).

Considering (2), a direct connection between the modularity of a genotype $j$ and the number of its offspring is not obvious; this equation reveals nothing about the fitness of the offspring and therefore about its success in further generations. In order to take such effects into account, we introduce the Next-Generation Fitness Matrix Element

$$n_{jk} = \sum_i w_i T(i \rightarrow j, k)$$

which describes the expectation value of the offspring fitness resulting from a mating of two genotypes $j$ and $k$. Similar to the Modularity Measure, these matrix elements...
can be used to construct a measure for a genotype \( j \) w.r.t. the selected population \( y \)

\[
N_{j,y} = \sum_k n_{jk} y_k
\]

(22)

which we will call the Next-Generation Fitness.

We will now construct a rather simple case, for which a simple connection between the matrix elements \( m_{jk}(\psi) \) and \( n_{jk} \) as well as the values for \( M_{j,y}(\psi) \) and \( N_{j,y} \) can be found theoretically\(^4\). We consider a simple fitness landscape having only two levels \( \beta > 0 \) and \( \alpha + \beta, \alpha > 0 \). We further consider an equivalence relation \( \psi \) subdividing the search space into a multitude of classes \( \xi \).

One of these classes will be called \( \xi^\uparrow \). The genotypes in this class will be the only ones having the higher fitness value \( \alpha + \beta \). Therefore, the fitness of a genotype \( j \) can be defined by

\[
w_j = \alpha c_j(\xi^\uparrow) + \beta
\]

(23)

using the definition in Eq. (7). For the sake of simplicity, we abbreviate \( c_i^\uparrow \equiv c_i(\xi^\uparrow) \), yielding

\[
w_j = \alpha c_j^\uparrow + \beta
\]

(24)

Assuming it to be relatively unlikely (because of the multitude of classes) that two genotypes \( j, k \not\in \xi^\uparrow \) have offspring in \( \xi^\uparrow \), and adding some basic symmetry assumptions, one can derive the following connection of the Modularity Matrix Elements and the respective Next-Generation Fitness Matrix Elements:

\[
n_{jk} = \alpha \left( \frac{c_j^\uparrow + c_k^\uparrow}{2} \right) m_{jk}(\psi) + \beta
\]

(25)

The complete derivation can be found in the Appendix.

For some special but important cases of population composition, \( \overline{y}_i \) and \( \overline{y}_j \) with

\[
\overline{y}_i \simeq 0 \quad \forall j \in \xi^\uparrow \quad \text{(almost no individual in} \quad \xi^\uparrow \text{is present)}
\]

(26)

\[
\overline{y}_j \simeq 0 \quad \forall j \not\in \xi^\uparrow \quad \text{(almost no individual outside} \quad \xi^\uparrow \text{is present)},
\]

(27)

the values \( M_{j,y}(\psi) \) and \( N_{j,y} \) are also closely interconnected: For genotypes \( j^\uparrow \in \xi^\uparrow \) and \( j^\downarrow \not\in \xi^\uparrow \) the following equations (also shown in Fig. 4) hold:

\[
N_{j^\uparrow,y} = \frac{\alpha}{2} \sum_k \left[ (c_j^\uparrow + c_k^\uparrow) m_{jk}(\psi) \right] \overline{y}_k + \beta = \approx \frac{\alpha}{2} \sum_k \left[ (1 + 1) m_{jk}(\psi) \right] \overline{y}_k + \beta = \approx \sum_k m_{jk}(\psi) \overline{y}_k + \beta = \alpha M_{j,y}(\psi) + \beta
\]

(28)

\(^4\)Similar connections are also likely to exist for more complex scenarios. However, it would be much more difficult or even impossible to derive them analytically and record them in complete form. In most real-world applications, a potential connection must be found by direct measurements in the computer simulation.
The “≈” refers to the fact that the population is only found “almost” completely inside $\xi^\uparrow$ or outside $\xi^\downarrow$, respectively. Similarly, the following equations can be derived:

$$N_{j;i,\downarrow} \approx N_{j;i,\uparrow} \approx \frac{\alpha}{2} M_{j;\psi} + \beta$$

(29)

$$N_{j;i,\downarrow} \approx \beta$$

(30)

Figure 4: $M-N$-plot as graphical representation of the equations (28)-(30).

The most striking feature of Eqs. (28) and (29) is that two genotypes having the same fitness value $w_j$ can largely differ in Next Generation Fitness dependent of their value of Effective Modularity. This leads to the hypothesis that Effective Modularity could indeed be a selection factor. This hypothesis is underpinned by experimental results described in section 5.

In these investigations, $M-N$-plots similar to the one shown in Fig. 4 will be important. Each individual of a population can be displayed as a data point in the diagram showing its Effective Modularity and its Next Generation Fitness, both of which can be obtained by means of Monte-Carlo simulations. The position of these data points can easily be compared with the theoretical results. Thus, the change of the joint distributions of Effective Modularity and Next Generation Fitness in the population (possibly indicating processes of Self-Organized Modularization) can be observed directly. This can be useful, e.g., to detect that two genotypes initially having different Built-in Modularity and Effective Modularity values can converge in their Effective Modularity in the course of the dynamics due to homogeneity effects.

One should note that $M-N$-plots can provide insight into the evolutionary dynamics in complex scenarios even if there is no theoretical result the data-points can be compared to.

4 A Simple Simulation Model

As mentioned in section 2.8, the modularity measures introduced can be applied to a variety of typical model approaches, where modularity is modified implicitly by different mechanisms (crossover, mutation or both) which have a number of complex
side-effects. In order to do fundamental research, a simplistic model was developed where Built-in Modularity can be controlled separately by influencing the mutation and crossover mechanisms in a much more explicit way, avoiding model-specific complexity. Since this model is in turn an abstract model of other evolutionary scenarios, it can be conceived as a meta-model and is therefore called MetaEA.

A model for the purpose of systematically investigating processes of Self-Organized Modularization should have the following properties:

1. Modularity-induced selective pressure is observable without any bias induced by fitness. Therefore, fitness and Built-in Modularity should be modelled independently.

2. Differences in Built-in Modularity are alterable in the simulations.

3. Differences in Built-in Modularity can be produced via crossover and/or mutation.

4. Respected equivalence relations, as well as relations that are not respected by the crossover operator, can be modelled.

4.1 The MetaEA Genotype

In order to achieve these properties, a MetaEA genotype consists simply of three integer numbers $P$, $R$, and $M$ which can take on $N^{[P]}$, $N^{[R]}$ and $N^{[M]}$ independent values, respectively. If we consider a number of different genotypes $j$, $k$, etc., we will refer to the respective numbers as, e.g., $P_j$ or $M_k$.

The value of $P$ corresponds to a given phenotype. $P_j = P_k$ models two genotypes $j$ and $k$, that are phenotypically equivalent and thus have the same fitness. In other words: the fitness of a genotype $j$ depends only on its value $P_j$:

$$w_j = w(P_j)$$  \(31\)

As already mentioned, the values of $R$ and $M$ are independent of the values of $P$. They correspond to properties concerning the representation in the genotype: If $P_j = P_k$, but $R_j \neq R_k$ or $M_j \neq M_k$, this models two phenotypically equivalent genotypes $j$ and $k$ that differ in their representation (and therefore possibly in their Built-in or Effective Modularity). If $P_j \neq P_k$ holds, equality or inequality of the $R$ and $M$ values has no further interpretation.

It may be unclear, why two other numbers besides $P$ are used to characterize a genotype. This is done in order to model the respect property: the crossover operator is constructed in such a way that the combination $(P, R)$ is always respected. In other words: If the crossover operator is applied to genotypes $j$ and $k$, and

$$P_j = P_k$$  \(32\)

$$R_j = R_k$$  \(33\)

holds, then we receive the following for type $i$ of an offspring individual:

$$P_i = P_j (= P_k)$$  \(34\)

$$R_i = R_j (= R_k).$$  \(35\)

(The function of the crossover operator in cases of unequal $P$ or $R$ parent values will be discussed later).
By setting $N^{[R]} = 1$, we can model the phenotype to be respected by the crossover operator.

In order to model a multiplicity of genotypes leading to the same phenotype (especially in this case of $N^{[R]} = 1$), the additional variable $M$ becomes necessary. If the phenotype modelled is not to be respected by the crossover operator, we use $N^{[R]} > 1$.

### 4.2 The Evolutionary Operators in MetaEA

Modeling different genotypes having the same phenotype is nearly futile, unless differences in the genotypic representation have no effect on the function of the evolutionary operators.

In MetaEA, the influence of the genotypes on the crossing over operator is modelled by introducing two simple functions: the mixing probability $p_{\text{mix}}(j)$ and the mutation probability $p_{\text{mut}}(j)$. In order to keep the scenario simple, each of these functions (depending on three parameters $P_j$, $R_j$, and $M_j$) are, in turn, composed of three simple functions, each depending on one parameter only and which are fixed for each simulation run:

\begin{align}
  p_{\text{mix}}(j) &\equiv p_{\text{mix}}(P_j, R_j, M_j) = \min \left( p_{\text{mix}}^{[P]}(P_j), p_{\text{mix}}^{[R]}(R_j), p_{\text{mix}}^{[M]}(M_j) \right) \\
  p_{\text{mut}}(j) &\equiv p_{\text{mut}}(P_j, R_j, M_j) = \min \left( p_{\text{mut}}^{[P]}(P_j), p_{\text{mut}}^{[R]}(R_j), p_{\text{mut}}^{[M]}(M_j) \right)
\end{align}

In order to make things simpler, each mating results in a pair of offspring individuals of types $i$ and $i'$, where $i$ is only influenced by $p_{\text{mix}}(j)$ (crossover) and $p_{\text{mut}}(j)$ (mutation). Similarly, $i'$ is only influenced by $p_{\text{mix}}(k)$ and $p_{\text{mut}}(k)$. The $M$ property of individuals is trivially transmitted from parents to the offspring and slightly mutated by some fixed mutation rate $p_m$. This is shown graphically in Fig. 5; the exact algorithm is given in Fig. 6 as pseudo code.

![Figure 5: Influence of the parameters $p_{\text{mix}}(\cdot)$ and $p_{\text{mut}}(\cdot)$ in the MetaEA-Algorithmus.](image_url)

Note that the combination $p_{\text{mut}}(j) = 0$, $p_{\text{mut}}(k) = 0$ and $p_m = 0$ corresponds to a pure crossover. A combination $p_{\text{mix}}(j) = 0$, $p_{\text{mix}}(k) = 0$ statistically corresponds to the “mutation only” case.
// Crossover of the \((P, R)\) properties of the mating individuals

\[
\text{IF } (P_j, R_j) = (P_k, R_k) \text{ THEN }
\]

// Respect property of the crossover operator

\[
(P_l, R_l) ← (P_j, R_j)
\]

\[
(P_l′, R_l′) ← (P_j, R_j)
\]

ELSE

// Generate virtual offspring individual of type \(l\)

WITH \text{PROBABILITY } p_{\text{mix}}(j)

DO

// Completely randomly generated offspring individual

\[
P_l ← \text{RANDOM}(0 \ldots N^P - 1)
\]

\[
R_l ← \text{RANDOM}(0 \ldots N^R - 1)
\]

ELSE

WITH \text{PROBABILITY } 0.5

DO

\[
(P_l, R_l) ← (P_j, R_j)
\]

ELSE

\[
(P_l, R_l) ← (P_k, R_k)
\]

// Generate virtual offspring individual of type \(l′\)

WITH \text{PROBABILITY } p_{\text{mix}}(k)

DO

// Completely randomly generated offspring individual

\[
P_{l′} ← \text{RANDOM}(0 \ldots N^P - 1)
\]

\[
R_{l′} ← \text{RANDOM}(0 \ldots N^R - 1)
\]

ELSE

WITH \text{PROBABILITY } 0.5

DO

\[
(P_{l′}, R_{l′}) ← (P_j, R_j)
\]

ELSE

\[
(P_{l′}, R_{l′}) ← (P_k, R_k)
\]

// Generate preliminary copy of the virtual offspring individual to the real offspring individual

\[
(P_i, R_i) ← (P_l, R_l)
\]

\[
(P_i′, R_i′) ← (P_{l′}, R_{l′})
\]

// Mutation of \((P_i, R_i)\)

WITH \text{PROBABILITY } p_{\text{mut}}(j)

DO

\[
P_i ← \text{RANDOM}(0 \ldots N^P - 1)
\]

\[
R_i ← \text{RANDOM}(0 \ldots N^R - 1)
\]

// Mutation of \((P_{i′}, R_{i′})\)

WITH \text{PROBABILITY } p_{\text{mut}}(k)

DO

\[
P_{i′} ← \text{RANDOM}(0 \ldots N^P - 1)
\]

\[
R_{i′} ← \text{RANDOM}(0 \ldots N^R - 1)
\]

// Trivial crossover for \(M\)

\[
M_i ← M_j
\]

\[
M_i′ ← M_k
\]

// Mutation of \(M_i\)

WITH \text{PROBABILITY } p_m

DO

\[
M_i ← \text{RANDOM}(0 \ldots N^M - 1)
\]

// Mutation of \(M_{i′}\)

WITH \text{PROBABILITY } p_m

DO

\[
M_{i′} ← \text{RANDOM}(0 \ldots N^M - 1)
\]

Figure 6: Algorithm realizing crossing over and mutation in MetaEA
4.3 Matrix Elements in MetaEA

In the following, we restrict ourselves to an equivalence relation \( \psi[\mathcal{P}] \), describing whether two types are phenotypically equal, i.e. whether they have the same value for \( P \). The modularity matrix elements of MetaEA w.r.t. this relation depend on

a) the equivalence of \( j \) and \( k \) w.r.t. \( P \) and \( R \) due to the modeling of the respect mechanism in the crossover operator.

b) the parameters \( p_{\text{mix}}(j), p_{\text{mix}}(k), p_{\text{mut}}(j) \) and \( p_{\text{mut}}(k) \), for which we will define two convenient abbreviations:

\[
\begin{align*}
\overline{p_{\text{mix}}} &= \frac{p_{\text{mix}}(j) + p_{\text{mix}}(k)}{2} \\
\overline{p_{\text{mut}}} &= \frac{p_{\text{mut}}(j) + p_{\text{mut}}(k)}{2}
\end{align*}
\]  

The modularity matrix elements of pure crossover are

\[
m^{(X)}_{jk}(\psi[\mathcal{P}]) = \begin{cases} 
1 & \text{if } (P_j, R_j) = (P_k, R_k) \\
\frac{1}{2} \overline{p_{\text{mix}}} + \frac{1}{N[\mathcal{P}]} & \text{if } P_j = P_k, R_j \neq R_k \\
\frac{1}{2} \overline{p_{\text{mix}}} + \frac{1}{N[\mathcal{P}]} & \text{if } P_j \neq P_k,
\end{cases}
\]

where the first case models the respect property independently of the \( p_{\text{mix}} \) values. In the second case, with probability \( \overline{p_{\text{mix}}} \) an offspring individual is chosen randomly. Since both parents have the same equivalence class, the offspring has probability \( \frac{1}{N[\mathcal{P}]} \) to be in exactly this equivalence class. The third case is similar, except for the fact that both parents have different equivalent classes and the probability to hit one of them is \( \frac{2}{N[\mathcal{P}]} \).

Note that for the special case \( \overline{p_{\text{mix}}} = 0 \) the equation

\[
m^{(X)}_{jk}(\psi[\mathcal{P}]) = 1 \quad \forall j, k
\]

holds.

Using similar arguments as above, for the general modularity matrix element (including mutation effects) one finds

\[
m_{jk}(\psi[\mathcal{P}]) = \begin{cases} 
(1 - \overline{p_{\text{mut}}}) m^{(X)}_{jk}(\psi[\mathcal{P}]) + \overline{p_{\text{mut}}} \frac{1}{N[\mathcal{P}]} & \text{if } P_j = P_k \\
(1 - \overline{p_{\text{mut}}}) m^{(X)}_{jk}(\psi[\mathcal{P}]) + \overline{p_{\text{mut}}} \frac{1}{N[\mathcal{P}]} & \text{if } P_j \neq P_k
\end{cases}
\]

5 Simulation Results

In the context of (Dauscher, 2003), a series of simulation experiments were performed. The most interesting results will be presented in the next three subsections.

We exclusively consider the case where one phenotype \( P^1 \) has fitness \( \alpha + \beta \), while all others have fitness \( \beta \) (similarly to the theoretical investigations in section 3).

5.1 Observations in a Simple Test Scenario

At first, we will show the results of a simple test scenario in order to explain some general effects which are important for further investigations. We only consider different phenotypes, i.e. \( N[\mathcal{P}] = N[\mathcal{M}] = 1 \). The parameters \( p_{\text{mix}} \) and \( p_{\text{mut}} \) are the same for all types of search space.
The $\mathcal{M}-\mathcal{N}$-plots for the initial population (uniformly distributed over the search space, corresponding to $\overline{y}$ because of the large number of phenotypes) and a later population (almost converged to the fittest phenotype, corresponding to $\overline{\overline{y}}$) are shown in Fig. 7. Some observations can be made:

- For both cases $\overline{y}$ and $\overline{\overline{y}}$, theoretical and simulation results are in good agreement (the clusters are situated on the relations obtained theoretically in section 3). The upper cluster corresponds to the $P^\uparrow$-individuals having a fitness $\alpha + \beta$, the lower cluster to individuals with fitness $\beta$ only.

- As expected, larger values of $p_{\text{mix}}$ result in lower values for $\mathcal{M}_{j,y}(\psi^{[P^\uparrow]})$ in the initial population, as the probability of obtaining individuals not equivalent to their parents in the crossover process grows with $p_{\text{mix}}$.

- Although the Built-in Modularity can be considered to be constant, the $\mathcal{M}_{j,y}(\psi^{[P^\uparrow]})$ and $\mathcal{N}_{j,y}$ values of the $P^\uparrow$-cluster grow in the course of the evolution. They reach values comparable to the $p_{\text{mix}} = 0$ case. This is due to homogeneity effects: If the population has almost converged to $P^\uparrow$-individuals, the probability for an $P^\uparrow$-individual to mate with another such individual is rather high. Since the phenotype property $P$ is respected, the value of $p_{\text{mix}}$ is irrelevant in this case (cf. Eq. (40)).

At first sight, the change of Effective Modularity due to homogeneity effects might seem to be an undesirable artifact, since the construction of the genotypes does not change in time. In subsection 5.3, however, it will turn out that this effect of growing Effective Modularity due to homogeneity is indeed highly relevant for evolution dynamics.

5.2 A Scenario with a Respected Phenotype

In this scenario, we define a search space, where for each phenotype $P$ a number $N^{[M]} > 1$ of different genotypes are possible. As in the test scenario, $N^{[R]}$ is set to 1,
so that the phenotype is respected: If two genotypes have the same phenotype $P$, the (virtual) crossover offspring always has phenotype $P$, too.

One of the genotypes for a given phenotype has a higher Built-in Modularity by construction (i.e., for some $\hat{M}$, the values $p_{\text{mix}}(P, R, \hat{M})$ or $p_{\text{mut}}(P, R, \hat{M})$ are lower). We consider two cases, separately:

**Case 1:** The $\hat{M}$ genotypes differ in their $p_{\text{mut}}$ value: $p_{\text{mut}}(\hat{M}) < p_{\text{mut}}(M \neq \hat{M})$.

**Case 2:** The $\hat{M}$ genotypes differ in their $p_{\text{mix}}$ value: $p_{\text{mix}}(\hat{M}) < p_{\text{mix}}(M \neq \hat{M})$.

In both scenarios, we can see that there are generally four clusters instead of two. Each of the original clusters are split in two, depending on whether $M$ equals $\hat{M}$ or not. In general, the $\hat{M}$ clusters have a higher Effective Modularity.

**Case 1: Difference in $p_{\text{mut}}$** Some $M$-$N$-plots for this case are shown in Fig. 8. Once more, for the initial and for the converged population, theory and simulation are in good agreement. Since $p_{\text{mix}} > 0$, the clusters move towards higher values for $M_{j,y}(\psi[P])$ and $N_{j,y}$. Their relative distance, however, remains constant.

![Figure 8: Three $M$-$N$-plots at different $t$ in Case 1. Especially the distance between the two $P^\uparrow$-clusters remains constant.](image)

In the context of Selforganizing Modularization, the development of modularity and the proportion of $\hat{M}$ genotypes (denoted by $\hat{z}^{(M)}$) in the course of the evolution process is of special interest (Fig. 9). The values of both observables increase with time and then remain on a nearly constant level.

Genotypes having a higher Built-in Modularity ($\hat{M}$ genotypes) tend to increase their proportion in the population, i.e., they are under positive selective pressure, although their fitness values $w_j$ are in no way different from those with $M \neq \hat{M}$. In other words: Self-Organized Modularization does indeed take place in the scenario of Case 1. One can assume that this positive selective pressure is due to the higher Next-Generation Fitness $N_{j,y}$. This will become even more obvious in the second case.

**Case 2: Difference in $p_{\text{mix}}$** A similar computer experiment has been performed for Case 2; the results plotted in Figures 10 and 11 show a remarkable difference to Case 1.

The distance between the two $P^\uparrow$-clusters in the $M$-$N$-plot diminishes until both clusters finally merge. Similarly to the shift of the clusters for $p_{\text{mix}} > 0$, this is due to homogeneity effects: Because of the respect of the $P$-property, the different values for $p_{\text{mix}}$ become less and less important in a population converging to $P^\uparrow$-individuals.

This has an important effect on the positive selective pressure on the $\hat{M}$ genotypes. In the beginning of the evolution, the $P^\uparrow$-Clusters differ in both Built-in and Effective
Figure 9: Development of the proportion of \( \hat{M} \) genotypes \( \hat{x}_p(M) \) and the average Effective Modularity \( \mathcal{M}_y(\psi[P]) \) in the course of the evolution process. The values of both observables increase with time and then remain on a nearly constant level.

Figure 10: Three \( M-N \)-plots at different \( t \) in Case 1. The distance between the two \( P^\uparrow \)-clusters decreases with time until both clusters finally merge.

**Modularity (and therefore in their Next-Generation Fitness).** In the course of time, however, the difference in \( \mathcal{M}_{j,y}(\psi[P]) \) (and therefore in \( N_{j,y} \)) diminishes. This causes the positive selective pressure to decrease as well, which can be observed in Fig. 11: The proportion \( \hat{x}_p(M) \) of \( \hat{M} \) individuals first increases significantly, but decreases again after the selective pressure has vanished. The average Effective Modularity, however, remains on a high level.

This shows that Built-in Modularity affects evolution dynamics mainly via Effective Modularity: If the difference in Effective Modularity decreases despite a persisting difference in Built-in Modularity, the selective pressure reduces, as well.

The inverse case, i.e. a selective advantage by an emergent higher Effective Modularity despite equal initial Built-in Modularity, is shown in the next section.

### 5.3 A Scenario with a Non-Respected Phenotype

This section shows the importance of respect. We define a scenario, where the phenotype is not respected \( (N^R > 1) \), i.e.: If two genotypes have the same phenotype \( P \) but different values for \( R \), the phenotype of the (virtual) crossover offspring has probability of (almost) \( p_{\text{mix}} \) to have a different phenotype value \( P' \neq P \). For simplicity \( N^{[M]} \) is set to 1. It is to be remembered that the genotype of MetaEA is always respected: If two individuals are identical in \( (P, R) \), the (virtual) crossover offspring individuals will also be of this type. Note that the different genotypes do not differ neither in \( p_{\text{mix}} \) nor
Self-Organized Modularization in EAs

Figure 11: Development of the proportion of $\hat{\mathcal{M}}$ genotypes $\hat{x}[\mathcal{M}]$ and the average Effective Modularity $\mathcal{M}_y(\psi[\mathcal{P}]$) in the course of the evolution process. $\hat{x}[\mathcal{M}]$ first increases because of positive selective pressure, but then decreases again, as the positive selective pressure vanishes because of homogeneity effects.

This leads to a remarkable competition, especially among the genotypes belonging to the phenotype $\mathcal{P}^\uparrow$, which can be observed in the $\mathcal{M}$-$\mathcal{N}$-plots shown in Fig. 12.

Initially, as one might expect, the $\mathcal{P}^\uparrow$-individuals have all the same Effective Modularity and hence form a unique cluster. In the course of the evolution process, however, this cluster splits up into a number of subclusters. At further timesteps, one of these subclusters attains higher and higher values for $\mathcal{M}_{j,y}(\psi[\mathcal{P}])$ and $\mathcal{N}_{j,y}$, while the values of the others decrease again and finally all clusters except one vanish.

5.4 A Hypothetical Mechanism

We propose a (as yet hypothetical) mechanism that probably leads to this kind of behavior\(^5\):

Let us consider a subset of high-fitness genotypes having a given, but arbitrary common value $R = R^*$, i.e. genotypes of the form $(\mathcal{P}^\uparrow, R^*)$. If we assume that, by random fluctuations, the proportion of these genotypes in the population becomes a bit higher than the other high-fitness genotypes $(\mathcal{P}^\uparrow, R \neq R^*)$.

1. Then, similarly to the simple test scenario described in section 5.1, resulting from the respect property, the Effective Modularity of the $(\mathcal{P}^\uparrow, R^*)$ individuals increases. The $(\mathcal{P}^\uparrow, R^*)$ cluster moves to higher values of $\mathcal{M}$ (i.e. to the right).

2. If Effective Modularity increases, Next-Generation Fitness also increases (the $(\mathcal{P}^\uparrow, R^*)$ cluster moves towards the right upper corner along the line derived theoretically as can be observed in the plots).

3. If a higher Next-Generation Fitness of a genotype increases its proportion in the population, it will lead to a further increase of the $(\mathcal{P}^\uparrow, R^*)$.

Now we continue our consideration at (1.) and so on. In other words: this positive feedback-loop would establish a self-amplifying “vicious circle” (as displayed in Fig. 13) having the following effect: although no value for $R$ causes a special property,

\[^5\text{This aspect is to be investigated in more detail in future theoretical work.}\]
some (random) \( R^* \) will obtain a positive and increasing selective pressure over time and therefore soon predominate the entire population. This can be observed in the experiments: in Fig. 12 (g)-(i) one can observe that the remaining cluster corresponding to the \((P^\uparrow, R \neq R^*)\) genotypes finally vanishes\(^6\).

6 Conclusion and Future Work

6.1 Conclusion

This paper has introduced a number of different modularization measures. These are rather general, as they are based on a combination of two formalizations by Radcliffe and by Altenberg, which are not dependent on special types of Evolutionary Algorithms (as GA, GP, LCS, etc).

One main result (obtained by theory as well as by MetaEA simulations) is that under certain conditions, Effective Modularity can definitely be a selection factor. Furthermore, it has been shown in MetaEA simulations that it is Effective Modularity rather than Built-in Modularity that influences the dynamics of the Evolutionary Algorithms.

\(^6\)It should be noted that in the experiments performed, the clusters could be colored interactively. One can observe then, that the remaining cluster corresponds to a given, but arbitrary value \( R = R^* \). The increasing proportion of this cluster could be observed by an interactive histogram.
If future research shows that this is similar in other approaches (which we consider quite likely) this would mean:

1. If individuals in an EA model with the same phenotype (and fitness) can have different values of Built-in Modularity (e.g. by different genotypic representation), this difference may affect the selective pressure (via Next-Generation Fitness).

2. However, one should consider that, in the MetaEA model, this difference in Built-in Modularity exhibits its effects only via the difference in Effective Modularity. If (e.g. due to homogeneity effects) Effective Modularity values are leveled, the positive selective effects might vanish, although the difference in Built-in Modularity remains unchanged.

3. Reciprocally, random fluctuations in an EA may amplify the Effective Modularity (and hence Next-Generation Fitness) of certain genotypes (although their Built-in Modularity and fitness are equal). This can lead to a positive selective pressure on these genotypes. Due to the hypothesized “vicious circle” mechanism, slight effects in the beginning might have remarkable consequences for the overall dynamics.

6.2 Future Work

Of course, the results presented here are only a few basic steps on the way towards a better understanding of the role of modularity in Evolutionary Dynamics. In order to achieve something like a theory of modularity in Evolutionary Algorithms, much more work is to be done. Possible further steps could be to integrate the modularity measures introduced here into other, more elaborate theoretical approaches as described, e.g., in (Vose, 1999). Another highly interesting question is, to what extent the modular structures of the search problem (cf. Mühlenbein & Mahnig, 2003; Mühlenbein & Höns, 2005) are related to the modularity of the individuals evolved within an Evolutionary Algorithm.

Besides further theoretical work, it would be particularly interesting to find out whether similar results as obtained from the MetaEA simulations can be found in the context of existing Evolutionary Algorithm approaches. The results obtained in this paper suggest some interesting starting points:

Premature convergence in models including adaptive mutation rates Typically, in models including adaptive mutation rates, these rates tend to decrease in the course of the evolutionary process. This effect can be very useful, since a period
of coarse adaptation can be followed by “fine tuning”. However, it is frequently observed that mutation rates are decreasing too fast, i.e. they reach almost zero level before an acceptable solution is found (premature convergence). This effect has been described in detail, e.g. by (Glickman & Sycara, 2000). It appears to be similar to Case 1 of section 5.2, where a strong selective pressure for $\hat{M}$ genotypes due to lower values $p_{\text{mut}}^{[M]}(\hat{M})$ and a resulting higher modularity was observed. It could be an interesting question, whether premature convergence can be interpreted as a special form of Self-Organized Modularization.

**Homogeneity-effects in the LLGA** In (Harik & Goldberg, 1996), the authors develop a kind of genome where a given phenotype can be achieved by various representations of “different linkage”. Some simple and exemplary considerations in (Dauscher, 2003) show that different representations (having different linkage) correspond to different values of Modularity Matrix Elements.

Furthermore, the authors report that genotypes with a strong linkage do not – as expected – predominate the population at later timesteps. This is very similar to the reduction of selective pressure in Case 2 of section 5.2. This suggests that the decreasing selective pressure LLGA is due to diminishing differences in Effective Modularity. A measurement of Effective Modularity in LLGA approaches are most likely to yield interesting results.

**Effective Modularity of schemata in Simple Genetic Algorithms** As already mentioned in 2.7, classes of respected equivalence relations correspond to schemata in Simple Genetic Algorithms. The well-known Schema theorem (Holland, 1975), once considered to be the basis of the power exhibited by Genetic Algorithms, has been reconsidered more skeptically, especially in the 1990s (e.g. (Altenberg, 1995b), (Polani, 1996)). In (Altenberg, 1995b), the author explicitly criticizes the “absence of a demonstration of the relevance of schemata”. Indeed, the Schema Theorem itself does not explain a difference between schemata and other equivalence classes.

The results from the MetaEA simulations, however, suggest that the respect property of an equivalence relation may strongly influence evolutionary dynamics. Especially the establishment of a “vicious circle” – as sketched in this paper – is closely connected to the respect property. Therefore, there might be a possibility that Schemata play an outstanding role as Building Blocks compared to other, arbitrary subsets. Further simulation experiments (as well as theoretical investigations) are currently performed in order to find out to what extent the MetaEA results can be transferred to the special case of Simple Genetic Algorithms.
Appendix: Derivation of $n_{jk}$ in the Two-Level System

In this two-level system, $n_{jk}$ can be rewritten as follows:

$$n_{jk} = \sum_i w_i \cdot T(i \leftarrow j, k) =$$

$$= \sum_i \left[ \alpha \cdot c_i^j + \beta \right] T(i \leftarrow j, k) =$$

$$= \alpha \sum_i T(i \leftarrow j, k) \cdot c_i^j + \beta \sum_i T(i \leftarrow j, k) =$$

$$= \alpha \sum_i T(i \leftarrow j, k) c_i^j + \beta c,$$

where we give the name $\sigma_{jk}$ to the sum. Furthermore (for the sake of simplicity), we will omit the ”$\psi$” as the argument of the functions $r_{i,jk}(\psi)$ etc.

Before we try to express the term $\sigma_{ij}$ in terms of $m_{jk}$, we consider the functions $r_{ij}$, $r_{ik}$ and $r_{i,jk}$ for three genotypes $i, j$ and $k$.

**Case (EQUIV)** Genotypes $j$ and $k$ are equivalent w.r.t. $\psi$, i.e. $r_{jk} = 1$. Because of the transitivity property of equivalence relations

$$\forall i : r_{ij} = r_{ik} = r_{i,jk} \quad (47)$$

holds, and therefore

$$\sum_i T(i \leftarrow j, k)r_{ij} = \sum_i T(i \leftarrow j, k)r_{i,jk} = \sum_i T(i \leftarrow j, k)r_{ik} \quad (48)$$

**Case (NONEQUIV)** If $j$ and $k$ are not equivalent w.r.t. $\psi$ i.e. $r_{jk} = 0$. We make use of the fact that

$$\forall i : r_{i,jk} = r_{ij} + r_{ik} - r_{ij} \cdot r_{ik} \quad (49)$$

always holds.

For this case $r_{jk} = 0$, the product term cannot equal 1 because of the transitivity property. Therefore

$$\sum_i T(i \leftarrow j, k)r_{i,jk} = \sum_i T(i \leftarrow j, k)r_{ij} + \sum_i T(i \leftarrow j, k)r_{ik} \quad (50)$$

holds.

We make the following (basic) symmetry assumption: For most crossover and mutation operators, given a fixed pair of parent individuals of genotypes $j$ and $k$, an offspring’s probability of being equivalent to $j$ is the same as its probability of being equivalent to $k$.

Thus we have

$$\sum_i T(i \leftarrow j, k)r_{ij} = \sum_i T(i \leftarrow j, k)r_{ik} \quad (51)$$
In combination with Eq. (50), this leads to
\[ \sum_i T(i \leftarrow j, k) r_{ij} = \sum_i T(i \leftarrow j, k) r_{jk} = \frac{1}{2} \sum_i T(i \leftarrow j, k) r_{i,jk} \] (52)

Now we return to our original task to rewrite
\[ \sigma_{jk} = \sum_i T(i \leftarrow j, k) c^\uparrow_i \] (53)

We will do this separately for all 4 possible combinations of \( c^\uparrow_j \) and \( c^\uparrow_k \):

**Case (1,0) \( c^\uparrow_j = 1, c^\uparrow_k = 0 \)**

One can easily see that
\[ c^\uparrow_i = r_{ij} \] (54)

holds and therefore also
\[ \sigma_{jk} = \sum_i T(i \leftarrow j, k) r_{jk} \] (55)

Furthermore, in this Case (1,0) \( j \) und \( k \) cannot be equivalent and therefore, we are in the above Case (NONEQUIV):
\[ \sigma_{jk} = \frac{1}{2} \sum_i T(i \leftarrow j, k) r_{i,jk} \] (56)

Due to the special setting of \( c^\uparrow_j \) and \( c^\uparrow_k \), this can be rewritten as follows:
\[ \sigma_{jk} = \frac{1}{2} (c^\uparrow_i + c^\uparrow_k) \sum_i T(i \leftarrow j, k) r_{i,jk} \] (57)

**Case (0,1) \( c^\uparrow_j = 0, c^\uparrow_k = 1 \)**

Because of symmetry, similarly to Case (1,0)
\[ \sigma_{jk} = \frac{1}{2} (c^\uparrow_i + c^\uparrow_k) \sum_i T(i \leftarrow j, k) r_{i,jk} \] (58)

holds.

**Case (1,1) \( c^\uparrow_j = 1, c^\uparrow_k = 1 \)**

Here both parent genotypes are equivalent
\[ c^\uparrow_i = r_{ij} = r_{ik} \] (59)

Since this case corresponds to the Case (EQUIV), this leads to
\[ \sigma_{jk} = \sum_i T(i \leftarrow j, k) r_{i,jk} \] (60)

and because of the setting of \( c^\uparrow_j \) and \( c^\uparrow_k \), this can be rewritten as
\[ \sigma_{jk} = \frac{1}{2} (c^\uparrow_i + c^\uparrow_k) \sum_i T(i \leftarrow j, k) r_{i,jk} \] (61)

which is identical to the equations in cases (1,0) and (0,1).
Case (0,0) \( e_j^0 = 0, e_k^0 = 0 \)

For this case, we have assumed that (because of the large number of equivalence classes) the probability for an offspring to be in equivalence class \( \xi \) is negligible.

\[
\sigma_{jk} = \sum_i T(i \leftarrow j, k)c_i^0 = 0 = \frac{1}{2} (c_i^1 + c_k^1) \sum_{i=0} T(i \leftarrow j, k)r_{i,jk}
\]

(62)

Thus, all four cases can be subsumed in one equation:

\[
\sigma_{jk} = \frac{1}{2} (c_i^1 + c_k^1) \sum_i T(i \leftarrow j, k)r_{i,jk} = \frac{1}{2} (c_i^1 + c_k^1)m_{jk}(\psi)
\]

(64)

For \( n_{jk} \) one therefore obtains

\[
n_{jk} = \alpha \cdot \sigma_{jk} + \beta = \frac{\alpha}{2} (c_i^1 + c_k^1)m_{jk}(\psi) + \beta,
\]

(67)

which is used in the further considerations.

Acknowledgments

The authors wish to thank the anonymous reviewers for their helpful comments and suggestions. Furthermore, we thank Micahela Brauburger and Chris Goodson whose thorough proof-reading has definitely enhanced the understandability of the text.

References


