of bounded difficulty, we demand high-quality representations to have low locality as only this guarantees that the difficulty of the problem f_p remains unchanged. Finally, we discuss why, in general, it is not possible to create representations that both, preserve complexity for easy problems, and reduce complexity for difficult problems.

Section 3.1 shows how the usage of redundant encodings affects genetic search. Based on the Gambler's ruin model (Harik et al. 1997), we develop a quantitative model of redundancy and verify it empirically for the trivial voting mapping, which is a synonymously redundant encoding. In Sect. 3.2, we show how the behavior of GEAs changes for exponentially scaled encodings by using the existing models of genetic drift and population sizing. We use two different drift models and develop models for the necessary population size and the convergence time that allows us to predict the solution quality more accurately than the previous models. To verify the theoretical models, we present an empirical investigation into the performance of GEAs using exponentially scaled representations. Section 3.3 shows that only representations with perfect locality guarantee that phenotypically easy problems remain genotypically easy and can still be solved using GEAs. If the locality of a representation is not perfect, the size and length of the BBs can be different for the genotypes and phenotypes and the complexity of the problem is changed. Fully easy problems can only become more difficult, and fully difficult problems can only become easier to solve for GEAs. The chapter ends with concluding remarks.

3.1 Redundancy

This section provides the first of three elements of a theory of representations. It identifies redundancy to be important for the design of representations, distinguishes between synonymously and non-synonymously representations, and uses existing complexity models to characterize the effect of redundancy on encodings. Furthermore, it presents theoretical models on how the population size, run duration and overall problem difficulty is influenced by synonymously redundant encodings. The model is used for the analysis of the trivial voting mapping, which is a synonymously redundant encoding.

3.1.1 Redundant Representations and Neutral Networks

Information theory provides us with a measurement of information. The information content[1] (measured in Bits) of a sequence is defined as the number of bits required to represent a given number of s possibilities using an optimal encoding (Shannon 1948; Shannon and Weaver 1949). It is calculated as $\log_2(s)$. Redundant encodings are less efficient codings that require more bits

[1] other notations are information, self-information, entropy, or Shannon entropy.

to represent the information but do not increase the amount of information represented.

For encoding one Bit of information content (for example the two possibilities 0 and 1) a binary string of at least length one is necessary (one bit). However, it is also possible to encode one Bit of information content using a bitstring of length $l > 1$. Then, more than one bit of the bitstring encodes one Bit of information, and the representation becomes redundant. We want to emphasize that it is important to distinguish between the amount of information (**B**it) that should be represented and the number of **b**its in a string that are used to represent the information. Redundant representations are encodings where the amount of encoded information (in Bit) is lower than the used number of bits. This means, that such encodings use a higher number of alleles for encoding phenotypic information in the genotype than is necessary for constructing the phenotype. Although the practice of redundancy has steadily increased over the last few years, there is little theory regarding the influence of redundant representations on the performance of GEAs.

Natural Selection and Neutral Theory: Different Concepts for Explaining Evolution

Examining the use of redundant representations reveals that redundant representations are not solely an invention of evolutionary computation researchers, but are commonly used in nature for the encoding of genetic information. Currently, in biology different opinions exist regarding the basic concepts that underly the process of evolution in nature and the role of representations therein. *Darwinism* goes back to Darwin (1859) and assumes that *natural selection* is the driving force of evolution (compare Mayr (1991)) and that random genetic drift is unimportant. Randomly advantageous mutations are fixed due to natural selection and can then be propagated from generation to generation. Genetic changes are a result of selection combined with variation operators such as crossover and random mutations. During the process of evolution the variation operators sometimes result in fitter individuals, which gradually replace less-fit individuals in the population.

The theory of natural selection has been extended and modified by the *neutral theory* which was proposed by Kimura (1983). It assumes that the driving force of molecular evolution is the random fixation of neutral mutations rather than the fixation of advantageous mutations by natural selection. Kimura observed that in nature the number of different genotypes which store the genetic material of an individual greatly exceeds the number of different phenotypes which determine the outward appearance. Therefore, the representation which describes how the genotypes are assigned to the phenotypes must be redundant, and neutral mutations become possible. A mutation is neutral if its application to a genotype does not result in a change of the corresponding phenotype. Because large parts of the genotype have no actual effect on the phenotype, evolution can use them as a store for genetic

information that was necessary for survival in the past, and as a playground for developing new properties of the individual that could be advantageous in the future. Neutral mutations are the tool for designing these new properties without interfering with the current phenotype. Although most of the mutations are neutral, some sometimes have an effect on the phenotype and bring new genetic material which was developed by neutral mutations into life. The neutral theory was highly disputed shortly after its formulation and the relative importance of neutral mutation and selection is still unclear. However, the importance of random genetic changes has generally been accepted in population genetics during the last few years.

Redundant Representations in Evolutionary Computation Research

Following the work of Kimura, some biological studies (Huynen et al. 1996; Huynen 1996; Schuster 1997; Reidys and Stadler 1998) focused on the neutral theory. These studies showed that the connectivity between fitness landscapes can be increased by the introduction of redundant representations and neutral mutations. Different genotypes which are assigned to the same phenotype (neutral sets) allow a population to move through the search space more easily and to find new advantageous areas of the search space that would not have been accessible without neutral mutations. Surprisingly, the neutral theory became even more popular in the field of genetic and evolutionary computation (Banzhaf 1994; Dasgupta 1995). There is great interest in how redundant representations and neutral search spaces influence the behavior, and especially the evolvability of GEAs (Barnett 1997; Barnett 1998; Shipman 1999; Shipman et al. 2000; Shackleton et al. 2000; Shipman et al. 2000; Ebner et al. 2001; Smith et al. 2001; Smith et al. 2001a; Smith et al. 2001b; Barnett 2001; Yu and Miller 2001; Yu and Miller 2002; Toussaint and Igel 2002). The general idea behind most of this work is that the evolvability of a population, which is defined as the ability of random variations to sometimes produce improvements, is increased by the use of redundant representations. Furthermore, because redundant representations allow a population to change the genotype without changing the phenotype, the ability of a population to adapt after changes and the performance of GEAs should increase.

However, in most of this work the focus has not been on the performance of GEAs, but on characteristics of the search like reachability of phenotypes, evolvability of populations, or connectivity of search spaces. No results have been presented up till now that clearly indicate the superiority of redundant representations and neutral search on practical test problems or real-world instances. Recently, Knowles and Watson (2002) presented an investigation into the performance of neutral search for NK landscapes, H-IFF, and MAX-SAT problems. The results showed that using arbitrary redundant representations (random boolean network mapping) does not increase the performance of

mutation-based search for the considered test problems. In most of the problems used, adding redundancy appeared to reduce performance.

Although, at the moment, the focus in investigating the role of redundant representations is mainly on neutral mutations and their effects on search characteristics, there is other work which tries to address the effects of redundancy on the performance of evolutionary search. Researchers used different types of redundant representations and sometimes observed either an increase or a decrease in the performance of GEAs. Over time, different opinions regarding the effects of redundancy on the performance of GEAs have been developed. Some work noticed that redundant representations lead to a reduction in GEA performance (Davis 1989; Eshelman and Schaffer 1991; Ronald et al. 1995). The low performance was argued to be either due to a loss of diversity in the population, or because different genotypes that represent the same phenotype compete against each other. Also, the larger size of the search space was listed as a reason for lower GEA performance.

In contrast, other mostly application-oriented work reports higher performance with additional redundancy (Cohoon et al. 1988; Gerrits and Hogeweg 1991; Julstrom 1999), which some researchers ascribe to an increase in diversity that hinders premature convergence. Further work considered the computational implications of genetic code-like representations in gene expression (Kargupta 2000b; Kargupta 2001). Kargupta investigated how redundant representations influence the energy of the Fourier spectrum. The results show that using redundant representations and encoding phenotypes with higher fitness by a larger number of genotypes results in a higher energy of the Fourier spectrum, reduces the difficulty of the optimization problem, and therefore allows a more effective evolutionary search.

This short literature review has shown that the influence of redundant representations on the performance of GEAs is a strongly disputed topic. It can be expected that there is no easy and general answer, and not all types of redundant representations will be useful (Harvey and Thompson 1997). To find answers, it is necessary to characterize the different types of redundant representations regarding their specific properties, and to develop quantitative models describing how solution quality and run duration of GEAs is influenced. This approach can help to clear up some of the disputed questions and to find out under which circumstances which type of redundant representation can be beneficial for GEAs. Consequently, in the following section we develop a classification for different types of redundant representations and in Sects. 3.1.4 and 3.1.5 we develop quantitative models.

3.1.2 Synonymously and Non-Synonymously Redundant Representations

We give some basic definitions and develop a classification for different types of redundant representations which is based on their synonymity.

As we have discussed in Sect. 2.1.2 we have to distinguish between genotypes and phenotypes. Φ_g is defined as the genotypic search space, where the operators crossover and mutation are applied. Φ_p is the phenotypic search space. The fitness of an individual depends on the properties of the phenotype $x^p \in \Phi_p$. A representation $f_g : \Phi_g \rightarrow \Phi_p$ determines which phenotypes $x^p \in \Phi_p$ are represented by which genotypes $x^g \in \Phi_g$. We want to assume that every phenotype x^p is assigned to at least one genotype x^g. Otherwise, if a phenotype x^p is not represented by some genotype x^g this solution can never be found by the used optimization algorithm.

A representation f_g is redundant if the size of the genotypic search space is larger than the size of the phenotypic search space, $|\Phi_g| > |\Phi_p|$. This means, there are more different genotypes than phenotypes. When using search spaces where not all possible phenotypes or genotypes are accessible by the used search method, a representation is redundant if the number of accessible phenotypes is smaller than the number of accessible genotypes. Therefore, in general a representation f_g is redundant if on average one accessible phenotype is represented by more than one genotype. Redundant representations are less efficient encodings, which use an additional number of genes, but do not increase the encoded information content. Therefore, a representation is redundant if l different phenotypes are assigned to m different genotypes where $m > l$. Although the larger number of possible genotypes would allow us to encode more individuals than there are phenotypes, some of the information that exists in the genotypes is not considered.

Distinguishing between Synonymously and Non-Synonymously Redundant Representations

To classify different types of redundant representations we want to measure how similar the genotypes are that are assigned to the same phenotype. A representation is defined to be *synonymously redundant* if the genotypes that are assigned to the same phenotype are similar to each other. Consequently, we denote a representation to be *non-synonymously redundant* if the genotypes that are assigned to the same phenotype are not similar to each other. Therefore, the synonymity of a representation depends on the metric that is defined on Φ_g and Φ_p. The metric defined on Φ_p depends on the properties of the considered problem and the metric defined on Φ_g depends on the used search operator. Depending on different operators and metrics used on Φ_g we get different synonymity of the representation f_g. In Fig. 3.1, we illustrate the differences between synonymous and non-synonymous redundancy. For this illustrative example we use the Euclidean distance between the individuals for indicating how similar different individuals are.

We want to formalize the classification into synonymously and non-synonymously redundant representations. In general, a redundant representation f_g assigns a phenotype x^p to a set of different genotypes $x^g \in \Phi_g^{x^p}$, where $\forall x^g \in \Phi_g^{x^p} : f_g(x^g) = x^p$. All genotypes x^g in the genotypic set $\Phi_g^{x^p}$ represent

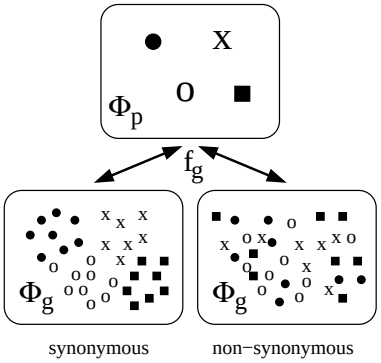


Figure 3.1. Synonymous versus non-synonymous redundancy. The different symbols indicate different genotypes and their corresponding phenotypes. When using synonymously redundant representations (left), genotypes that represent the same phenotype are similar to each other. When using non-synonymously redundant representations (right), genotypes that represent the same phenotype are not similar to each other but distributed over the whole search space

the same phenotype x^p. A representation is synonymously redundant if the genotypic distances between all $x^g \in \Phi_g^{x^p}$ are small for all different x^p. Therefore, if for all phenotypes the sum over the distances between all genotypes that correspond to the same phenotype

$$\left(\sum_{x^p} \frac{1}{2} \left(\sum_{x^g \in \Phi_g^{x^p}} \sum_{y^g \in \Phi_g^{x^p}} d(x^g, y^g) \right) \right), \tag{3.1}$$

where $x^g \neq y^g$, is reasonably small a representation is denoted to be synonymously redundant. $d(x^g, y^g)$ depends on the mutation operator used and measures the distance between two genotypes $x^g \in \Phi_g^{x^p}$ and $y^g \in \Phi_g^{x^p}$ which both represent the same phenotype x^p. The distance between two genotypes depends on their genotypic similarity and is small if the two genotypes are similar.

A different but equivalent approach of defining the synonymity of redundant representations is to use equivalence classes (compare the approach from Radcliffe outlined in Sect. 2.4.2). All genotypes x^g in the genotypic set $\Phi_g^{x^p}$ belong to the same equivalence class. If the sum of the genotypic distances between the individuals that belong to the same equivalence class is small, then the representation is synonymously redundant.

Synonymity and Locality of Redundant Representations

The synonymity of redundant representations is related to the locality of non-redundant representations (compare Sect. 3.3). A genotype x^g is a neighbor

to some other genotype y^g if the distance $d(x^g, y^g) = d_{min}$, where $d_{min} \neq 0$ is the minimal distance between two individuals in the genotypic search space. When using binary representations, $d_{min} = 1$ and two genotypes are neighbors if they differ in one allele. As discussed in Sect. 3.3, the locality of a representation describes how well neighboring genotypes correspond to neighboring phenotypes. If neighboring genotypes correspond to neighboring phenotypes, a representation has high locality and small changes in the genotype result in small changes in the corresponding phenotype. In contrast, representations have low locality if neighboring genotypes do not correspond to neighboring phenotypes. There is evidence, both analytical (for example Whitley (1999) for mutation-based search or Sect. 3.3.6 for crossover-based search), and empirical (for example Gottlieb et al. (2001) or Rothlauf and Goldberg (2000)), which shows for easy problems that low-locality representations result in low GEA performance. The genetic operators mutation and crossover no longer work properly as they create new offspring that are not similar to their parent(s).

Low-locality representations result in low GEA performance as guided search methods like GEAs, that use knowledge gained during search for determining future search steps, can only perform better than random search if on average similar solutions have similar fitness (individuals are similar if there are only a few search steps between them). In general, guided search methods assume that in the neighborhood of high-quality solutions other high-quality solutions can be found (Manderick et al. 1991; Horn 1995; Deb et al. 1997; Christensen and Oppacher 2001). High-quality solutions are grouped together and are not scattered over the whole search space (Radcliffe 1991a; Whitley 2002). Therefore, to perform well, guided search methods have to search more often in the neighborhood of already found promising high-quality solutions than around low-quality solutions (compare the optimal allocation of trials for the two-armed bandit problem as discussed in Holland (1975)). This behavior guarantees high performance of the search method if on average neighboring solutions have similar properties, which means the fitness values of neighboring solutions are correlated.

However, search heuristics could not use any information learned during the search for determining future search steps, and consequently show low performance if the fitness values of neighboring, or similar, genotypes are not correlated (Weinberger 1990; Manderick et al. 1991; Jones and Forrest 1995). On the one hand, the fitness values of neighboring genotypes can be uncorrelated if the problem itself is difficult, which means the fitness values of neighboring phenotypes are uncorrelated. Then, guided search methods will behave as a random search even in the presence of high-locality representations. On the other hand, the fitness values of neighboring genotypes are also uncorrelated if low-locality representations are used and neighboring genotypes do not correspond to neighboring phenotypes. The low-locality representations destroy existing correlations between phenotypes and their corresponding fitness values, and genotypic neighbors no longer have similar properties and fitnesses.

In this case, search heuristics can not use any information learned during the search for determining future search steps. As a result, it makes no sense for guided search approaches to search around already found high-quality genotypes and guided mutation-based search algorithms become random search. A mutation does not result in a solution with similar properties but in a random solution. Analogously, crossover is not able to create new solutions with similar properties to their parents, but creates new, random solutions. Therefore, high locality representations are a necessity for efficient evolutionary search. When using low-locality representations, no guided search is possible and guided search methods become random search.

These concepts can also be applied to redundant representations. When using non-synonymously redundant representations, genetic operators like mutation or crossover can result in an offspring that is phenotypically completely different from its parent(s). Therefore, non-synonymously redundant representations have the same effect on GEAs as when using low-locality representations. Using neutral search spaces where the connectivity between the phenotypes is strongly increased by the use of a redundant representation allows us to reach many different phenotypes in one single search step. However, increasing the connectivity between the phenotypes by using non-synonymously redundant representations results in random search and decreases the efficiency of GEAs. As for low-locality representations, a search step does not result in a similar phenotype but creates a randomly chosen individual. Therefore, guided search is no longer possible and guided search methods become random search. As a result, we get reduced GEA performance on problems that are easy for guided search methods (that means the fitness values of similar phenotypes are correlated) when using non-synonymously redundant representations. Examples for non-synonymously redundant representations are the direct binary mapping, the cellular automaton mapping, or the random boolean network mapping, which have been proposed by Shackleton et al. (2000). Although the use of these types of representations strongly increases the connectivity between phenotypes, we get low GEA performance as neighboring genotypes do not correspond to neighboring phenotypes. Initial evidence of the low performance of mutation-based search when using such non-synonymously redundant representations was shown by Knowles and Watson (2002) for the random boolean network mapping.

In contrast, when using synonymously redundant representations, the connectivity between the phenotypes is not increased. Therefore, small genotypic variations can not result in large phenotypic changes but either in the same, or a similar, phenotype. Figure 3.2 illustrates this behavior and compares it to non-synonymously redundant representations. Examples for synonymously redundant representations are the trivial voting mapping (Shackleton et al. 2000) which is investigated more closely in Sect. 3.1.6.

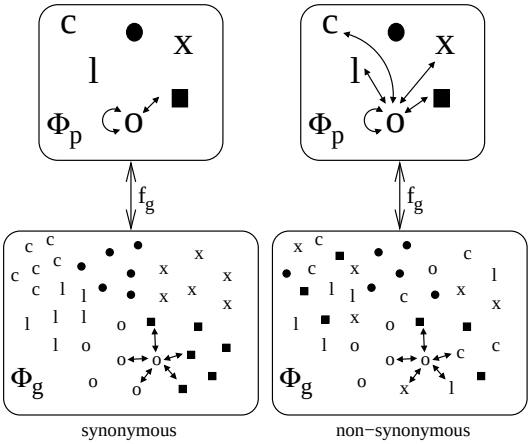


Figure 3.2. The effects of small mutation steps for synonymously versus non-synonymously redundant representations. The different symbols indicate different genotypes and their corresponding phenotypes. The arrows indicate search steps which result in neighboring individuals. When using synonymously redundant representations, a mutation results in either the same or a similar phenotype. In contrast, when using non-synonymously redundant representations the mutation of a genotype results in completely different phenotypes.

Formalizing Synonymously Redundant Representations

In this subsection, we introduce some quantities that can be used for characterizing the properties of synonymously redundant representations. We use the definitions from Sects. 2.1.2 and 3.1.2.

To describe a redundant representation, we introduce k_r, the *order of redundancy*. k_r is defined as $\log(|\Phi_g|)/\log(|\Phi_p|)$ and measures the amount of redundant information in the encoding. There are k_r bits and 2^{k_r} different possibilities (individuals) to encode 1 Bit of information. When using binary genotypes and binary phenotypes, the order of redundancy can be calculated as

$$k_r = \frac{\log(2^{l_g})}{\log(2^{l_p})},$$

where l_g is the length of the binary genotype and l_p is the length of the binary phenotype. When using a non-redundant representation, the number of genotypes equals the number of phenotypes and $k_r = 1$.

Furthermore, we want to characterize not only to what degree a representation is redundant, but also in what way it is redundant. We are especially interested in the overrepresentation and underrepresentation of specific solutions. Therefore, we introduce r as the number of genotypes that represent the one phenotype that has the highest fitness. When using non-redundant representations, every phenotype is assigned to exactly one genotype and $r = 1$. However, in general, $1 \leq r \leq |\Phi_g| - |\Phi_p| + 1$.

In the following discussion, we want to focus on how redundant representations influence the behavior of selectorecombinative GAs. Selectorecombinative GAs use crossover as the main search operator and mutation only serves as a background operator. When focusing on selectorecombinative GAs we implicitly assume that there are building blocks (BBs) and that the GA process schemata. Consequently, we must define how k_r and r depends on the properties of the BBs.

In general, when looking at BBs of size k there are 2^k different phenotypic BBs which are represented by 2^{kk_r} different genotypic BBs. Therefore,

$$k_r = \frac{k_g}{k_p},$$

where k_g denotes the genotypic size of a BB and k_p the size of the corresponding phenotypic BB. As before, a representation is redundant if $k_r > 1$. The size of the genotypic BBs is k_r times larger than the size of the phenotypic BB. Furthermore, r is defined as the number of genotypic BBs of length kk_r that represent the best phenotypic BB of size k. Therefore, in general,

$$r \in \{1, 2, \ldots, 2^{kk_r} - 2^k + 1\}. \tag{3.2}$$

In contrast to k_r, which is determined by the representation used, r depends not only on the representation used, but also on the specific problem that should be solved. Different instances of a problem result in different values of r. If we assume that k_r is an integer (each phenotypic allele is represented by k_r genotypic alleles) the possible values of the number of genotypic BBs that represent the optimal phenotypic BB can be calculated as

$$r = i^k, \text{ with } i \in \{1, 2, \ldots, 2^{k_r} - 1\}. \tag{3.3}$$

A representation is *uniformly redundant* if all phenotypes are represented by the same number of different genotypes. Therefore, when using a uniformly redundant representation every phenotypic BB of size $k = k_p$ is represented by

$$r = 2^{k(k_r - 1)} \tag{3.4}$$

different genotypic BBs. Table 3.1 gives an example for a uniformly redundant encoding. Two bits in a phenotype x^p are represented by four bits in the genotype x^g. Therefore, $k_r = 2$ and $r = 4$. With $|\Phi_p| = 2^k = 2^2$ the size of the genotypic space is $|\Phi_g| = 2^{kk_r} = 2^4 = 16$.

x^g	x^p
00 00, 00 01, 01 00, 01 01	0 0
10 00, 10 01, 11 00, 11 01	1 0
00 10, 01 11, 00 11, 01 11	0 1
10 10, 10 11, 11 10, 11 11	1 1

Table 3.1. An example of a uniformly redundant representation, where $k_r = 2$ and $r = 4$

By introducing redundancy the search space for a GA using binary phenotypes of string length $l = l_p$ is increased from $|\Phi_p| = 2^l$ to $|\Phi_g| = 2^{lk_r}$. The length of the individuals increases from $l = l_p$ in the phenotypic space to $l_g = k_r \times l$ in the genotypic space. To represent all phenotypes, each individual $x^p \in \Phi_p$ must be represented by at least one genotype $x^g \in \Phi_g$. If $|\Phi_g| = |\Phi_p|$, and each phenotype is represented by at least one genotype, we have a non-redundant, one-to-one mapping.

3.1.3 Complexity Model for Redundant Representations

For modeling the effects of redundant representations on the performance of GEAs, we can use the complexity model from Goldberg (1991a) and Goldberg et al. (1992). If we want to understand the effects of redundancy, we must decompose the problem into smaller sub-problems and try to solve these separately. We could decompose the problem step by step and subsequently collate all the sub-problems. The decomposition (Goldberg 1998) takes place as follows:

- GAs process building blocks.
- Problems are tractable by BBs.
- GAs must ensure proper supply of BBs in the initial generation.
- GAs must grow the high quality BBs.
- GAs must mix the BBs well.
- GAs must decide well among competing BBs.

This decomposition gives us a framework for investigating the effects of redundancy on the performance of GAs. We want to examine how redundancy affects the problem decomposition point by point:

Using redundant encodings does not change the principal behavior of GAs. After adding redundancy, GAs still process building blocks.

A problem is still tractable by building blocks when using a redundant encoding. However, the question arises as to whether the size of building blocks is changed by redundant encodings. On one hand, redundant representations increase the number of bits that are part of a building block in the genotype from k to kk_r. On the other hand, the number of building blocks in the genotype that represent the same BB in the phenotype increases from 1 to on average $2^{k(k_r-1)}$. Furthermore, the number of different fitness values that can be assigned to the genotypes remains constant. Taking these effects into account, we assume that problems are still tractable by BBs and the larger size of the genotypic BBs is compensated by the higher number of genotypic BBs.

How do redundant encodings change the initial supply of BBs in the initial population? For uniform redundancy (every phenotype $x^p \in \Phi_p$ is represented by the same number of genotypes $x^g \in \Phi_g$), the initial supply of BBs $x_0/N = 1/2^k$ is the same as for non-redundant representations. If the number of genotypes x^g that represent a phenotype x^p is above average, then

the phenotype x^p, and the containing schemata h^p, are overrepresented in the initial population. GEAs are pushed more towards solutions that are similar to these x^p. Analogously, the performance of GEAs decreases if the proportion of less fit phenotypes in the initial population is increased by redundancy. As a result, the supply of BBs could be modified by redundant encodings.

For a proper growth and mixing of BBs, above average BBs must be preferred by selection, and BBs should not be disrupted by the crossover operator. As the genotypic defining length of a BB $\delta(h^g)$ increases when using redundant representations, GEA operators that do not obey the linkage disrupt BBs more frequently. To overcome this problem, competent GAs (Mühlenbein and Paaß 1996; Goldberg 1999; Larranaga et al. 1999; Mühlenbein and Mahnig 1999; Pelikan 2002) could be used. These kind of GEAs obey the linkage, and genotypic building blocks h^g are not disrupted by recombination. Thus, no reduction of performance should occur, and redundancy should have no negative effect on the proper mixing of BBs. However, when using redundant representations there are different genotypes x^g and y^g that represent the same phenotype x^p. Recombining x^g and y^g could result in offspring that do not represent x^p. This effect is also known as cross-competition among isomorphic identical BBs. As an example for a one-bit problem, the genotype $\{00\}$ represents the phenotype $x^p = 0$, and the genotypes $\{01, 10, 11\}$ represent the phenotype $y^p = 1$. If the genotypes 01 and 10 are recombined, the possibilities for the offspring are 00 and 11. Although both parental genotypes represent the same phenotype y^p, one of the offspring represents x^p. Cross-competition among isomorphic identical BBs is a result of using non-synonymously redundant representations, which were discussed in the previous subsection. Non-synonymously redundancy randomizes genetic search as new BBs are introduced into the search that did not exist in the parents. When using such representations, the recombination of genotypes that represent the same phenotype can result in completely different new genotypes and phenotypes (compare also Fig. 3.2). When using synonymously redundant representations, cross-competition can not occur.

Finally, the decision making between competing BBs is not affected by using redundant representations. With redundancy there are different genotypic BBs h^g that represent the same phenotypic BB h^p, but the selection process does not decide between the different h^g because they all have the same fitness. The fitness evaluation is based on the fitness of the phenotypic BBs h^p, and not their genotypic representation.

After recognizing that redundancy could have a major effect on the supply, and a minor effect on the proper mixing of building blocks, we want to quantify its effect on BBs supply in the following subsection.

3.1.4 Population Sizing for Synonymously Redundant Representations

In Sect. 3.1.2, we described how non-synonymously redundant representations result in randomized search as they increase the connectivity of the search space. Therefore, in this section we want to focus on synonymously redundant representations and develop a population sizing model that describes their influence on the performance of selectorecombinative GAs.

As we focus in our investigation on selectorecombinative GAs we can use the existing theory describing the behavior of selectorecombinative GAs from Harik et al. (1997) and Thierens and Goldberg (1994). They describe for non-redundant representations how the population size and the time to convergence that is necessary to solve a specific problem depend on the characteristics of the problem.

Following Harik et al. (1997) the probability that a GA with a population size N converges after t_{conv} generations to the correct solution is

$$P_n = \frac{1-(q/p)^{x_0}}{1-(q/p)^N},$$

where x_0 is the expected number of copies of the best BB in the randomly initialized population, $q = 1-p$, and p is the probability of making the right choice between a single sample of each BB

$$p = \mathbb{N}\left(\frac{d}{\sqrt{2m'}\sigma_{BB}}\right). \tag{3.5}$$

$\mathbb{N}$ is the cumulative distribution function for a normal distribution, d is the signal difference between the best BB and its strongest competitor, $m' = m-1$ with m is the number of BBs in the problem, σ^2_{BB} is the variance of a BB, and $q = 1-p$ is the probability of making the wrong decision between two competing BBs. It has been shown in Harik et al. (1997) that this random walk or Gambler's ruin model can be used for describing the behavior of selectorecombinative GAs propagating schemata and BBs. In the following paragraphs, this model is the basis for describing the influence of synonymously redundant representations on the behavior of GAs.

For a randomly initialized population with no redundancy, $x_0 = N/2^k$. The situation changes when using redundant representations. Then, the initial supply depends on the characteristics of the representation, namely r and k_r. With r the number of genotypic BBs of length kk_r that represent the best phenotypic BB of length k, we get

$$x_0 = N\frac{r}{2^{kk_r}}, \tag{3.6}$$

where k_r is the order of redundancy. The assumption that redundant representations affect the initial supply of BBs is the core idea behind the proposed

model describing the influence of synonymously redundant representations on GA performance. We assume that other effects of synonymously redundant representations on GA performance can be neglected. Consequently, when using uniformly redundant representations, $r = 2^{k(k_r-1)}$ and $x_0 = N/2^k$. These are the same values as when using non-redundant representations. Therefore, GA performance does not change when using uniformly redundant representations.

As the variance σ^2_{BB} and the number m of BBs is not affected by the use of a redundant representation, the probability of GA failure $\alpha = 1 - P_n$ can be calculated as

$$\alpha = 1 - \frac{1 - (q/p)^{x_0}}{1 - (q/p)^N}. \tag{3.7}$$

If we assume that x_0 is small and $q < p$ we can assume that $1 - (q/p)^N$ converges to 1 faster than $1 - (q/p)^{x_0}$. Using these approximations (see also Harik et al. (1997)) the equation can be simplified to

$$\alpha \approx \left(\frac{1-p}{p}\right)^{x_0}.$$

Therefore, for the population size we get

$$N \approx \frac{2^{kk_r}}{r}\left(\frac{\ln(\alpha)}{\ln\left(\frac{1-p}{p}\right)}\right). \tag{3.8}$$

The normal distribution in (3.5) can be approximated using the first two terms of the power series expansion (see Abramowitz and Stegun (1972)) as $\mathbb{N}(x) \approx 1/2 + x/2$, where $x = d/\sqrt{\pi m'}\sigma_{BB}$. Substituting p from (3.5) into (3.8) we get:

$$N \approx \frac{2^{kk_r}}{r}\ln(\alpha)/\ln\left(\frac{1-x}{1+x}\right),$$

Since x is a small number, $\ln(1-x)$ can be approximated with $-x$ and $\ln(1+x)$ with x. Using these approximations we finally get for the population size N:

$$N \approx -\frac{2^{k_r k-1}}{r}\ln(\alpha)\frac{\sigma_{BB}\sqrt{\pi m'}}{d}. \tag{3.9}$$

The population size N goes with $O\left(\frac{2^{k_r}}{r}\right)$ when using synonymously redundant representations. With increasing r the number of individuals that are necessary to solve a problem decreases. Using a uniformly redundant representation, where $r = 2^{k(k_r-1)}$, does not change the population size N in comparison to non-redundant representations.

3.1.5 Run Duration and Overall Problem Complexity for Synonymously Redundant Representations

To describe the performance of GAs, we must calculate not only the number of individuals that are necessary for solving a problem, but also the expected number of generations until convergence.

Based on Mühlenbein and Schlierkamp-Voosen (1993) and Thierens and Goldberg (1994), Miller and Goldberg developed a convergence model for selectorecombinative GAs (Miller and Goldberg 1996b; Miller and Goldberg 1996a). The convergence time t_{conv} depends on the length of the phenotypes $l = l_p$ and the used selection scheme. Using the selection intensity I the convergence model is

$$p(t) = 0.5 \left(1 + \sin\left(\frac{It}{\sqrt{l}} + \arcsin(2p(0) - 1)\right)\right),$$

where $p(0) = x_0/N$ is the proportion of best building blocks in the initial population. I depends only on the used selection scheme. The number of generations t_{conv} it takes to fully converge the population can be calculated by putting $p(t_{conv}) = 1$:

$$t_{conv} = \frac{\sqrt{l}}{I}\left(\frac{\pi}{2} - \arcsin\left(2p(0) - 1\right)\right). \tag{3.10}$$

If we assume $k = 1$ and uniform redundancy (equal proportion of 1s and 0s in the initial population) we get $p(0) = 0.5$. Then, the number of generations until convergence simplifies to

$$t_{conv} = \frac{\pi}{2}\frac{\sqrt{l}}{I}.$$

With redundancy the initial proportion of building blocks is $p(0) = \frac{r}{2^{k k_r}}$ (see (3.6)). Using $\arcsin(x) = x + o(x^3)$ the time until convergence could be approximated by

$$t_{conv} \approx \frac{\sqrt{l}}{I}\left(1 + \frac{\pi}{2} - \frac{r}{2^{k_r k - 1}}\right). \tag{3.11}$$

With increasing $r/2^{k_r}$ the time to convergence t_{conv} is reduced. Therefore, the optimal solution is found after a lower number of generations if it is overrepresented by the synonymously redundant representation. For uniform redundancy $r = 2^{k(k_r - 1)}$, we get

$$t_{conv} \approx \frac{\sqrt{l}}{I}\left(1 + \frac{\pi}{2} - \frac{1}{2^{k-1}}\right).$$

The time until convergence when using uniformly redundant representations is the same as without redundancy.

After we have calculated the number of individuals that are necessary for solving a problem (see (3.9)), and the number of generations that GAs using only crossover need to converge (see (3.11)), we can calculate the absolute number of fitness calls that are necessary for solving a problem:

$$N \times t_{conv} \approx -\frac{2^{k_r k-1}}{r} \ln(\alpha) \frac{\sigma_{BB}\sqrt{\pi m'}}{d} \times \frac{\sqrt{l}}{I} \left(1 + \frac{\pi}{2} - \frac{r}{2^{k_r k-1}}\right) = \\ = \frac{\sqrt{\pi l m'}}{I} \ln(\alpha) \frac{\sigma_{BB}}{d} \left(1 - \frac{2^{k k_r}}{4r}(2+\pi)\right)$$

The overall number of fitness calls goes with $O(2^{k_r}/r)$. In comparison to non-redundant representations, the number of fitness calls stays constant for synonymously redundant representations if $r = 2^{k(k_r-1)}$. Then $x_0/N = 1/2^k$ and the representation is uniformly redundant.

3.1.6 Analyzing the Redundant Trivial Voting Mapping

In the previous subsection, we developed theoretical models describing how synonymously redundant representations influence the quality of the solution and the time that is necessary to find the good solutions. In this subsection, we investigate whether or not the proposed models allow a good prediction of GA performance for the trivial voting (TV) mapping. The TV mapping is a synonymously redundant representation and we use it for one-max and concatenated deceptive trap problems. During our investigation we are particularly interested in whether the developed models allow us to accurately predict the expected solution quality and running time of a selectorecombinative GA.

The Trivial Voting Mapping

We give a short introduction into the trivial voting mapping.

When using the TV mapping, a set of mostly consecutive, genotypic alleles is relevant for the value of one allele in the phenotype. Each allele in the genotype can only influence the value of one allele in the phenotype. The value of the phenotypic allele is determined by the majority of the values in the genotypic alleles. In general, the different sets of alleles in the genotype defining one phenotypic allele have the same size. The TV mapping is a synonymously redundant representation as all genotypes that represent the same phenotype are similar to each other. A mutation in a genotype results either in the same corresponding phenotype, or in one of its neighbors.

The TV mapping can be easily characterized using the representation parameters defined in Sect. 3.1.2. The order of redundancy k_r is simply the number of genotypic alleles that determine the value of one phenotypic allele. Figure 3.3 gives an example for the TV mapping.

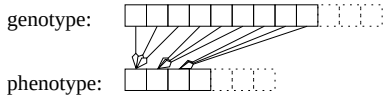


Figure 3.3. The trivial voting mapping

Shackleton et al. (2000) applied the TV mapping to binary strings in the context of the neutral theory. When used for binary strings, binary genotypes $x^g \in \mathbb{B}^{l_g}$ are assigned to binary phenotypes $x^p \in \mathbb{B}^{l_p}$. The length of a genotype is larger than the length of a phenotype, $l_g > l_p$. The value of one phenotypic bit is determined by the majority of the values in the corresponding genotypic bits (majority vote). However, if k_r is even then the number of ones could equal the number of zeros. Therefore, half the cases that result in a tie should encode a one in the corresponding phenotypic allele, and half the cases should represent a zero. For example, for $k_r = 4$ the genotypic BBs 1100, 1010, and 1001 represent a 1 and the genotypic BBs 0011, 0101, 0110 represent a zero.

Because the majority of the votes determines the values of the corresponding phenotypic allele, the TV mapping is a uniformly redundant representation. Each phenotypic BB is represented by the same number of genotypic BBs which is $2^{k(k_r-1)}$, where k is the size of the phenotypic BB.

As we are not only interested in uniformly redundant representations, but also want to know how non-uniformly redundant representations influence GA performance, we extend the TV mapping to allow the encoding to overrepresent some individuals. Therefore, we want to assume that if the number of ones in the genotypic alleles $x^g_{k_r i+j}$, where $i \in \{0,\ldots,l_p-1\}$ and $j \in \{0,\ldots,k_r-1\}$, is larger or equal than a constant u then the value of the phenotypic allele x^p_i is set to one. Vice versa, the phenotypic allele x^p_i is set to zero if less than u of the corresponding genotypic alleles are set to one. Therefore,

$$x^p_i = \begin{cases} 0 \text{ if } & \sum_{j=0}^{k_r-1} x^g_{k_r i+j} < u \\ 1 \text{ if } & \sum_{j=0}^{k_r-1} x^g_{k_r i+j} \geq u, \end{cases}$$

where $u \in \{1,\ldots,k_r\}$. x^g_i (respectively x^p_i) denotes the ith allele of the genotype (respectively phenotype). u can be interpreted as the number of genotypic alleles that must be set to one to encode a one in the corresponding phenotypic allele. This representation is denoted as *extended trivial voting* (eTV) mapping. For $u = (k_r + 1)/2$ (k_r must be odd) we get the original TV mapping. Extending the TV mapping in the proposed way allows us to investigate how non-uniform redundancy influences the performance of GAs.

When using the eTV mapping, the number r of genotypic BBs that can represent the optimal phenotypic BB depends on the number of ones in the genotypic alleles that determine the value of the corresponding phenotypic allele. Considering (3.3) we get

$$r = \left(\sum_{j=u}^{k_r} \binom{k_r}{j} \right)^k , \tag{3.12}$$

where $u \in \{1, \ldots, k_r\}$. We assume that a BB is optimal if all phenotypic bits are set to $x_i^p = 1$. k denotes the size of the phenotypic BB. To give a short illustration, we use a redundant representation with $k_r = 3$, $k = 1$ (compare Fig. 3.3). The optimal BB is $x_i^p = 1$. Because $u \in \{1, \ldots, k_r\}$ there are three different values possible for r. For $u = 1$ the phenotypic allele x_i^p is set to one if at least one of the three corresponding genotypic alleles $x_{ik_r}^g$, $x_{ik_r+1}^g$, or $x_{ik_r+2}^g$ is set to one. Therefore, a one in the phenotype is represented by $r = \sum_{j=1}^{3} \binom{k_r}{j} = 7$ different genotypic BBs (111, 110, 101, 011, 100, 010, and 001). For $u = 2$, the optimal genotypic BB $x_i^p = 1$ is represented by $r = \sum_{j=2}^{3} \binom{k_r}{j} = 4$ different genotypic BBs (111, 110, 101, and 011) and the representation is uniformly redundant. For $u = 2$ we get the original TV mapping. For $u = 3$, the optimal phenotypic BB is represented only by one genotypic BB (111).

Experiments and Empirical Results

Here we present empirical results when using the binary trivial voting mapping for the one-max problem and the concatenated deceptive trap problem.

One-Max Problem

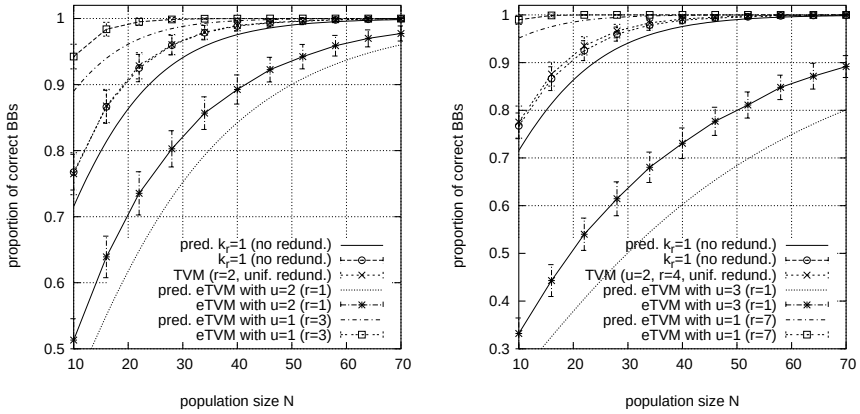
The first test example for our empirical investigation is the one-max problem. This problem is very easy to solve for GEAs as the fitness of an individual is simply the number of ones in the binary phenotype. To ensure that recombination results in a proper mixing of the BBs, we use uniform crossover for all experiments with the one-max problem. Furthermore, in all runs we use tournament selection without replacement and a tournament size of 2. For the one-max function the signal difference d equals 1, the size k of the building blocks is 1, and the variance of a building block $\sigma_{BB}^2 = 0.25$.

x_i^p	$x_{2i}^g x_{2i+1}^g$ (with $k_r = 2$)		
	extended TV		original TV
	$r = 1$	$r = 3$	$r = 2$
0	00, 01, 10	00	00, 01
1	11	01, 10, 11	10, 11

Table 3.2. The trivial voting mapping for $k_r = 2$

When using the binary TV mapping for the one-max problem each bit of a phenotype $x^p \in \Phi_p$ is represented by k_r bits of the genotype $x^g \in \Phi_g$. The string length of a genotype x^g is $l_g = k_r \times l_p$ and the size of the genotypic search space is $|\Phi_g| = 2^{k_r l_p}$. Table 3.2 illustrates for $k_r = 2$ the two possibilities ($r = 1$

and $r = 3$) of assigning genotypic BBs $\{00, 01, 10, 11\}$ to one of the phenotypic BBs $\{0, 1\}$ when using the extended TV mapping described in the previous paragraphs. With denoting x_i^p the value of the ith bit in the phenotype, the $2i$th and $(2i+1)$th bit of a genotype determine x_i^p. Because the size of the BBs $k = 1$, the number of genotypic BBs that represent the optimal phenotypic BB is either $r = 1$ or $r = 3$ (compare (3.12)). Furthermore, Table 3.2 also lists the case where $r = 2$. This case is the original uniformly redundant TV mapping. The second bit of each genotypic BB does not contribute to the construction of the phenotype.



(a) $k_r = 2$ (b) $k_r = 3$

Figure 3.4. Experimental and theoretical results of the proportion of correct BBs on a 150-bit one-max problem using the trivial voting mapping for $k_r = 2$ (left) and $k_r = 3$ (right). The lines without line points show the theoretical predictions. When using non-uniformly redundant representations, GA performance is changed with respect to the overrepresentation or underrepresentation of the high-quality BBs.

		extended TV mapping			original TV
		$u = 1$	$u = 2$	$u = 3$	mapping
$k_r = 2$	r	3	1	-	2
	x_0/N	3/4	1/4	-	$2/4 = 1/2$
$k_r = 3$	r	7	4	1	4
	x_0/N	7/8	$4/8 = 1/2$	1/8	$2/4 = 1/2$

Table 3.3. Properties of the different TV mappings for the one-max problem ($k = 1$)

In Fig. 3.4(a) ($k_r = 2$) and Fig. 3.4(b) ($k_r = 3$), the proportion of correct BBs at the end of a run for a 150 bit one-max problem using the TV mapping is shown. For this problem 2^{150} different phenotypes are represented by either

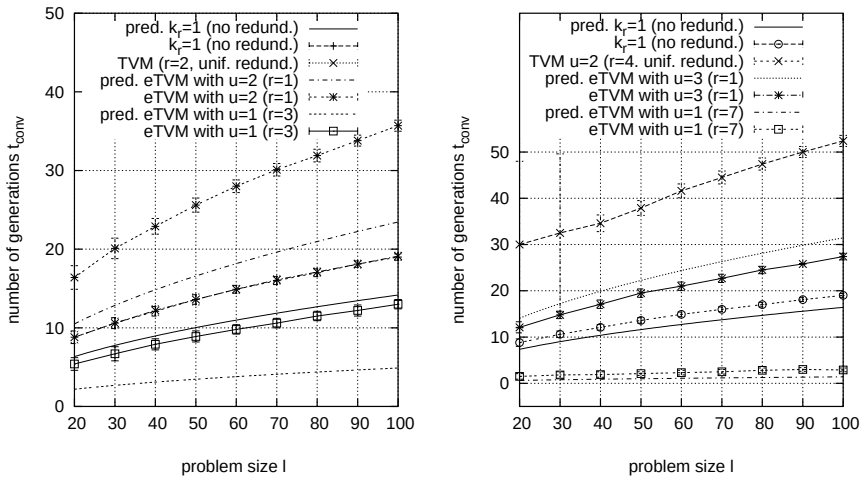
2^{300} ($k_r = 2$) or 2^{450} ($k_r = 3$) different genotypes. If we use the eTV mapping (indicated in the plots as eTVM) we can set u either to 1 or 2 ($k_r = 2$) or to 1, 2, or 3 ($k_r = 3$). The corresponding values for r, which can be calculated according to (3.12), as well as x_0/N are shown in Table 3.3. x_0 is the expected number of copies of the best BB in the initial population and N is the population size. Furthermore, the figures show the results when using the original, uniformly redundant TV mapping, and when using the non-redundant representation with $k_r = 1$. The lines without line points show the theoretical predictions from (3.7), and the lines with line points show the empirical results which are averaged over 250 runs. The error bars indicate the standard deviation.

The results show that for the uniformly redundant TV mapping, $r = 2$ ($k_r = 2$) or $r = 4$ ($k_r = 3$), we get the same performance as for using the non-redundant representation ($k_r = 1$). As in the original model proposed by Harik et al. (1997) the theoretical model slightly underestimates GA performance. As predicted by our model which we proposed in Sect. 3.1.4, GA performance does not change when using a uniformly redundant representation. Furthermore, we can see that if the optimal BB is underrepresented ($u = 2$ for $k_r = 2$ and $u = 3$ for $k_r = 3$) GA performance decreases. Equation 3.7 gives us a good prediction for the expected solution quality if we consider that the non-uniform redundancy of the representation changes the initial BB supply according to (3.6). If the optimal solution is overrepresented ($u = 1$ for both cases, $k_r = 2$ and $k_r = 3$) GA performance increases. Again the theoretical models give a good prediction for the expected proportion of correct BBs.

Summarizing the results, we can see that using the uniformly redundant TV mapping does not change GA performance as compared to using the non-redundant representation. Only if we overrepresent the optimal phenotypic BB, does GA performance increase; likewise, if we underrepresent the optimal BB, GA performance drops. As our derived model is able to make accurate predictions for the expected solution quality, our assumption that synonymously redundant representations influence GA performance by changing the initial supply seems to be valid.

In the remaining paragraphs, we perform an empirical investigation into the effect of the TV mapping on the number of generations until the population of a selectorecombinative GA converges. Again we use the one-max problem and the TV mapping from above with the same parameters except the population size is set to $N = 2l_p$ to allow reliable decision making for the one-max problem (Goldberg et al. 1992). As we use tournament selection without replacement of size two the selection intensity $I = 1/\sqrt{\pi}$.

Figures 3.5(a) ($k_r = 2$) and 3.5(b) ($k_r = 3$) show the number of generations that are necessary until 90% of all phenotypic BBs are found over the problem size which is equal to $l = l_p$. The lines without line points show the predictions from (3.10) and the lines with line points plot the empirical results. We can see that the run duration of a GA when using the non-redundant representa-



(a) $k_r = 2$ (b) $k_r = 3$

Figure 3.5. Theoretical predictions and experimental results for the number of generations that are necessary until 90% of all phenotypic BBs are correctly identified. The plots are for one-max problems and trivial voting mapping with $k_r = 2$ (left) and $k_r = 3$ (right).

tion ($k_r = 1$) is exactly the same as when using the uniformly redundant TV mapping with $k_r = 2$. For $k_r = 3$ and $u = 2$ (uniform redundancy) the run duration is slightly increased in comparison to the non-redundant encoding. We expect that this difference increases with larger k_r. In agreement with the results from Thierens (1995), we report a small underestimation of the expected number of generations when using either non-redundant, or uniformly redundant, representations.

When using non-uniformly redundant variants of the eTV mapping the underestimation is larger, but nevertheless the model gives a good approximation for the expected number of generations. We can see that increasing r increases the run duration. For example, if each phenotypic bit is represented by three genotypic bits ($k_r = 3$) and a one is represented if at least one out of three genotypic bits is set to one ($u = 1$) then a GA finds the good solutions after very short time (compare eTVM with $u = 1$). The expected number of generations shows the predicted behavior. The necessary number of generations increases by about $O(\sqrt{l})$. We see that the proposed model allows us to make good predictions for the expected run duration.

Concatenated Deceptive Trap Problem

Our second test example uses deceptive trap functions. Traps were first used by Ackley (1987) and investigations into the deceptive character of these func-

tions were provided by Deb and Goldberg (1993). Figure 3.6 depicts a 3-bit deceptive trap problem where the size of a BB is $k = 3$. The fitness value of a phenotype x^p depends on the number of ones u in the string of length l. The best BB is a string of l ones which has fitness l. Standard GEAs are misled to the deceptive attractor which has fitness $l - 1$. For the 3-bit deceptive trap the signal difference d is 1, and the fitness variance equals $\sigma^2_{BB} = 0.75$. We construct a test problem for our investigation by concatenating $m = 10$ of the 3-bit traps so we get a 30-bit problem. The fitness of an individual x is calculated as $f(x) = \sum_{i=0}^{m-1} f_i(u)$, where $f_i(u)$ is the fitness of the ith 3-bit trap function from Fig. 3.6. Although this function is difficult for GEAs it can be solved with proper population size N.

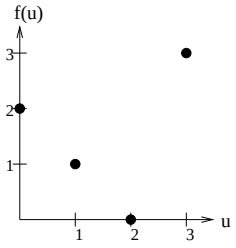


Figure 3.6. A 3-bit deceptive trap problem

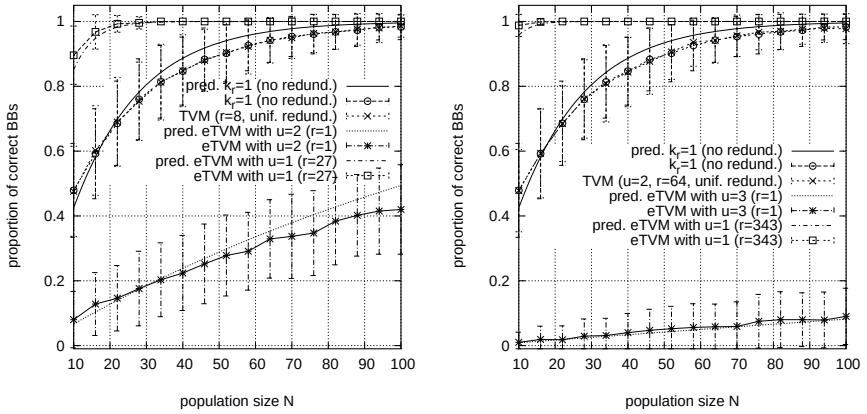
For deceptive traps of size $k = 3$ we can calculate the number r of genotypic BBs that represent the optimal phenotypic BB according to (3.12). Table 3.4 summarizes, for the binary TV mapping, how r and x_0/N depends on u, which describes how many of the genotypic alleles must be set to 1 to encode a 1 in the phenotype. x_0 is the expected number of copies of the best BB in the initial population and N is the population size. We have also included the properties of the original uniformly redundant TV mapping.

		extended TV mapping			original TV
		$u = 1$	$u = 2$	$u = 3$	mapping
k_r=2	r	$3^3 = 27$	$1^3 = 1$	-	$2^3 = 8$
	x_0/N	27/64	1/64	-	$8/64 = 1/8$
k_r=3	r	$7^3 = 343$	$4^3 = 64$	$1^3 = 1$	$4^3 = 64$
	x_0/N	343/512	$64/512 = 1/8$	1/512	$64/512 = 1/8$

Table 3.4. Properties of the different TV mappings for the trap of size $k = 3$

By analogy to the previous paragraphs, Figs. 3.7(a) ($k_r = 2$) and 3.7(b) ($k_r = 3$) show the proportion of correct BBs at the end of a run over different population sizes for ten concatenated 3-bit deceptive trap problems. In this problem, 2^{30} different phenotypes are represented by either 2^{60} ($k_r = 2$) or 2^{90} ($k_r = 3$) different genotypes. As before, we use tournament selection without replacement of size 2. In contrast to the one-max problem, two-point crossover was chosen for recombination. Uniform crossover would result in an improper

mixing of the BBs because the genotypic BBs are either of length $l_g = k_r l_p = 6$ ($k_r = 2$), or of length $l_g = 9$ ($k_r = 3$). Again, the lines without line points show the predictions of the proposed model for different r. Furthermore, empirical results, which are averaged over 250 runs, are shown for various values of r. The results show that for the uniformly redundant TV mapping we get the same performance as when using the non-redundant representation ($k_r = 1$). As in the experiments for the one-max problem the proposed model predicts the experimental results well if the eTV mapping is used and some BBs are underrepresented or overrepresented.



(a) $k_r = 2$ (b) $k_r = 3$

Figure 3.7. Experimental and theoretical results of the proportion of correct BBs for ten concatenated 3-bit deceptive traps. We show results for different variants of the TV mapping and $k_r = 2$ (left) and $k_r = 3$ (right). The lines without line points show the theoretical predictions. As predicted, GA performance sharply decreases if the eTV mapping underrepresents the optimal BB.

The presented results show that the effects of synonymously redundant representations like the TV mapping on the performance of GEAs can be explained well by a change of the initial supply of high-quality BBs. If the eTV mapping favors high-quality BBs then the performance of GAs is increased. If good BBs are underrepresented the performance is reduced. If the representation is uniformly redundant, GAs show the same performance as when using the non-redundant encoding.

3.1.7 Conclusions and Further Research

This section investigated how redundant representations influence the performance of GEAs. It distinguished between synonymously and non-synonymous-

ly redundant representations and illustrated that non-synonymous redundancy does not allow genetic operators to work properly and therefore reduces the efficiency of evolutionary search. When using synonymously redundant representations, GEA performance depends on the change of the initial supply. Based on this observation, models were developed that give the necessary population size for solving a problem, and the number of generations as $O(2^{k_r}/r)$, where k_r is the order of redundancy and r is the number of genotypic BBs that represent the optimal phenotypic BB. As a result, uniformly redundant representations do not change the behavior of GAs. Only by increasing r, which means overrepresenting the optimal solution, does GA performance increase. In contrast, GA performance decreases if the optimal solution is underrepresented. Therefore, non-uniformly redundant representations can only be used advantageously if information exists a-priori regarding the optimal solution. The validity of the proposed theoretical concepts is illustrated for different variants of the redundant trivial voting mapping. The results show that the developed population sizing and time to convergence models allow an accurate prediction of the expected solution quality and solution time.

The proposed classification, population sizing, and time to convergence models allow us to evaluate redundant representations in a systematic and theory-guided matter. This approach will help users and researchers to answer some of the disputed questions regarding the benefits of redundant representations and to use redundant representations such that they increase the performance, reliability and efficiency of evolutionary computation methods.

In this study, we only considered crossover-based search and neglected the influence of redundant representations on mutation-based search. However, we believe that many of the discussed topics are also relevant when using mutation. According to Sect. 3.1.2, we believe that in analogy to the results from Knowles and Watson (2002), using non-synonymously redundant representations reduces the performance of mutation-based search. As these representations have low locality, mutation will not work properly and the search becomes random. Furthermore, there is some theoretical evidence (Radcliffe 1991a; Whitley et al. 1997; Rana and Whitley 1997; Whitley 1999; Whitley 2000a; Christensen and Oppacher 2001) that mutation-based search only performs well if the connectivity of the phenotypic search space is preserved by the used representation. If the connectivity is either not preserved, such as for low locality representations, or greatly increased (which results in a reduction of the relevant connectivity) like in many non-synonymously redundant representations, the performance of mutation-based search decreases. In contrast, we expect when using synonymously redundant representations that mutation-based search will show similar behavior and performance as when using crossover-based search. Using synonymously redundant representations introduces many plateaus in the fitness landscape but does not change the structure of the search space. Mutation can still easily find neighboring phenotypes. When using non-uniformly redundant representations, some plateaus in the fitness landscape have a larger size which increases the probability

that mutation finds the solution represented by the genotypes forming this plateau. As a result, the performance of mutation-based search increases if a synonymously redundant representation overrepresents the optimal solution, and decreases otherwise.

3.2 Scaling

This section provides the second of three elements of a theory of representations and addresses representations which change the importance of alleles when mapping genotypes on phenotypes. Common representations for GEAs often encode the phenotypes by using a sequence of alleles. When assigning phenotypes to genotypes, a representation can change the importance of the alleles. For example, a phenotype is a list of integers and all alleles (integers) are equally relevant for calculating the fitness of a phenotype. We know from previous work that the BBs (alleles) are solved in parallel if all alleles are equally relevant (Goldberg 1989c). The situation that all alleles are equally relevant is equivalent to the situation that the BBs are uniformly scaled. However, when encoding the phenotypic integers using binary strings, the contributions of the genotypic bits to the fitness function are no longer equal and some bits are more relevant than others. When using such non-uniformly scaled representations, the BBs are solved sequentially and domino convergence occurs. Therefore, the time to convergence increases and the genetic search is affected by genetic drift. This means that lower salient BBs are fixed before they can be reached by the search process.

Based on previous work (Rudnick 1992; Thierens 1995; Thierens et al. 1998; Harik et al. 1997), we describe how the performance of GEAs is influenced by the use of representations with non-uniformly scaled BBs. We develop a population-sizing model with, and without, considering genetic drift. The theoretical models are verified with empirical results.

In the following subsection, we review the effects of domino convergence and genetic drift. In Sects. 3.2.2 and 3.2.3 we develop population sizing models for domino convergence with, and without, considering drift. We present empirical verification of the proposed models in Sect. 3.2.4 and end with concluding remarks.

3.2.1 Definitions and Background

Representations assign phenotypes x^p consisting of different alleles x_i^p, where $i \in \{0, \ldots, l_p\}$, to genotypes, which also consist of different alleles x_i^g, where $i \in \{0, \ldots, l_g\}$. Furthermore, the function f_p assigns to each phenotype x^p a corresponding fitness value. Usually, the fitness of a phenotype depends on the values of x_i^p. Therefore, different phenotypic alleles can have a different contribution to the fitness of an individual x^p. Representations are defined to be *uniformly scaled* if the genotype-phenotype mapping f_g does not change