Adaptive Scaling of Evolvable Systems

Simon P. Hammond

A thesis submitted

to The University of Birmingham

for the degree of Doctor of Philosophy

School of Computer Science

The University of Birmingham

Edgbaston

Birmingham

B15 2TT

United Kingdom

November 21, 2007

Abstract

Neo-Darwinian evolution is an established natural inspiration for computational optimisation with a diverse range of forms. A particular feature of models such as Genetic Algorithms (GA) [18, 12] is the incremental combination of partial solutions distributed within a population of solutions. This mechanism in principle allows certain problems to be solved which would not be amenable to a simple local search. Such problems require these partial solutions, generally known as building-blocks, to be handled without disruption. The traditional means for this is a combination of a suitable chromosome ordering with a sympathetic recombination operator. More advanced algorithms attempt to adapt to accommodate these dependencies during the search.

The recent approach of Estimation of Distribution Algorithms (EDA) aims to directly infer a probabilistic model of a promising population distribution from a sample of fitter solutions [23]. This model is then sampled to generate a new solution set.

A symbiotic view of evolution is behind the recent development of the Compositional Search Evolutionary Algorithms (CSEA) [49, 19, 8] which build up an incremental model of variable dependencies conditional on a series of tests. Building-blocks are retained as explicit genetic structures and conditionally joined to form higher-order structures. These have been shown to be effective on special classes of hierarchical problems but are unproven on less tightly-structured problems.

We propose that there exists a simple yet powerful combination of the above approaches: the persistent, adapting dependency model of a compositional pool with the expressive and compact variable weighting of probabilistic models. We review

and deconstruct some of the key methods above for the purpose of determining their individual drawbacks and their common principles. By this reasoned approach we aim to arrive at a unifying framework that can adaptively scale to span a range of problem structure classes.

This is implemented in a novel algorithm called the Transitional Evolutionary Algorithm (TEA). This is empirically validated in an incremental manner, verifying the various facets of the TEA and comparing it with related algorithms for an increasingly structured series of benchmark problems. This prompts some refinements to result in a simple and general algorithm that is nevertheless competitive with state-of-the-art methods.

Acknowledgments

If you have built castles in the air,
your work need not be lost;
that is where they should be.
Now put the foundations under them.
— Henry David Thoreau

This section is the easiest to write in the thesis since it is addressed to just a few people.

Professor Xin Yao worked to keep me focused and in return I aimed to keep him stimulated. His readiness to ask difficult questions made this thesis far sharper than it otherwise would have been. My thesis group of Prof. Yao, Dr Ela Claridge and Dr Peter Tiňo took the time to both motivate and challenge me with regular grillings and bastings. Their perspectives were invaluable in balancing the themes of this thesis.

I am indebted to the school of computer science for facilities, funding and a fascinating environment. One of the main pleasures of academic life is the rare quality of the characters that abound. Different people have seasoned the experience at different times but some deserve special attention. Chris Bowers was always ready to demonstrate an eager capacity to debate any point. Vineet Khare was an ideal sounding-board in front of a white-board, quick to understand and add clarity. Jorge Cham never ceased startling me with the accuracy of his observations. Others, in their individuals ways, positively influenced me (in alphabetical order): Gavin Brown, Guo Chen, Dave Gurnell, Zhenyu Liu, Mark Roberts, Max Salazar-Lechuga, Hang Zheng and countless others.

The weekly workshops and regular socials of the Purple Mermaid crowd gave me a whole different circle of friends, a complete change of scenery and an excuse to throw

things at people.

I have always taken for granted the unquestioning, and largely uncomprehending, support of my parents and siblings. My *fereshteh*, Nazli, has been a constant source of joy, always able to raise a smile and keeping me from becoming a robot.

Finally, a small note of thanks to JSP of Oxfordshire whose 'Big Blue' ear defenders removed the hum of the lab so I could listen to the noise in my head.

Contents

List of Figures

List of Tables

Learn each small people's genius, policies,
The ant's republic, and the realm of bees;
How those in common all their wealth bestow,
And anarchy without confusion know;
And these for ever, though a monarch reign,
Their separate cells and properties maintain.
Mark what unvaried laws preserve each state,
Laws wise as Nature, and as fix'd as fate.

— Alexander Pope, Essay on Man.

Chapter 1

Introduction and Background

1.1 Philosophical Preamble

E conchis omnia (everything from shells) was the motto that Erasmus Darwin added to the family crest over 80 years before the publication of *Origin of Species* by his grandson [3]. It reflected his original conviction that the incredible complexity of living systems was rooted in simpler origins. The mechanism for this, based on selection over heritable variation, was left for Charles Darwin to describe. The classical characterisation of Darwinian adaptation is of gradual change under natural selection. Fossil records show species adapting over time, shaped by the forces in their particular environment. It is a generally accepted process; yet it may not be the whole story.

A current biological school of thought suggests a further mechanism for increasing adaptive complexity. This has previously independently replicating entities develop a long-term and intractable association to protect and exploit a mutually-beneficial interaction. The so-called *major transitions* in evolution, it is proposed, are the result of pre-adapted entities coming together rather that a straightforward mutation [29].

This phenomenon is compelling in its own right but also may have implications for key areas where computational abstractions of evolutionary dynamics are charged with finding solutions to complex problems which resist analytical solution. Natural evolution has demonstrated a capacity to produce systems that continue to humble our state-of-the-art models in terms of their adaptive complexity. Biological organisms and ecosystems, as well as social, cultural and technological networks, exhibit a nuanced and creative complexity that emerged without design. Even cognition may have an evolutionary underpinning [9]. We would like a handle on how these complex systems emerge so that we might wield the process for our own ends.

1.2 Adopting Evolution for Search

Search is a fundamental computational task with an accumulated arsenal of techniques. The natural process of evolution inevitably inspired more techniques to add to this list. The generic term of *Evolutionary Algorithm* (EA) is applied to all these.

Whereas many search algorithms are tailored to different domains with their own particular characteristics EAs are widely touted as good, robust, all-round search algorithms which are easily adapted to representing potential solutions for different domains. Since it is generally rather easy to implement an EA (even without much prior knowledge of the problem domain) they have been applied to very many diverse domains. These include simple artificial life forms, game strategies — in fact wherever there is a representation and a means to vary them then all that is required is a method for differentiating them.

Naturally, some applications are more successful than others. These tend to be ones where knowledge of the domain has been integrated into the algorithm. Therefore, the success of any particular application still relies largely on the skill of the

designer rather than being a straightforward application of engineering principles.

Such tweaks may reduce the search space or improve the quality of variants of fit individuals. These are instances of the 'stone soup' effect[1], where the benefits of an approach lie more in its additional customisation rather than its intrinsic qualities.

In inexperienced hands, different EAs are applied to different problems often with scant justification. Novel representations are often processed by operators which have been adopted without a clear understanding of their intrinsic biases.

Whilst it can be impractical to do a formal analysis of algorithms beyond a certain level of complexity, this does not exclude a principled approach to developing an evolutionary algorithm. This is the ambition for this thesis which aims to produce a framework with minimal arbitrariness and maximal justification.

1.3 Adaptive Complexity in Nature and EAs

One capacity that turns out to be central is the ability to adapt to complex networks of interactions between the elements in the search [25]. In the biological domain, these entities were originally regarded as individual organisms, competing, preying and evading others but genes have also been considered as interacting entities [4] and, via evolutionary transitions [30], entities scaling entirely from replicating molecules to the most complex societies have been unified under a common model.

This prompts us to define *complexity* for the purposes of this thesis. In terms of the problem (or environment) it refers to the number and structure of interactions between a set of entities. A scale of complexity is given in section 1.4. Conversely, in terms of solutions, we use complexity in referring to the number and structure of genetic linkages in the gene pool. Adaptive complexity is the result of a correspondence

[1]Named by the author after the traditional folk story in which soup is promised from a magic stone but is actually the product of additional, *ad hoc* ingredients.

between the structures inherent in the problem and those produced in the search.

The term *evolutionary transition* from biology is applied to the process whereby previously independent entities develop a mutually-beneficial interaction which then becomes maintained by their binding into a higher-level entity, resulting in increasing complexity [30]. Examples range from the formation of chromosomes, through the emergence of multi-cellular organisms and right up to the appearance of societies and language. This process has been started to really be considered from the perspective of Evolutionary Computation only very recently [24, 39, 8] and still tends to be considered within a biological context rather than a purely optimisational one.

An original mechanism similar in spirit to the evolutionary transition was proposed in the *building-block hypothesis* which proposes that high-quality solutions may be produced by repeatedly combining highly-fit partial solutions of an increasing order [12]. This lies at the heart of standard EA theory yet the current method for carrying this out is burdened with issues, described in section 2.1.2.

Given these apparent parallels between a theory for the emergence of biological complexity and a requirement from optimisation, a further exploration into evolutionary transitions models of optimisation seems promising. Before this, we need to be somewhat clearer about the form for the problems which our prospective method will address.

1.4 Definition of Problem Classes

The general class of problem addressed herein is of discrete combinatorial optimisation. Simply put, we have a repeatable outcome from a fixed set of choices and we are trying to use this outcome to find the best decisions to make for our choices in future. Moreover, we are limited in our number of tries and aim for as few as possible.

Each choice is represented by a *variable* which may take a *value* from any discrete, finite set, e.g. an integer range, symbol, bit, logical operator or colour. Moreover, each variable may be drawn from a different set. For the purposes of this thesis we confine ourselves to a conventional binary representation.

The term we will borrow from biology as a shorthand for a specific value assigned to a particular variable is *gene*. We presume that the range of possible solutions, the *search space*, resulting from the combinatorial nature of the problem makes an exhaustive search unfeasible or undesirable.

We also note the following strict limitations on our initial knowledge of the problem with the expectation of improvement after a number of evaluations. The aptness of any particular search algorithm lies in how successfully it departs from this initial state.

Value ignorance Any value is as likely to be optimal as any other for any particular variable. There is no initial bias for one over another. Hence we generally start with uniformally random solutions.

Neighbourhood ignorance Whilst some domains (such as continuous numerical ones) tend to imply a bias in substituting one value over another in a variable, we assume no such knowledge[2]. Our variables 'mutate' to values irrespective of their current value.

Dependency ignorance We have no prior knowledge of which variables interact with each other in determining the fitness. More specifically, we have no knowledge of whether the optimality of a particular value is dependent on the presence of a par-

[2]For a purely binary representation, as we will be using, there is actually no scope for such knowledge.

ticular value for another variable. The setting of a variable is therefore irrespective of the settings for other variables in a given solution.

These conditions are invoked later when considering the advantages of relevant algorithms to ensure a hard but fair comparison. For example, dependency ignorance effectively rules out the deliberate ordering of variables in the GA's chromosome. For simple EDAs there is no mechanism for tackling dependency ignorance at all.

The feedback is limited to a single, quantitative value measuring the quality of the solution which, according to convention and in line with the evolutionary metaphor, we call *fitness* and seek to maximise.

Formally, the fitness feedback f is a function of the set of decisions made for the mandatory choices. Given that the significance and character of each choice is unknown, we may for notational convenience index them for representation as a set of variables, C, such that:

$$f(C) = f(c_1, c_2, ..., c_l) \tag{1.1}$$

Where there are l decisions to be made as we seek to maximise $f(C)$. It is also important to stress that, given dependency ignorance, there is no significance to the ordering of the variables. This total absence of prior problem knowledge qualifies this definition for the class of 'black box' optimisation. We have no prior information about the relative quality of the values for c_i, any relation between the options for c_i *or* between c_i and c_j (where $i \neq j$).

1.4.1 Value Substitution

Value ignorance asserts the lack of any prior knowledge of the relative quality of the range of values for a variable. Clearly this must change as evaluations accumulate.

Otherwise, each successive solution will be as likely as the first to be optimal.

Initially, when switching a choice, we have no bias toward any replacement (as per the neighbourhood ignorance rule). As the search progresses, we have the opportunity to adapt the value of any particular choice in light of fitness feedback and favour some values for a variable over others, either probabilistically or systematically. This notion is explored further in section 1.5.

1.4.2 Variable Dependencies

We now consider what sort of relation may exist between variables in a problem. For the simplest problems, there is no interaction between the variables; an optimal value is optimal regardless of the other variable settings. In such cases, each variable can be optimised independently by fixing the values of the other variables and testing all values for optimality. Since the global optima can be found by sequentially optimising each variable, these problems can be easily solved in linear time via straightforward enumeration.

Problems which are both more interesting and more relevant to real world applications tend to have unforeseen interactions between the variables, i.e. choices have 'knock-on' effects. These interactions are often termed *epistatic dependencies*. If any dependency within a problem is limited to k variables then that problem is termed *k-bounded* [19]. Clearly, the difficulty of a problem increases with k. In the extreme cases, with k as 1 we have the *univariate* problem as before. Where k is equal to the number of variables, we would require a complete enumeration of the search space to guarantee finding a global optima. We refer to problems where $1 < k < l$ as *multivariate* or *modular* problems.

Hierarchical problems were originally identified by Simon [42] and brought into

EC via the Royal Road R2 function [32] and, in a stronger form by the H-IFF [50]. The Hierarchical Problem Generator [6] provides instances of this class. A problem function has a *hierarchical* structure if dependencies exist between interdependent sets. Finally, where a value (or set of values) may be incorporated in several 'higher-level' sets, we adopt the term *polyhierarchical*.

Several of the following classes of dependency structures have been discussed elsewhere, e.g. by Pelikan [35]. We identify each one next, with reference to figure 1.1 for illustration.

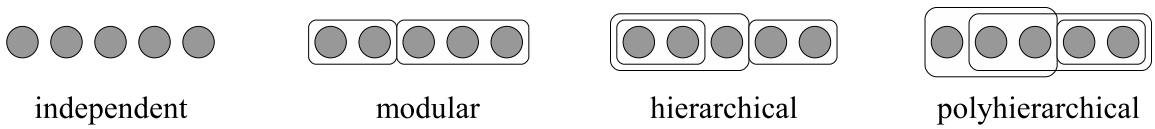

Figure 1.1: Classes of dependency structure. Adjacency of interacting variables is for ease of visualisation. Variables are indexed from 0 to 4, left to right.

Linear (or univariate) problems involve no fitness interaction between variables, e.g onemax. The fitness contribution, and therefore the optimal fitness, for any value is constant, regardless of the presence of any other values.

$$f(x) = f^0(x_0) + f^1(x_1) + f^2(x_2) + f^3(x_3) + f^4(x_4)$$

Modular problems contain variables which can only be evaluated as part of a set of order up to k. It is said to have k-bounded dependencies [5]. The linear problem may then be viewed as a modular problem where the modules are of order 1.

$$f(x) = f^1(x_0, x_1) + f^2(x_2, x_3, x_4)$$

Hierarchical problems contain modules that contain modules. They can have an order up to k components where each component may be a variable or set of variables.

$$f(x) = f^2(f^1(x_0, x_1), x_2) + f^3(x_3, x_4)$$

Polyhierarchical problems allow a variable or module to be part of multiple, alternative hierarchies. A well-known example is the Travelling Salesman Problem (TSP) where a city can be included in several alternative sub-routes.

$$f(x) = f^1(x_0, x_1, x_2) + f^3(x_1, x_2, f^2(x_3, x_4))$$

The structure of these dependencies determine the character and ultimately the difficulty of the problem. The problems above are listed in increasing difficulty which naturally follows from their increasing generality. By the dependency ignorance rule, we have no prior knowledge of these dependencies. Therefore they must be inferred in some way from the evaluation of sampled solutions.

1.5 EA Motivations and Methods

We now give a high-level overview of the evolution-inspired approach to optimisation and define some of the terms we will be using.

Heuristic techniques are applicable where a complete search of all possibilities is out of the question. Either the space to be searched is unfeasibly large or the cost per evaluation means we must be willing to accept a less than perfect solution. In any case, the result is the same: we must restrict ourselves to considering only a subset of the possible solutions. In the absence of problem knowledge, the selection of this

subset is arbitrary. We have no information about where to start or where to go after that. It is essentially an exhaustive search prematurely terminated. The only advantage over a random search is in ensuring that no solution will be considered more than once. For some problems (so-called 'needle-in-the-haystack' problems) this is the best strategy that we can apply. For other problems, where we believe that similar solutions are of a similar quality, we can have higher expectations. We can generate variants of our best solutions and replace them if we find variants of a higher quality. This process resonates strongly with the classical Darwinian theory of adaptation by natural selection and the analogy is recognised in the search algorithms thus inspired.

The group of heuristic approaches under the current umbrella term of Evolutionary Computation has myriad roots. In adopting the evolutionary metaphor from nature, we use (and sometimes abuse) various biological terms. We use the term *individual* to refer to a candidate solution and the term *population* for a group of these, possibly containing duplicates.

EC approaches can be divided into individual or population-based algorithms as detailed below. We consider population-based algorithms to be not just about concurrent lineages but also involving the genetic interaction of individuals.

1.5.1 Individual-based Algorithms

Evolution Strategies (ES) [40, 41] are based on maintaining a best individual and replacing it with improved or equally fit variants of itself. It generates variants by perturbing the values in an operation termed *mutation*. The search is therefore a stochastic one with the mutation operator generating variants that have shared genes with their parents. As such, it assumes a limited order of dependency between vari-

ables such that small numbers of variables can be modified to achieve a fitness gain and so ES are typically applied to real-valued domains where a degree of relatedness can be assumed between values and the neighbourhood rule is mitigated.

The nature of variation requires some consideration here. Variants of an individual must have some commonality with the source, the degree of which determines balance between exploration and exploitation in the search. At one end of this range we have random search, independent of past individuals, and at the other we have individuals which are indistinguishable from their origins. The goal is to find an appropriate degree of commonality (or even learn the common features for improved search).

A *lineage* is the historical trajectory of preceding variants of an individual. A proposed benefit of maintaining a population is that multiple promising lineages may be pursued concurrently and only terminated as they fail to compete with better performing lineages. In this way, we save effort by having a heuristic for terminating some lineages to expand on more promising ones. This effect is achieved independently of any direct individual interaction such as recombination used in population-based algorithms, discussed next. This means there can be an advantage for populations in individual-based algorithms (which are still individual-based algorithms).

The source of the variants, the current population of best individuals, is used in a way that distinguishes it from a systematically ordered search.

The *search state* for a population-based algorithm is the population, whereas in an ordered search it is the current best individual and the last individual to be evaluated. Any search can be terminated and continued at any time by restoring its respective search state. Together with the *search operators* which determine which individuals are going to be (or are likely to be) considered next, this comprises the *search model*. If this is a stochastic process, it results in a probabilistic distribution over the search space for future individuals which we will refer to as the *search distribution*.

1.5.2 Population-Based Algorithms

Population-based search extends the search state to embody a set of interacting individuals. This offers several key advantages over the single-individual approach, which stem from the additional high-level information that can be obtained from the adapting population.

As noted above, for a run in which multiple individuals are being adapted concurrently, the fitness of any particular individual can be compared to its peers. Poorly performing individuals can be removed via selection and replaced with variants of more promising individuals. This can be viewed as terminating some lineages and forking others. It gives a straight gain over a series of individual-based run in terms of resources.

A more interesting possibility opened up by the maintenance of concurrent lineages is for the transmission of values *between* individuals. This is a chief characteristic of the *Genetic Algorithm* (GA) [18, 12] and many of its extensions. The canonical form maintains a population of a fixed number of individuals and employs a recombination (or crossover) operator to allow the transmission of structures between individuals. They also employ mutation to maintain 'diversity' in the population.

The motivation behind the *recombination* operator mentioned earlier is to combine parallel lineages, bringing together useful elements of parents into a single individual. This makes the search distribution a function of the entire population rather than individuals. Indeed, new individuals may have as few as half their values in common with any current individuals. The rationale behind this is rooted in Holland's *building-block hypothesis* where a building block is a set of genes implicitly representing a sub-solution. The hypothesis suggests that building-blocks of increasing order can be combined to obtain an optimal solution. Recombination acts on an ordered

representation of the variables; and so for the original 1-point (and later 2-point) crossover operators a crossover point — and therefore disruption — is less likely to occur within sets of values that are closer in the ordering. As such, experienced GA practitioners take care over the ordering of the variables on the chromosome to maximise proximity between variables believed to interact under some fitness function.

This mix of recombination and representation is the means by which designers may import prior knowledge regarding dependencies into the search. This 'tight encoding' between variables is part of the *genetic linkage* implicit in the GA. It also contravenes our dependency ignorance condition and is a clear target for an improved algorithm. We return to the central issue of supporting genetic linkage in due course.

The *schema theorem* [12] was presented in order to define the conditions under which the building-block hypothesis is expected to operate. A schema in this case is defined as a set of values for a subset of all the variables that constitute an individual. The original conception of a building-block was of a set of values which were correlated with a significant fitness contribution. For the original GA, these were additionally required to have a short defining length to avoid disruption under positional recombination operators. One of the properties of the model to be presented herein is a lifting of that problematic requirement.

The population property of *convergence* is used widely. It could mean adaptation toward an optima *or* increasing homogeneity. The general understanding does not distinguish except in the particular case of 'premature convergence' where the latter occurs without the former. To avoid such conflation, we will only take the latter use. We will use the term *adaptation* for the process giving incrementally better solutions. The term *diversity* is seen as antagonistic to convergence. Either of these is the expected outcome of the competing forces of selection and variation. Selection tends to homogenise the population and variation, usually in the form of a mutation oper-

ator, is used to prevent total homogeneity into a sub-optimal solution. Maintaining a productive balance of these is a non-trivial challenge for the algorithm designer, particularly as the balance may change over time. Various population maintenance strategies have been proposed in order to retain rich diversity in the face of strong selective pressures. These range from *elitism*-based methods, where a fixed proportion of individuals are retained, to niching methods such as *deterministic crowding* where offspring may only replace their closest parent [28].

The ultimate aim is neither convergence nor diversity but the generation of an optimal solution with the least expenditure. We argue this is an unnecessary conflict once we deepen our understanding of the underlying processes of selection and mutation. The positive feedback loop of multiple copies of individuals, building-blocks and genes is the cause and effect of convergence. Deconstructing the interplay between these is key to the development of an appropriately adaptive search strategy.

1.5.3 Probabilistic Models

A relatively recent technique within evolutionary optimisation replaces the genetic variation operators with an explicit probabilistic distribution for existing individuals which is then sampled to generate new individuals. Such algorithms are referred to as *Estimation of Distribution Algorithms (EDAs)* [23].

The model may be derived from a population or even replace it entirely. If the model is supplementary to the population then its derivation occurs alternately with the sampling of a new population. The extraction of a search distribution from the population necessarily entails a selective phase as we use fitness values to either select an elite subset of the population to model, or weight a mapping of the population distribution to a search distribution.

The probabilistic model may be a simple one that is adapted toward new, fitter individuals [1, 16]. More commonly, it is the result of a systematic search to find a distribution that approximates the population to a required degree. These probabilistic model-building approaches iteratively search for distributions that match a population, select within it and then sample from that selection.

The search can be computationally expensive, ever more so as we increase the complexity range of considered models. It also bears the problem of determining an appropriate degree of fitting. A model that is too specific will not vary enough from current population members. On the other hand, a model that is too general will fail to capture important dependencies suggested by the population. Moreover, the search for a model is likely to have to restart as the population changes significantly. In almost all cases the model and (the large part of) the population are alternatively discarded at each iteration[3]. This means wasted effort as the search for a model begins anew each time and potentially useful individuals are lost from the population.

1.5.4 Comparing Population-Based Algorithms and EDAs

Both these approaches rely on a series of individual evaluations in order to progress their search model. They both employ a fitness-based bias in order to focus on some individuals above others, i.e. selection. The difference is that the population-based approach takes the population itself as a model of a good search distribution and processes its implicit statistics via the genetic operators. The EDA approach, on the hand, takes the population distribution as a data set to be separately modelled.

The weakness of the population-based approach is that the genetic representation and operators need to be 'tuned' to ensure adaptability and this is a non-trivial

[3]The Estimation of Bayesian Networks Algorithm (EBNA) is an exception, since it takes the current network as a starting point [10].

problem. The EDA approach is more expressive and adaptive but adds a work overhead in attempting to discern the underlying search distribution. Even if the class of distribution is known, the specific structure needs to be revealed.

Both approaches have a degree of success which may well be compounded by a model that combines probabilistic sampling with an adaptable population structure.

1.5.5 Symbiotic Algorithms

An alternative approach to both the population-based and the probability-model approach is inspired from the biological phenomenon of symbiosis highlighted earlier.

One computational abstraction of the symbiotic mechanism is given in the *compositional algorithms* approach. First presented as the *Symbiogenic Evolutionary Algorithm Model* (SEAM) [49] and later optimised into the *Hierarchical Genetic Algorithm* (HGA)[19] these base their search on an adapting set of modules rather than individuals. This contains only the most primitive modules, i.e. single bits, initially and, conditional upon validation, these are incrementally and irreversibly joined to explicitly construct new building-blocks.

Both SEAM and HGA validate candidate modules via a rigorous testing phase. Following this, specific combinations of modules are removed from the pool, effectively reducing the search space. If the test is unreliable due to an inadequate number of test cases then global optima may be permanently excluded from the search. Despite a history of evaluated individuals, candidate modules are selected either randomly (SEAM) or in an arbitrary order (HGA). Furthermore, SEAM discards individuals which have been used for validation testing. HGA retains and recycles a population of individuals but does not use this to suggest modules which may be valid building-blocks. Therefore, the compositional algorithms above do not fully exploit

the information regarding likely building-blocks that may be present in a population.

Another very recent algorithm is the *Evolutionary Transition Algorithm* (ETA) [8] which combines the principles of natural selection (implemented in a classical, generational GA) with an interactional framework derived from transitional biological theory. Heavily inspired by biological theory, it is concerned with developing the link between evolutionary transitions and compositional models; and proposes the term *Compositional Search Evolutionary Algorithms* (CSEA) for similarly inspired algorithms.

1.6 Hypotheses

Our thesis is founded on the proposal that a selecto-transmutative process can incrementally adapt a search model (as defined earlier) from the simplest to an appropriate degree of complexity for highly-structured problems. This relies upon a series of dependent hypotheses. Each of these will be explored and tested individually as far as possible. Only then will we be able to delineate its limitations. Specifically, we will be verifying the following:

1.6.1 Population as a Structured Sampling Model

Hypothesis *An elitist population of solutions, composed from multivariable components, can be sampled as an expressive and compact search model.*

This concerns the adequacy of the search model and proposes it is capable of representing appropriate search distributions for a wide range of structured problem classes. If it is not able to model the structure of any of these problems then it would be expected to be severely curtailed its ability to solve them. Sub-optimal fixation is addressed by ensuring primitives have a baseline relative weighting. We can verify

its accuracy in discerning problems with a known structure and comparing this with the explicit resulting TEA model.

1.6.2 Transmutation for Structural Adaptation

Hypothesis *Structural mutation acting at the individual level can, in principle, adapt the search model to reflect the genetic interactions of a problem.*

Having the capacity to represent an appropriate search distribution with the structured population model from the first hypothesis is of little value unless this can be reached from the initial unstructured population. Our proposed operator for this is structural mutation (which we refer to as *transmutation*) and is applied uniformly to individuals in the population.

1.6.3 Structural Validation via Individual Selection

Hypothesis *Selection can efficiently amplify and distinguish modules, in parallel, which encapsulate valid building-blocks, and demote and remove invalid modules.*

For structural adaptation to progress effectively will require not only genetic substructures to be produced (as for the second hypothesis) but for these to be promoted or discarded. We can employ the same selective dynamics that allow genes to compete to allow modules to compete with their components.

1.6.4 Intersection of Compositional and Probabilistic Models

Hypothesis *There exists a model which unifies the most significant elements of the CSEAs with those of the EDAs to obtain key advantages over both of these.*

Specifically, we are looking to combine the explicit representation of modules from compositional algorithms with the explicit frequency modelling of EDAs. The result

is expected to be a more general model that may be applicable to a broader range of problem classes than these two approaches individually.

1.7 Contributions

Whilst the prior techniques described demonstrate various different strengths and weaknesses, we propose the key advantages are not mutually exclusive and can be unified in a straightforward and reasoned way.

Specifically, we suggest that it is not necessary to supplement the population with a separate probabilistic model if the population itself is able to adapt its own composition in terms of structure as well as frequency. Neither is it necessary to perform a strong conditional test for incremental structure in models if we can subject genetic structures to the same selective processes we use to compete individuals or alleles.

Our method proposes to combine the explicit modules of the compositional representation and explicit frequency over these, as per the EDAs, in a generalised scheme. This is expected produce efficiency gains by removing the need to repeatedly search for a separate probabilistic model whilst providing an effective bias toward testing modules with a greater prospect of success. The specific contributions resulting from this work are identified now.

1.7.1 Analysis of Overlap with GAs, CSEAs and EDAs

The theoretical contribution is in analysing the operation of compositional and probabilistic models to highlight aspects which might be rationalised in order to overcome their individual drawbacks.

1.7.2 A Transitional Evolutionary Algorithm

The practical contribution is a novel algorithm from an interpretation of existing models that combines a symbiotic genetic representation of weighted, overlapping modules with a method for adapting this incrementally. We argue that this more powerful representation enables it to effectively discern and apply problem structure purely from selective dynamics. This makes it effective on a broad range of problems classes, from those with few variable dependencies to those with hierarchically-structured dependencies.

1.7.3 Empirical Comparison of Relevant Algorithms

Empirical comparison of relevant algorithms on an range of problem classes ordered by structural complexity. This not only directly compares algorithms which share a problem niche but also identifies where the bounds of their competence lies.

1.8 Methodological Principles

Earlier designs for evolutionary algorithms were naturally a product of a particular biological abstraction (crossover and mutation acting on chromosomes processed by individual-level selection) and a particular computing environment (a fixed-size population of fixed-length, unstructured binary strings). As evolutionary paradigms and computing environments increase in sophistication there is a tremendous scope for correspondingly complex models. This does not necessarily imply that they are more useful or illuminating; the opposite is likely to be true. We should resist being seduced by our own ingenuity and adhere to the more minimalist approach, extending our model only when it proves inadequate.

As far as possible, each design decision and extension needs to be justified within the context of the algorithm without reference to hazy intuitions or Evolutionary Computation lore. The benefits of adhering to this principle are multiple with the most significant are listed here.

1. **Comprehension**. A simpler algorithm is easy to analyse — both formally and informally — as well as implement and debug. The fact that the problem may be a black box does not imply the algorithm needs to be likewise.

2. **Adoption**. Ease of comprehension removes a key barrier to widespread adoption. Although there are many superior alternatives to the canonical GA, variants of it are often still applied because of its appealing and intuitive operation. A real achievement for this work would be an algorithm which was similarly intuitive yet surmounted key issues which beleaguer established algorithms.

3. **Generality**. When an algorithm has been extended only reluctantly, it is in less danger of chasing improvements in a smaller class of problems. Simpler algorithms are more general algorithms.

1.9 Summary

Neo-Darwinian evolution is an established paradigm for artificial optimisation techniques but a mechanism for increasing adaptive complexity remains an active research area. The biological theory of evolutionary transitions to produce adaptive complexity via symbiosis [29, 30] is a possible contender to implement the principle of a building-block process, long held in evolutionary algorithms.

We use a scale of black-box combinatorial optimisation problem classes in order to classify degrees of structural complexity. Modular problems and hierarchical problems

are of particular long-standing interest. Solving such problems clearly requires the identification of the structure along with the values. Two recent approaches to this are identified.

Compositional models, inspired by evolutionary transitions, explicitly seek and encode this structure. These can efficiently solve hierarchical problems via a set of strong assumptions regarding the composition of the pool and the limitations on its complexity. Alternatively, probabilistic models aim to infer the genetic linkage from the population as a sample of fit solutions. They work by substituting the genetic operators with a more general sampling of an explicit probabilistic distribution.

We identified a shared weakness across these in how they used evaluations to identify, represent and exploit problem dependencies. This led us to our hypotheses, contributions and the methodological principles we will be following throughout this work.

1.10 Dissertation Structure

In this chapter we have set out the background for our research. This starts with the inspiration from biological concepts for the emergence of complexity and its relation to the open and active issue of scalability in artificial optimisation. A description of the problem domain follows along with an overview of previous work, connected to the area of interest. The hypothesis and contributions conclude the chapter.

In chapter 2 we consider the inadequacy of recombination for performing a building-block process and go into more depth regarding alternative methods to the GA. We draw out the significant aspects of these to mark out the niche we will be attempting to occupy.

For chapter 3, with the help of a visual notation, we delineate a limited unification

of aspects of GAs, CSEAs and EDAs. Within this context we introduce our notion of a virtual pool of components drawn from genetically structured individuals and the genetic operator for adapting their structure.

In chapter 4 we formally describe the structures and methods of the TEA: a simple, novel and efficient algorithm which implements the principles and methods previously described. This is followed by comparison of the TEA to related algorithms and concepts.

An empirical validation is carried out in chapter 5 traversing the scale of problem structures. Related algorithms are compared to each other on instances of appropriately structured problem classes as the various facets of TEA are tested and, where indicated, revised.

Chapter 6 interprets the results of the experimental studies in light of the original hypotheses, concludes on the contribution of the thesis and proposes future directions.

Chapter 2

Background Review

In this chapter we review related work regarding their motivations and principles. We identify commonalities between them, suggest their individual limitations and draw out some themes in preparation for the development of a novel approach.

2.1 The Canonical Genetic Algorithm

The widespread adoption of the canonical GA is at least partially attributable to the ease with which it can be understood and implemented. Although many more sophisticated evolutionary algorithms have been developed, the simplicity of the GA means that close variants of it continue to dominate in application. This is particularly true outside the EA research community. We now inspect its main feature of interest: the use of recombination to produce new solutions from the population.

2.1.1 Recombination Biases

The operation of the GA associates genes in two distinct ways. Firstly, crossover is restricted to a pair of parents rather than combining genes from a larger subset of the

population. This has the overall effect of increasing the probability that co-occuring genes from an individual will remain together. The recombination operator itself may apply a further, more nuanced, bias based on the ordering of genes. For the original single-point (and later 2-point) dependent genes that span fewer variables (their defining length) are less likely to be disrupted once they emerge. This has been well studied in a detailed investigation by Spears [7] and is referred to as a *positional bias* by Hinterding [17].

The *uniform crossover* operator [44] removes the positional bias by making crossover at any point independent of any other, to occur with a probability of 50%. Hinterding defines this as the *distributional bias* since half of the genes are expected to be sampled from each of the two parents (though the genes may be the same). Spears investigated generalising uniform crossover probability as an algorithmic parameter and found that the optimal crossover rate varied over time [43].

We label recombination as a static operator because its bias does not change over the course of an evolutionary run. Any set of variables from an individual have the same probability of being crossed over as they start with. This is irrespective of the values of those variables or the relative success of prior recombination operations. The motivation to move from a static to an adaptive bias arises from the conditions for a building-block process.

2.1.2 Supporting a Building-Block Process

Whilst there is some theoretical backing for the canonical GA supporting this process in the schema theorem, there are also a number of hard caveats. We summarise here:

- *Building blocks must be constructed.* This requires active search within the particular partition which could come from either mutation or crossover.

- *Building blocks must be preserved.* This is rather contrary to the objective above. The conventional heuristic is to attempt to ensure building blocks have a short defining length, so that positional crossovers are less likely to disrupt them.

- *There must be a sufficiently rich pool of building blocks in the population.* Not only must building blocks not be disrupted, but they must persist long enough in the population to enable *mixing*, i.e. the conjunction of higher-order modules.

Thierens, analysing the mixing issue [46], showed that the conditions for which that can perform (the 'sweet spot') are rather constrained, required tightly-linked genes. This issue was shown not to be resolved by elitism, niching or restricted mating. The need to avoid disruption is especially problematic since it requires prior domain knowledge and the capacity to translate this into the representation. Thierens concludes that:

> ...competent, scalable genetic algorithm performance can only be obtained when building block linkage information is already in the problem encoding or is identified before or during the genetic algorithm's search. The scalability of genetic algorithms depends critically on the representation. Once the appropriate schemata are identified, the problem becomes "mixing-easy", and fast, reliable, scalable evolutionary computation can be achieved.

By precluding prior dependency information[1] we must discern it as the search progresses via a process known as *linkage learning*.

[1]Theirens refers to it as the 'no-linkage information assumption'. It is synonymous with the 'dependency ignorance' rule in section 1.4.

2.2 Linkage Learning Algorithms

In this section we briefly review a selection of *linkage learning* algorithms. This class of self-adaptive evolutionary algorithms extend the search model provided by the population with acquired information regarding the structure of the problem, thereby avoiding the conventional GA necessity of an appropriately biased encoding as outlined earlier. Instead, they are set up to identify variable dependencies during the search and respect this genetic linkage via auxiliary structures. These may adapt the structure of the individuals in some way, e.g. by re-ordering the variables or by explicitly linking variables or values.

The Messy Genetic Algorithm (mGA) [13] was an early attempt to allow evolutionary adaptation to find not only an optimal set of values but also to allow it to order the variables on the chromosome. It aimed to do this during a 'primordial' phase via specialised operators such as inversion before combining building-blocks in a juxtapositional phase. The anticipated effect of this was that individuals which had grouped interdependent genes would be less susceptible to deleterious recombination and would come to dominate the population. It had limited success being somewhat brittle in its operation. This was extended into the Gene Expression Messy Genetic Algorithm (GEMGA) [20] which was guided more by an inductive process.

The Linkage Learning Genetic Algorithm (LLGA) [14] was similarly inspired by the mGA and extends it with several novel techniques to reconcile building-block mixing with allele selection.

2.2.1 Compositional Models

The compositional approach actually performs a specific case of linkage learning in that it identifies values for pairs of variables and not just the variables themselves.

It is strongly based in a symbiotic view of evolution, first advocated by Margulis [29]. It was formulated by Watson [49] in the Symbiogenic Evolutionary Algorithm Model (SEAM) and later optimised by De Jong et al. [19] in the Hierarchical Genetic Algorithm (HGA). The compositional algorithm has also been framed in terms of complex dynamical systems [47].

A very recent work, the *Evolutionary Transition Algorithm* (ETA) [8] stays relatively close to the biological metaphor of cooperative co-evolution and does not claim to be competitive with other optimisation algorithms. Nevertheless, it demonstrated competitive performance against SEAM on the H-IFF problem and effectiveness on the *Binary Contraints Satisfaction Problems* (BINCSP) where SEAM was ineffective.

Definition of SEAM

SEAM was specifically developed to tackle the hierarchical class of problem, instantiated by the recursively-structured IF-and-only-IF (H-IFF) problem which was shown to be extremely challenging for existing evolutionary techniques [48]. The H-IFF function was defined as:

$$F(B) = \begin{cases} 1, & \text{if } |B| = 1 \\ \substack{|B|+F(B_L)+F(B_R),\\ F(B_L)+F(B_R)} & \text{if } |B|>1 \text{ and } (\forall i{:}b_i{=}0 \text{ OR } \forall i{:}b_i{=}1) \\ & \text{otherwise.} \end{cases}$$

Where B is the set of features, $(b_1, b_2, ..., b_k)$, $|B|$ is the size of the set$= k$, b_i is the i^{th} element of B, B_L and B_R are the left and right subsets of B, i.e. $B_L = (b_1, ..., b_{k/2}), B_R = (b_{k/2+1}, ..., b_k)$. The length of the string evaluated must equal 2^p where p is an integer (the number of hierarchical levels.)

Instead of a population of individuals, its search model comprised a set of modules or partial solutions. This initially enumerated all the primitive modules; for the H-IFF problem this meant all possible bits. Individuals are generated by adding modules from the set until all variables had been assigned a value.

Rather than maintaining and adapting individuals, the search model represented by the pool was adapted by performing irreversible joins on modules following a successful testing phase. The test compares the candidate join with its independent components in a number of contexts randomly generated from the remaining components. Only if the join was shown to *Pareto dominate* the independent components[2] was the join deemed to be 'stable' and made permanent. If this test failed then the pool structure (and therefore the search model) remained unchanged. A pseudocode description of SEAM is given in algorithm 2.1.

This testing can be viewed as performing the role that competitive selection might otherwise be expected to carry out. If a module Pareto dominates its components, this implies ongoing selection between the containing individuals would ultimately cause the module to displace the independent components from this niche. This notion is developed more fully later.

Definition of HGA

The HGA optimised SEAM in several key ways:

- It maintained a population of individuals specifically for the purpose of testing the validity of candidate joins. In this way it was able to recycle individuals rather than generate them *ad hoc* and thereby reduce the number of required evaluations.

[2]The join would Pareto dominate the independent components if there was at least one context in which it was fitter than either of the components and none where it was less fit.

```
SEAM main()
initialise pool of single-value modules as M
repeat
    uniformly sample modules a, b from M
    if is_stable(a, b) join a, b in new module
until terminating criterion met

// test whether join dominates component, i.e. is context optimal
// note: c_x denotes the values of x superimposed onto c
is_stable(a, b)
    dominates = false
    for i = 1 to n
        c = sample_context()
        if f(c_a) > f(c_ab) or f(c_b) > f(c_ab) return false
        if f(c_ab) > f(c_a) or f(c_ab) > f(a) dominates = true
    return dominates

// sample an individual for join test
sample_context()
    context = fully unspecified solution
    M' = set of applicable modules initialised from M
    while context still has unspecified variables
        uniformly sample component m from M'
        remove from M' all modules that intersect m
        add m to context
    return context
```

Figure 2.1: Pseudocode for SEAM

[note distinct use of term module as set of variables rather than values]
HGA main
$k \leftarrow$ maximum order of module considered
$M \leftarrow$ pool initialised with the set of single-variable modules
initialise *pop*
$modified \leftarrow$ true
while $modified$
    evolve_pop
    $modified \leftarrow$ **module_formation**
    replace_modules

evolve_pop
do
    $f_{max} \leftarrow$greatest fitness in population
    remove all samples with fitness less than $f_{max} - \epsilon$
    $new\_individuals \leftarrow$ false
    while pop size $<$ n
        $new\_individuals \leftarrow$ true
        sample new individual
while $new\_individuals$

module_formation
$formed \leftarrow$false
$k' \leftarrow 2$
while $k' \leq k$ and $\neg formed$
    if $k' > |M|$ return false [insufficient modules for a join]
    $M^{k'} =$ all k-order subsets of M
    for each $m \in M^{k'}$
    $nrsettings \leftarrow$ number of possible settings for m
    $COsettings \leftarrow$ number of context-optimal settings for m
    if $COsettings < nrsettings$
        $M \leftarrow m \cup M \setminus \{c|c \in m\}$
        $formed \leftarrow$ true
    increment k'
return $formed$

replace_modules
for any module combination in the population that forms a composite module
    if the composite modules is ϵ-optimal
        replace the combination with the composite module

Figure 2.2: Pseudocode for HGA

- It considered all the possible joins in a systematic order. Of course, given no prior knowledge or applied feedback, this ordering is necessarily arbitrary. The main benefit of this, we conclude, is to ensure that no module is considered multiple times.

- It removed small numbers of settings frequently, rather than removing large numbers more infrequently. Given the incremental nature of the search, this was a significant optimisation.

The above points signal a subtle difference in the concept of a module between the SEAM and the HGA. In the SEAM[3], a module is a fixed set of values analogous to a GA schema. The HGA has a module as a set of variables, akin to the partitioning approach of ECGA [15] and GEMGA [20] where the search space is constrained according to proposed dependencies.

Another fundamental difference in the approach between these two algorithms is shown by their policy in constructing new modules. Whereas SEAM only allows the construction of a module following stringent testing, the HGA assumes any considered module to be valid unless its distribution in the population indicates otherwise. This means that any module validated in SEAM is likely to represent an actual value dependency whereas a module under HGA is a suggested variable dependency with value dependencies yet to be discounted.

There are a number of strong issues with both these algorithms of the compositional approach, largely based around the nature of the modularising operation.

Adequate sample size The join operation is irreversible and incrementally restricts the search space. Therefore it is crucial that the process by which candidate

[3]Also in the earlier preliminary HGA framework [5].

modules are tested is reliable. A sufficiently large number of testing samples is advised to ensure this although too many additional testing samples will clearly incur an unnecessary overhead.

Primed pool Another issue concerns the initial composition of the set of primitive modules. Each of these can only be used once and therefore makes the module set a finite source of building-blocks. Defining the initial module set composition such that the optimal solutions, and all their ancestors, are not artificially constrained is what we call the 'model-kit' assumption.

Arbitrary module selection Both SEAM and HGA use the population solely to test candidate modules. This ignores available statistics in the population which may be used to suggest modules.

An approach of 'soft' joins was marked by Watson [49, p302] as a prospective line of research. This was conceived as incorporating some kind of explicitly probabilistic component[4]. However, this variation remains as yet untried.

2.3 Probabilistic Models (EDAs)

Although the algorithms up to this point clearly have a probabilistic aspect, e.g. in the selection of individuals and crossover actions, we turn now to algorithms which are entirely based on an explicit probability distribution to generate individuals.

We review them in a chronological order which roughly corresponds to an increasing range of dependencies they are able to model. The earlier ones model allelic

[4]In personal communication.

frequencies whilst the later are able to model population distributions where the distribution of alleles is dependent on the other loci, known as *linkage disequilibilium*.

2.3.1 Unstructured Population Modelling

The most general probabilistic population model assumes no dependencies between the variables. These univariate models simply sample each value from a vector of independent distributions across each variable. This structure therefore models allele frequencies but not linkage disequilibrium. Although their models are equivalent, the process by which they are derived and adapted differ. We review these for the purpose of mapping out the development of the early work. More detailed surveys can be found in [52, 38].

The earliest of these models was *Bit-based Simulated Crossover* (BSC) [45] which retains a population and samples values from it according to their fitness weighted frequencies. This is effectively fitness-proportionate selection from within a set of individuals; alleles are totally independent of each other. Earlier versions used mutation but this was discontinued later. The *Univariate Marginal Distribution Algorithm* (UMDA) [33] differs from the BSC algorithm in that the probability distribution over each variable is derived from the relative frequency of its values within a subset of a previous population sampling. The subset is obtained via tournament selection which reduces premature convergence since large differences in fitness do not equate to a proportional representation in the next generation.

The *Population-Based Incremental Learning* algorithm (PBIL) [1] frames the adaptation of allelic frequencies as a type of incremental learning using a simple update rule to move the population-modelling vector toward the best in each generation. The *Compact Genetic Algorithm* (CGA) takes a more nuanced approach,

sampling a pair of individuals at a time and adjusting the vector in favour of the fitter for each variable where the values differ [16].

2.3.2 Structured Population Modelling

The Bivariate Marginal Distribution Algorithm (BMDA) [36] is a direct extension to the UMDA [33] which uses Pearson's χ^2 statistics to indicate pairwise dependencies between variables.

The Bayesian Optimisation Algorithm (BOA) [34] aims to construct a Bayesian network to model each generation then samples from this to produce the next generation. This is a powerful representation but does require the definition of a metric to compare the quality of models, as well as a secondary search of the space of these. A related algorithm is the Estimation of Bayesian Network Algorithm (EBNA) [10] which differs in that it takes the previous network as a starting point for the successive search.

The *Hierarchical Bayesian Optimisation Algorithm* (hBOA) extends the BOA using hierarchical local structures and a niching technique called Restricted Tournament Replacement (RTR) [37]. Whilst this algorithm is powerful in application, it is more of a careful application of existing complementary techniques than a novel design in itself.

The *Extended Compact Genetic Algorithm* (ECGA) [15] is founded on the principle that the choice of a good probability distribution is equivalent to learning linkage. It operates by repeatedly searching for a partitioning of the variables into sets which appear to represent a compact model of the population[5]. Compactness here is equivalent to *Minimum Description Length* (MDL). Following this, each combination of values is temporarily treated as a module and the relative frequencies of each is ex-

[5]According to the Kullback-Liebler divergence [22].

```
EDA main
initialise sampling distribution model
repeat
    replace population with solutions sampled from model
    derive new model of better solutions from population
until terminating criterion met
```

Figure 2.3: Pseudocode for EDA

tracted to complete the *Marginal Product Model (MPM)*. The MPM is a probabilistic model, like the model from its predecessor CGA[16], but it is selecting entities that are larger and fewer than individual genes. The ECGA has since been extended from its binary form to one with arbitrary cardinalities in the χ-ary ECGA [26].

2.4 Comparing CSEAs and PMBAs

The compositional approach and the structured EDA approach have unrelated origins. The former from biological theory for the emergence of complexity, the latter from statistical machine learning. Despite this, there are various shared motivations, principles and practices between them that can be illustrated with the forerunners of the HGA and the ECGA.

In terms of representation, the MPM from the ECGA resembles the module pool from the HGA in that it specifies partitions of the variables. However, the frequency-derived graduated weighting of the MPM is more expressive than the compositional pool which simply allows or discounts settings.

Both the ECGA and the HGA take a similar systematic, exhaustive, and greedy approach to partitioning the variables to derive the modular structure of the population. They consider all possible partition-pairs and retain them as partitions if they pass specific criterion. In HGA, pairs of genes are discounted if they are absent in an

elite subset of a sample population. A partition which contains less than a complete set of configurations (i.e. possible value combinations) is considered a module. In ECGA, the pairing which leads to the greatest decrease in complexity results in a merger. This is repeated until the combined complexity can be reduced no further at which point the final MPM is used to generate the next population.

Both techniques also place an emphasis on simpler models over more complex ones. The HGA starts with the simplest possible model and builds up. The ECGA directly favours lesser minimum description lengths.

The modules of HGA persist and incrementally merge throughout the run. It does not have generations as such. The population sample is entirely conditional; individuals are only replaced within it when they become incompatible with the module pool. The MPM, in contrast, produces a fresh dependency model for each generation and discards the previous model. The HGA representation is therefore more efficient since it builds on the structural model of value dependencies incrementally and irreversibly.

2.4.1 Summary of a Meta-EA

Having described the workings of the individual algorithms and made comparisons between aspects of them we can envisage an abstract form which encompasses them. With this view we aim to draw out their essential commonalities whilst highlighting their significant differences. Figure 2.4 summarises this view. It takes each algorithm as employing a solution set with some rule for replacing this. The solution set is generated from a search model which may itself be relatively simple function of the solution set or the result of a more sophisticated process.

The proposed algorithm, TEA, is included for easy comparison with a fuller introduction to follow directly.

Evolutionary State Comparison

Search Model Form and Derivation
SGA A population (i.e. the solution set) to which predefined genetic operators (mutation and recombination) are applied. **EDA** A probabilistic distribution which is generally derived at each iteration from the solution set. **CSEA** A persistent set of unweighted solution components. Incrementally joined conditional on the composition of the solution set. **TEA** A persistent, adaptive structuring of the solution components that form the solution set.
Solution Set Maintenance
SGA Some solution persistence in set possible via elitism. **EDA** Set resampled from search model for each generation. **CSEA** Set regenerated from model at each iteration (SEAM) or maintained according to model (HGA). **TEA** Persistent set of unique solutions with specialised elitism.

Meta Evolutionary Algorithm

1. Initialise search model to simplest, unbiased distribution. 2. Sample number of solutions from model.
SGA applies genetic operators to produce solutions for next generation. **EDA** samples new set of solutions according to model. **CSEA** generates or maintains set of solutions from pool of modules. **TEA** samples a candidate solution from the weighted components.
3. Derive new search model.
SGA applies selection to replace less fit solutions with new solutions. **EDA** derives weighted, structured model from sampled solutions. **CSEA** adds structure to model, conditional on the solutions set. **TEA** adapts model structure and weighting via specialised selection.
4. Repeat from step 2 until terminating condition met.

Figure 2.4: Comparison of state and pseudocode for 'meta-EA', emphasising commonalities and differences between the Simple Genetic Algorithm (SGA), Estimation of Distribution Algorithm (EDA), Compositional Search Evolutionary Algorithm (CSEA) and the proposed Transitional Evolutionary Algorithm (TEA).

2.5 Principles of Evolutionary Optimisation

2.5.1 Population Usage

We look at past solutions in order to suggest future ones and since we are interested in generating solutions of higher fitness we tend to consider only a subset of previous solutions. Retaining a set of high-quality individuals, a population, allows us to infer more about the relative quality of particular values — and combinations of values — than just a single champion individual. Furthermore, the more distinct individuals we can refer to, the more information we have about the problem. However, some individuals are more informative than others regarding prospective high-fitness individuals; these tend to be the ones with a higher fitness themselves and so we retain these in the population and discard those less fit. From this point, the population may be used in various ways:

Fitness function information Everything known about the problem is rooted in the raw feedback contained in evaluated individuals. In this capacity, the population may be viewed as a set of inputs which map via an unseen function to a range of outputs. Our task is to produce a search distribution to maximise the final output. EDAs generally pursue this by repeatedly modelling the structure of an elite subset of each generation, or adapting the weights of a predefined structure.

Modelling Search Distribution Noting here that duplicate individuals contain no more information than a single individual, convergence represents a particular loss of information. Frequency only becomes useful if we intend to extend the population into a complete search model. The canonical GA uses the implicit frequency information of genes and schemata by allowing duplicates of individuals. This ensures the

reinforcement required by the schema theorem. The selection probability of the fitter individuals (and their containing schema) is increased in proportion to their frequency which reflexively acts to increase the frequency. The end result is the allocation of the finite population to the fitter schema.

Validating Building-Blocks The compositional algorithms retain a population to judge the validity of any specific module as a building-block. The test is in the form of a comparison between a specific module and other modules specifying the same variables. New building-blocks are identified when a particular module is found to dominate other combinations of components. This essentially anticipates the fixation that schema processing might be hopefully expected to produce.

In fact the latter two usages are fundamentally the same: some schema (or modular settings) are amplified at the expense of others following the consideration of a number of solutions which contain them. Whereas GA models do this gradually and implicitly, the compositional algorithms do explicit checking.

2.5.2 Processing Partial Solutions

A key feature posited of recombination-based EAs is the ability to combine structures from different lineages. This is the so-called building-block hypothesis, yet it imposes a number of highly taxing conditions in order to operate. Although a population is able to implicitly hold individuals with genetic linkage and amplify them, the static linkage from conventional crossover operators is generally disruptive for problem with much structure. Mixing is therefore limited.

Self-adaptive linkage-learning algorithms aim to support mixing (i.e. reduce disruption) by adapting the action of recombination over time. Compositional algo-

rithms essentially only perform mixing, with no predetermined bias on genetic linkage. They are able to incrementally discard genetic substructures as an intrinsic part of the search. Yet they lack the adapted bias of a selective weighting over the modules and therefore select candidate modules regardless of the population.

EDAs support such a weighting, and more sophisticated ones are able to generalise the modular structures represented by the compositional algorithms. However, unlike the incremental compositional approach, the search for dependency structure must be reset at each iteration. The structure does not persist with the lineage.

2.6 Gaps in the Literature to be Addressed

Both the GA approach, motivated by the building-block hypothesis and strongly expressed in the compositional algorithms, and the EDA approach appear to be distinct alternatives, yet they share key facets. They both attempt to discern variable interactions from a population and adapt the search distribution accordingly. In the next chapter we go into more depth regarding how these methods actually intersect.

Interestingly, the point at which these methods meet is also where they diverge from their evolutionary inspiration. The EDAs interleaves a model derivation with each generation and the CSEAs attempts to modularise components. The CSEAs have some basis for structural adaptation in evolutionary transitions yet this has not been abstracted out of the biological metaphor into a generally applicable method that addresses the core principles that motivate recombination in population-based algorithms.

If the population is a model of a good search distribution, why do we have to model the model? We would suggest that the need for a separate structural model of the population arises purely from the fact that it conventionally contains only flat

individual structures. Compositional algorithms do not require a separate model to be constructed; the dynamic pool of sub-structures incrementally grows to order but still only use individuals for testing. The compositional testing is a substitute for repeated selection.

We would like to not have to perform structural search separately from the search for values. We would prefer to define a richer population representation that allows structure to be adapted concurrently with values.

2.7 Summary

We began this review by focussing on the distinguishing feature of recombination in GAs and how it is poorly equipped to fulfil the requirements of the building-block hypothesis. A selection of linkage learning algorithms were reviewed, culminating in the compositional models which we critically examined.

The alternative approach of the probabilistic models was described for models of increasing complexity before highlighting their commonalities and differences. We pay particular attention to how they use the information within the population and how partial solutions are processed and combined.

We conclude the review by marking the gap in the previous models that we aiming to fill in successive chapters.

Chapter 3

Towards a Transitional Model

In this chapter we draw out and develop ideas and motivations ear-marked in the review to provide justification and lay the foundations for a transitional model.

We start by taking the perspective of selection with which to view the algorithms of interest, where they deal with selectable entities at various scales. We go on to describe how the same competitive pressures resulting from selective processes might be used to validate genetic structures as modules competing directly with their components.

A simple visual notation has been found to be helpful in illustrating the discussion. This uses blocks of differing shades to represent alternative values[1] for variables and sets of variable. A bordered block, or set of blocks, represents an indivisible, selectable entity or unit of selection, e.g. an individual or component. The relative size of a block reflects its sampling probability. Blocks which overlap along the x-axis are mutually exclusive in an individual.

For simplicity, we only use this visualisation for structures which span adjacent variables.

[1]Limited to binary values for our purposes of this analysis.

3.1 Selection and Sampling

Selection, the act of picking one structure rather than another to transmit into a future population, is usually associated with the GA approach where the individual is the structure being transmitted. We will argue here that when it is applied stochastically, as in fitness proportionate selection, the GA approach is distinguished from probabilistic models only by the scale of the entities being selected. GAs apply selection to individuals whereas probabilistic models apply selection to the entities that constitute individuals. In selecting an individual, we are effectively sampling all of the elements within it. Therefore the key to a coherent compound lies in framing GAs as simple probabilistic models where individuals are the aggregated entities being sampled. This interpretation is illustrated in figure 3.10.

We now expand this by considering GA selection in isolation from genetic operators before we consider the GA operators of mutation and crossover specifically. Then we turn to examine EDA sampling in isolation from the model search.

3.1.1 A Perspective in Competition

The following thought experiment may provide a valuable intuition for the abstract interpretations which follow it.

Consider a group of individuals that are effectively genetically identical, i.e. a species, denoted S. Assume a particular mutation within S which prevents breeding with S but is still competing for the same niche. This constitutes a new species, S'. If there is a phenotypic difference which gives members of S' a fitness advantage over those of S then it will be expected to eventually take over the niche.

Note that we have not detailed the nature of the mutation or the changed interaction with the environment. Suffice to say that the species are genetically similar but

distinct enough to constitute a different species which continues to compete within the same niche.

The point is that from a species perspective, one is taking over from another. However, from the perspective of the genes common to both species, there is nothing of significance happening. It is only *differing genes* that are competing to persist and replicate where change will occur. The relevance for this thesis is in terms of the genetic structures we support within individuals.

3.1.2 Amplification from Selection (GA approach)

The role of selection in GA models is to adapt the frequency of competing individuals rather than to generate novel ones. This is essential to raise the odds of building-blocks being combined. Selection can be deterministic or stochastic occurring with a probability that increases with fitness, either proportionately or according to some other mapping, such as linear ranking or truncation selection.

This mapping, illustrated in figure 3.1, shows the selective probability of the individual as a function of its ranking within the fitness distribution for the entire population. If a population can contain multiple copies of an individual then this multiplies the selective probability for that individual. The resulting positive feedback loop is intended to amplify the difference in selective probability arising from the difference in fitnesses and focus the processing on the more promising regions [18]. When this feedback from frequency is too strong then promising leads are lost and premature convergence is said to occur. Selective pressure needs to be reduced in this case.

At any point our best solution is that with the greatest fitness. Yet we also select other individuals because there is a possibility that they could contribute to better

solutions in the future. However, the best mechanism for achieving this is unclear. GA selection does not in itself perform exploration whereas EDA selection, or sampling, does. A comparative example of this is contained in figure 3.10.

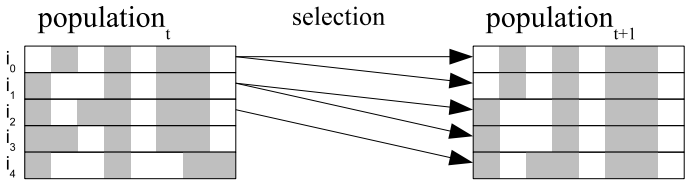

Figure 3.1: Population selection amplifies individual frequencies.

3.1.3 Exploration from Selection (EDA Approach)

Within simple EDAs, we can consider the sampling of values from the variable probability distributions also to be a form of selection akin to the selection of individuals within GAs. Values compete with other values that occupy the same variables and co-evolve with other values for other variables.

In effect, we are selecting from *within* individuals, allowing us to explore novel ones. The scale of the selected entity (allele rather than individual) is the difference which fundamentally changes the role of selection as illustrated in figure 3.10. Whilst it is still responsible for adapting the frequency of entities it is now also the operation which produces new and novel solutions; it is responsible for exploration. This is how it is able to assume the traditional role of recombination and mutation as genetic operators.

For a set of alleles, A, the probability of sampling allele a at position i is given by ω:

$$p(x_i = a) = \frac{\omega_i(a)}{\sum_{a\prime \in A} \omega_i(a\prime)}$$

We currently limit our scope to $A = \{0, 1\}$ and normalise ω, i.e. $\sum_{a \in A} \omega_i(a) = 1$.

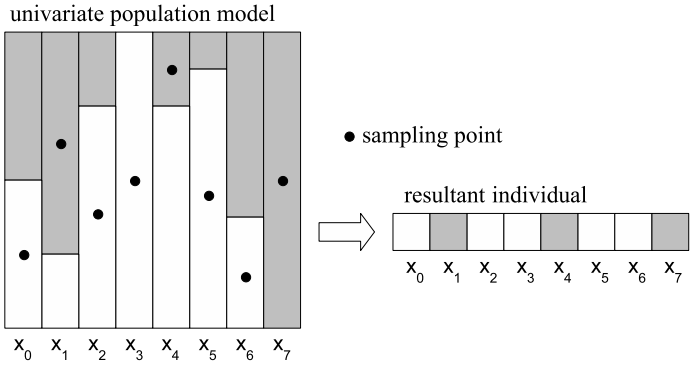

Figure 3.2: Univariate model sampling defines a non-zero search distribution over 2^6 individuals (x_3 and x_7 have converged to a fixed value).

The probability of sampling x from this distribution is then:

$$p(x) = \prod_{i=1}^{n} \omega_i(x^i)$$

For the plot, the relative value of $\omega_i(a)$ is then indicated by the relative height of the corresponding bar in the i^{th} column.

3.2 GA Operators as Sampling Operations

Our aim here is to show how a simple model based on probabilistic sampling can subsume the essential functions of the GA genetic operators. The capacity to model linkage is extended later.

3.2.1 Mutation as Sampling

The classic rationale behind the mutation operator is to provide 'diversity' to the population, yet it does not take much probing to find the weaknesses in the standard form. The most prominent of these is in its fixed level of disruption. As the search

state becomes more highly adapted, the action of mutation becomes increasingly likely to be decremental. Certain adaptive variants have been proposed to reduce the mutation rate over time linked either to time or fitness [17]. However, this feels like a patch rather than a resolution.

A better approach becomes apparent if we consider mutation in terms of value sampling. With this interpretation, mutation is simply sampling from a uniform set of individuals with a fixed, low probability. Figure 3.3 shows a graphical representation of this.

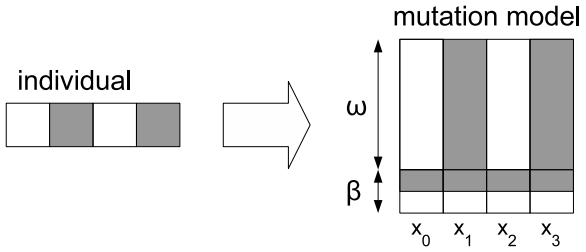

Figure 3.3: Simple mutation as sampling

It shows a random-replacement mutation operator acting with a per-locus probability of β. This gives our selective probabilities for our values as:

$$p(x_i = 1) = \omega_i + \beta/2 \quad \text{and} \quad p(x_i = 0) = 1 - p(x_1 = 1)$$

Where the weighting vector ω is simply set by the current individual which represents the search state.

$$\omega_i = \begin{cases} 1 - \beta \text{ if } x_i = 1 \\ 0 \text{ otherwise} \end{cases}$$

By further defining the replacement rule for the current individual we fully emulate an ES-style search.

This problem is that, just as random search ignores the current best solution in generating the next solution, simple mutation is ignoring information from the rest

of the population in setting the variables. This is why it is so disruptive in well-adapted individuals. The wider population may suggest that some values should be more 'mutable' than others. Thus, a refinement presents itself: sample solutions from the entire population, not just one solution with background sampling, i.e. allow the distribution of values across each variable to determine the future distribution. This should produce a more nuanced operation where the entire population biases the 'mutation' rather than a single individual.

An illustration is given in figure 3.4 where we show an example model derivation of this kind.

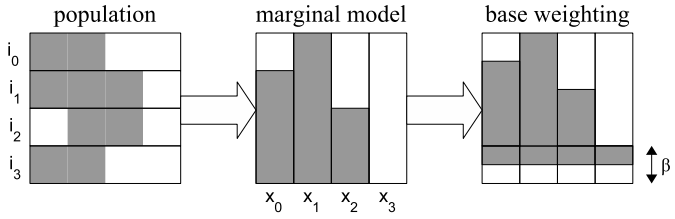

Figure 3.4: Population-biased sampling

This refinement of sampling to take in mutation results in a scheme which appears to operate in an equivalent way to uniform crossover with each solution in the population being a contributing parent. This perspective is considered next now we have subsumed mutation.

3.2.2 Recombination as Sampling

We now consider how crossover can also be represented as a straightforward sampling operation. For this, we use the uniform crossover operator which places no significance on the ordering of the variables but does associate values since the parentage is limited.

The action of uniform crossover on a pair of parents in terms of sampling distributions is shown in figure 3.5. This illustrates that, for the operations described,

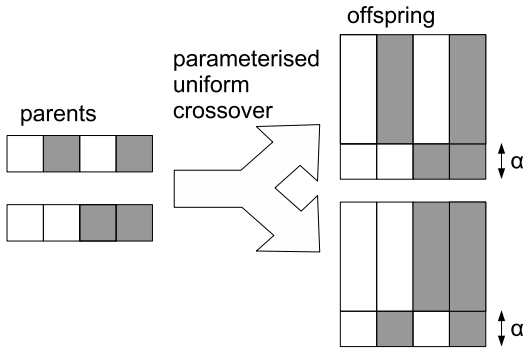

Figure 3.5: Crossover as sampling where α is the uniform crossover probability. For $\alpha = 0.5$ we have the standard crossover and a single distribution.

mutation is equivalent to low-level uniform crossover[2] with a random individual, i.e. when $\alpha = \beta$. Also, if we expand to n parents with $\alpha = 1/n$ then we are using the equivalent of a marginal model, i.e. we have completely dissociated the values. They would no longer have any implied association with their peers in the individual.

This form of crossover is less disruptive than mutation for a population with any degree of non-uniformity. However, the implicit genetic association that comes from $\alpha < 1/n$ is merely incidental. We want a more specific form of association that excludes incidental genetic association arising from limited parentage. We show how this might be achieved shortly, allowing us to benefit from the input of the whole population.

In fact, it is only the canonical GA which derives the search state purely from the population. Other, self-adaptive, algorithms augment the state with knowledge regarding the structure of the variable dependencies. This is in the form of a probabilistic model or a decomposed representation. To varying degrees, these use the population as a source of feedback regarding the problem, to be discarded as soon as the persistent model has been adapted.

In GAs, selection is usually performed at the level of individuals according to a

[2]An investigation into parameterised recombination is given by Spears [43].

probability distribution based on fitness[3]. Individuals are competing to persist in the population. In simple, univariate EDAs, selection is performed on single values, also competing to be expressed in the population. The difference between these is that genes only compete with other genes which share the same loci. This represents their niche. They are not concerned with other genes in the individual.

3.2.3 Defining Selection via Weighting

From both the approaches highlighted above we have drawn a common thread. This can be summarised as the probabilistic selection between weighted entities where selection is between entities from a mutually-exclusive set, e.g. individuals from a population, alleles at a locus, or settings for a partition.

The simplest weighting derived from the frequency statistics of a fit population would allow a search distribution to model the population. If there is a varying range of fitnesses, then the weighting can also be a product of this, i.e. the sum of fitness values of associated individuals rather than the frequency of those individuals.

The *weighting* itself can be either directly represented (as for the EDAs) or is a product of fitness and frequency (as in GA fitness-proportionate selection). The selective probability of an entity is its normalised weighting relative to its *competitors*, the mutually-exclusive entities.

3.3 Role of the Population

As noted earlier, the population generally has a combined role. First and foremost, it provides raw information regarding the composition of known likely optima in the form of a set of evaluated sample solutions. It represents in a subset of previously

[3]Fitness sharing techniques excepted.

evaluated individuals the entire state of the search, i.e. everything that is known about the problem. Since this set has been selected via fitness, we can infer the quality of genes and individuals by their presence and relative frequencies within it. All optimisation algorithms are rooted to the population in this respect (even if they maintain it only in a minor or transient way). Where algorithms differ is in how they use this feedback to adapt the search distribution.

A common secondary role extends the population with accumulated knowledge to adapt the search distribution. The simplest form of this is a particular feature of GA models where multiple instances of an individual are allowed to bias the weighting of that individual via increased frequency, as discussed in section 3.1.2. It acts via the genetic operators (such as crossover and mutation) as the source of new solutions. The resultant search distribution is a static function of the population distribution with some selection mapping.

The selection mapping can be deterministic, e.g. truncation selection, or stochastic, e.g. fitness-proportionate selection, ranked selection or tournament selection.

This linking of the roles via the genetic operators to merge the 'input' and 'output' of the iterative method is where the weakness of the GA lies. These hinder it in its goal of fulfilling the building-block process. We have shown how the important actions of crossover and mutation can be produced by sampling in a manner that equivalent to a type of individual selection.

No new information is obtained simply by allowing exact duplicates into the population. This could even be decremental if they displace other individuals with novel building-blocks. However, there is no reason why the frequency bias of multiple instances cannot be extracted into an explicit associated value assigned to an individual.

Under such a scheme, which marks a departure from the traditional GA model, our population would check for duplicates of newly-created individuals and only adds

them if none are found. If they are already present then the frequency value would be updated. This would eliminate the problem of duplicate individuals displacing informative but less fit individuals as we would be maintaining a set of genetic 'species' [4], rather than a population of individuals. By making individual frequencies explicit, we are extracting search state information from the population, which can retained purely as a selected sample of informative and unique solutions.

By assigning and maintaining a frequency value with unique individuals we are able to obtain at least two clear advantages:

Compact Representation An arbitrarily large population can be simulated, albeit with a bounded diversity. This means that we only need specify the *unique* individuals in the population rather than all individuals. Conversely, a minimum diversity of individuals can be ensured.

Evaluation Efficiency The required process of checking for uniqueness of a new individual gives us a straightforward means of caching fitness evaluations. This enables us to perform the frequency adaptation at a reduced cost in evaluations. We would use the frequency values wherever we need to select an individual within the population, e.g. in order to modify or remove it.

3.4 Population Compression Via Partitions

By associating a frequency value with each unique individual in the population, we can trivially produce a compressed but lossless model of the population. Combined with fitness information, this can be selectively processed as before. Repeated selection over this model would result in some individuals being amplified over others,

[4]The term 'ecosystem' might be appropriate for this set.

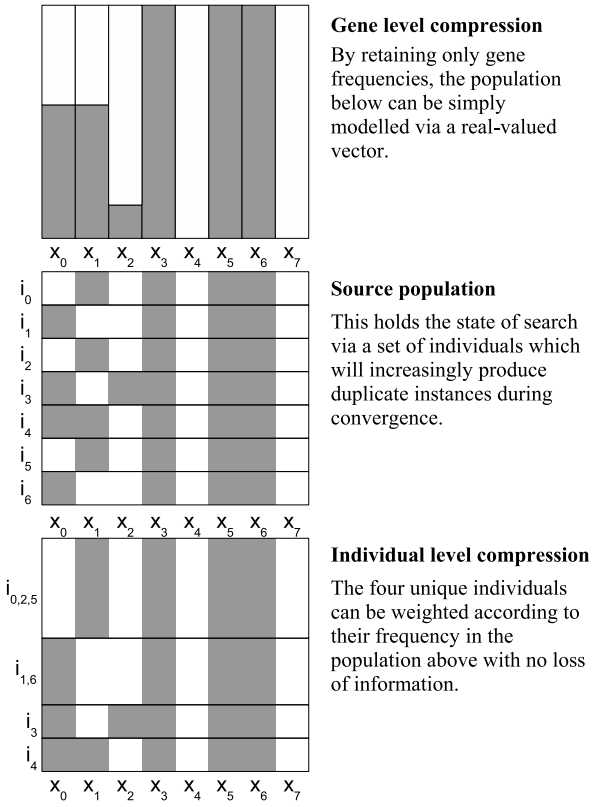

Figure 3.6: Extreme degrees of population compression. The source population in the centre can be compressed via individual frequencies below, or via genetic frequencies above. In the first case we have no loss of information whereas in the second we lose genetic association information.

eventually dominating the population, yet with no individuals being lost from the population – and none found. This only reinforces the interpretation of GA selection as simply a sampling of large entities, i.e. sets of genes rather than individual genes, as portrayed in figure 3.10. The bottom plot in figure 3.6 demonstrates this as minimal compression of the raw population of the middle plot. The top plot shows a compressed model where only frequency information for values is retained to give maximal compression.

The univariate EDAs assign a weighting to each value, if this simply reflects the frequency of the value in the population then we have the maximal compression illustrated in figure 3.6. The reason that such a statistical compression is not generally carried down to the level of variables is that we generally cannot discount dependencies between them. If we find we can impose independent partitions of the variables, i.e. they are separable, then we can perform a frequency-based compression on each one. The ECGA temporarily imposes such partitions between sets of variables which, according to its criteria, appear to be independent. This defines a model with an intermediate degree of compression, such as the example given in figure 3.7.

The univariate EDAs disregard any dependencies but if we partition the variables into sets, as ECGA does, then we have competitive selection within the niche of each partition.

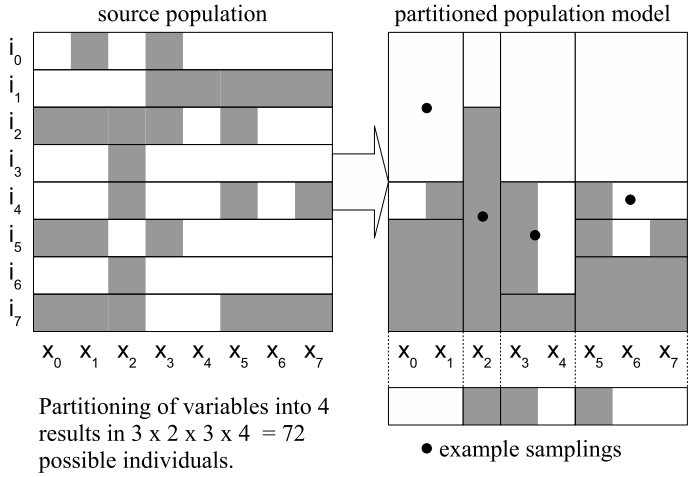

Figure 3.7: Partitional model sampling defines a non-zero search distribution over 72 individuals ($3 \times 2 \times 3 \times 4$) rather than $2^8 = 256$.

The imposition of any particular partitioning effectively determines a genetic linkage pattern. Each unique set of values then constitutes a selectable entity (or a module) with its frequency extracted from the population. Like alleles at a locus, their

mutual exclusivity implies they must compete for presence within the limited confines of the adapting population. The frequency weighting of any selectable entity can only increase at the expense of the others which overlap with it and vice versa.

The compositional algorithms assign no such weighting to the explicit modules they retain, referred to as *configurations* in the HGA, which are either discounted or weighted uniformly.

3.5 Competing Overlapping Modules

The previous figures illustrate the operation of selection on GAs and partitioning EDAs. A common constraint between these search models is that entities competing for selection specify identical sets of variables, be they individuals, partitions or alleles. We believe it is worthwhile to relax this constraint such that entities which merely overlap[5] can compete, as demonstrated in figure 3.8 with another comparative example in figure 3.10.

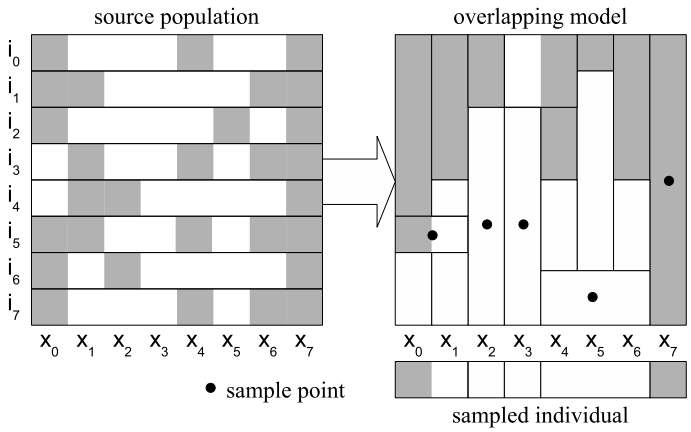

Figure 3.8: Overlapping sampling model.

This is a generalisation with significant consequences. It implies that such models

[5]That is, they specify values for a non-empty intersection of variables.

produce individuals via a variable number of samplings; the result of a particular sampling may determine whether other samplings are required. Specifically, if a module specifying several variables is sampled then those variables need not be sampled independently. Conversely, when a value is sampled then it precludes any modules being sampled which also specify that variable. The motivation for this is to enable competition between modules and their independent components, potentially performing the same function as the module validation phase of the compositional algorithms.

Some additional definitions are needed from us at this point. With the variables previously partitioned, we only had to define the relative weighting, or selective probability, for the set of mutually exclusive entities for each partition. Now, with a more complex distribution, we have the consider how the weighting is distributed.

We need to ensure that for any model extended with additional structure, the probability of sampling any of the genes concerned remains constant, that the difference is a merely structural one. This is important since the module is competing for multiple variables against its components. It is simply achieved by weighting modules proportionally to their genetic size. For the visual plot, this intuitively corresponds to the area of the selectable entity.

This module entails the explicit representation of a genetic structure intermediate with alleles and individuals. The outcome from this extension is that we can apply the same selective process to larger-scale genetic structures that we apply to individual alleles. This revision replaces the implicit genetic linkage imposed by positionally-biased operators and ordering, and restricted parentage with a unifying linkage scheme.

Finally, we need a source of candidate structures. Just as mutation substitutes new alleles in an individual, we propose a novel operator to perform structural mutations, producing new selectable entities from existing ones.

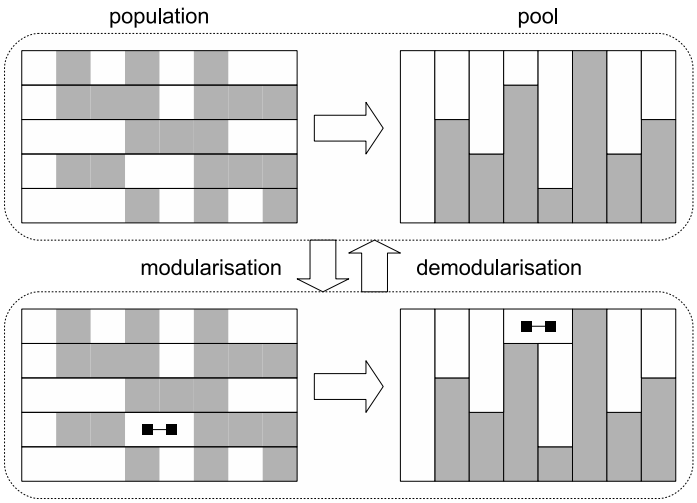

Figure 3.9: Example transmutation showing the reversible effect of modularisation and demodularisation on a population and the derived pool model.

3.5.1 Structural Mutation: Transmutation

The *transmutation* operation constructs and deconstructs modular structures within individuals. We distinguish it from the conventional mutation operator in that it modifies the genetic association between values in a solution, rather than the values themselves. An individual undergoing a transmutation operation evaluates identically but may become more or less adaptive. A sample transmutation operation is shown in figure 3.9.

This operator is similar to the join operation of the compositional algorithms but acts at the local level of the individual rather than a global population level. This means it does not completely replace its components in the pool but takes a proportional share of their selective probability. As such, it can be applied provisionally, on the basis of population instantiation.

Transmutation acts only on a single individual at a time. Selection then acts to propagate or eliminate the new entities in the population. Our optimistic starting point of maximal independence would place the initial model in the class of univariate

EDAs. By adding genetic linkage support in the form of modules, transmutation can move the pool out of this class via more complex joint distributions. Selection is then able to amplify this weak association if the module is competitive. If they do so, then they add adaptive complexity to the search distribution.

This operation creates the possibility for two individuals to arise which represent the same solution but with differing genetic structures. To avoid ambiguity, we use the term *phenotype* to refer to the information of the individual that contributes to its evaluation, i.e. the variable assignments. This distinguishes it from the changed representational structure that transmutation produces. Following transmutation, the phenotype is unchanged but the genotypic representation is modified. This means two individuals may be *phenotypic clones*, yet one may have a more adaptive representation.

The genetic structures produced by transmutation specify values for a subset of the variables that comprise a solution and, when present, set these in an individual. In addition, they may retain the hierarchical modular structure resulting from their historical groupings. This allows a demodularising transmutation to reverse a single modularisation transmutation rather than a series of them. By restricting possible modules to those actually instantiated in the population we are considering only modules for which we already have information. This may or may not reduce the space of possible modules but it does ensure we have feedback in a form with which we might differentiate modules.

3.6 The Pool As Search Model

Our population contains a selected set of individuals constructed from components of various possible scales. The method for generating new individuals is to construct

them from components sampled according to their weighting relative to their competing peers. We refer to this set of weighted components as the genetic *pool*.

The pool contains no extra information than is in the population. Rather, it maintains simple statistics from the population in order to produce new individuals. The resulting search distribution is then the probability distribution of individuals resulting from the pool model.

In the first instance, we might be aiming to model the population with perfect fidelity, i.e. to perform the minimal compression of figure 3.6. However, in order to explore we need to assume there exist some independence between the variables; that the individuals are, in fact, divisible. Where the population provides examples of fit individuals, the challenge of the model is to use these to adapt a search distribution which generates even fitter ones. This requires it to generalise the information in the population; we want new individuals which follow the underlying distribution rather than simply revisiting current population members.

The transitional pool is a more general model of the population than partitioning EDAs (in that it can represent overlapping modules) and more expressive than the compositional pool (in that it represents a range of weightings). It includes the selectional aspects of GAs, where weighting is set by the selectional probability of a specific selection scheme.

The key advantage is that, not only is this structure more expressive but it allows us to naturally go from a simple model to a more complex one in a continuous way. This raises the possibility of an approach which does not require the particular model to be defined but can adapt to an appropriate level of complexity automatically. An implementable description of this pool model is given in chapter 4.

3.7 Component Weighting

We have outlined a general, frequency-based representation of a population which can store, and be sampled for, an increased number of individuals without loss of information or capability. We now address the weighting of components.

Individuals should not be allowed to contain overlapping modules even if these concur on values. This is necessary to ensure competition. For the same reason, the weighting of a module should be proportional to its genetic size. This would ensure that its selective probability is balanced against all the smaller components that would exclude it. The smallest modules, i.e. genes, need special consideration for their weighting, due to the following situations:

Initial Sampling Before we have a population sample, we have no information with which to weight the primitive modules, i.e. genes. A uniform weighting seems obvious but a non-zero ongoing weighting is also suggested below.

Background Sampling If the population is particularly small or otherwise lacking diversity for any variable then it may not contain a particular gene (the initial selection can be considered an extreme case of this). In this case, if the weighting is a product of the frequency, then the gene has no way to be reintroduced into the population. If this gene is required for an optimal solution then the search would become permanently trapped at a sub-optimal distribution.

The standard GA operator to avoid this type of situation is mutation but we have already shown how this same effect can be produced. By introducing a suitably low 'base weighting'[6] for genes we can ensure that the search can avoid being permanently constrained. The entire search space is raised above a zero probability. This also

[6]Shown as an even proportion of β in figure 3.3

implicitly addresses the issue of the initial search distribution before any individuals have been generated.

3.8 Summary

This chapter provides an analysis explore the common ground between individual selection in population-based models and the sampling of EDA models.

3.8.1 Unification of Model Operations via Sampling

A visual notation of weighted, structured probability models is introduced and used as a unifying device to encompass key operations from GA and probabilistic models. The weighting of components at all levels allows search to be a product of the entire population rather than just one or two individuals. A comparative overview is shown in figure 3.10. In particular, we isolate the linkage-biasing aspects of GA search from the frequency-biasing process to be supported, as below.

3.8.2 Extending Selection to Scaled Genetic Structures

By embedding explicit modular structures within solutions, we support a selection process *within* individuals in an exploratory role and not just *on* them in an amplifying role. This combines the compositional and the probabilistic approaches in a useful and straightforward way. This means we can adapt the composition of the pool model incrementally by making intersecting modules of various order compete with each other for selection in a evolutionary manner. This may give us adaptive complexity via an open-ended progression from simpler models to ones with more genetic structure when it confers an adaptive advantage.

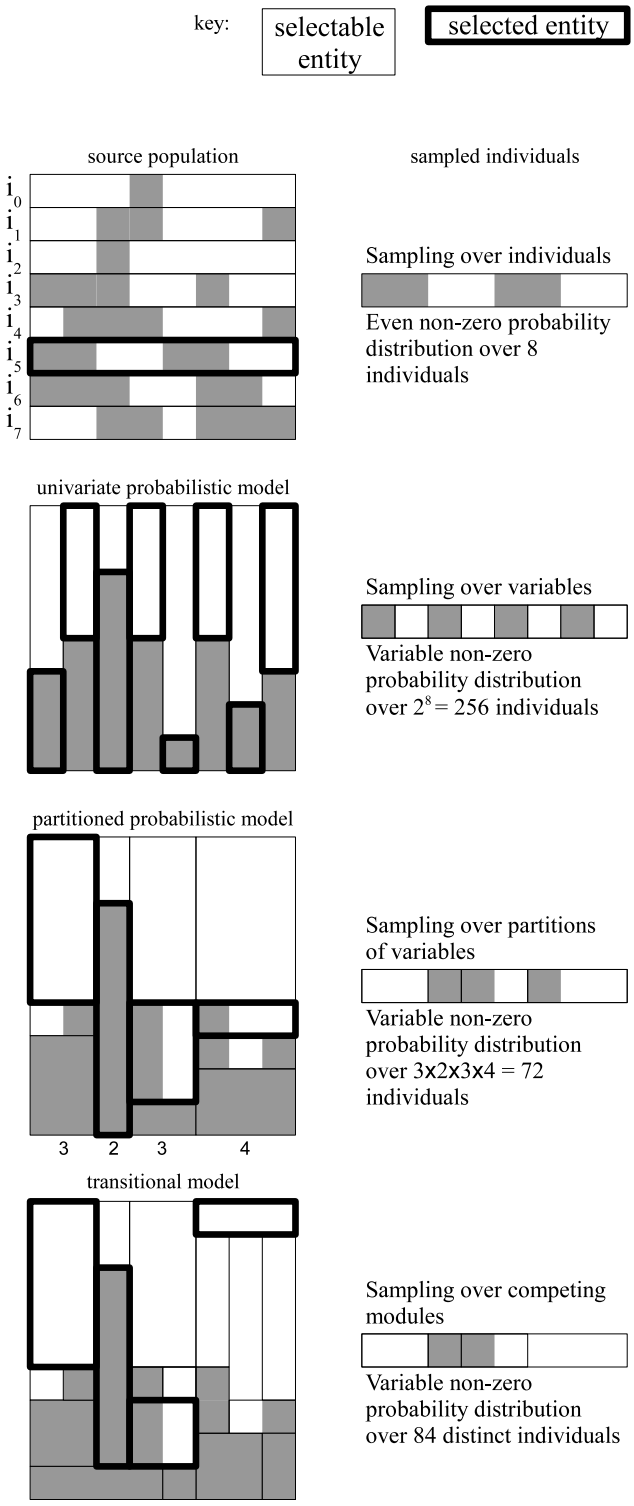

Figure 3.10: Comparison of different search models of example population. Note that module variables are adjacent for graphical convenience.

Chapter 4

The Transitional Evolutionary Algorithm (TEA)

In this chapter we bring together principles harvested from the earlier analysis and set them in an implementable algorithm. This will allow us to empirically validate them and highlight any issues which may have been overlooked up to this point. We now define the methods and structures of TEA starting with an overview before moving in to describe them in more detail.

4.1 Overview of Algorithm

The adaptive cycle of the algorithm is outlined in figure 4.1. A population of elite individuals is maintained, with individuals only being discarded when their fitness no longer qualifies them. New individuals then either expand the population or, if they are fitter, reduce the population to include only themselves.

Individuals are constructed by sampling components from a pool which maintains records of their frequency in the population. Transmutation is randomly applied

to individuals to generate provisional genetic linkage. The resultant modules are maintained in the pool as new components and are subject to selection acting on their containing individuals.

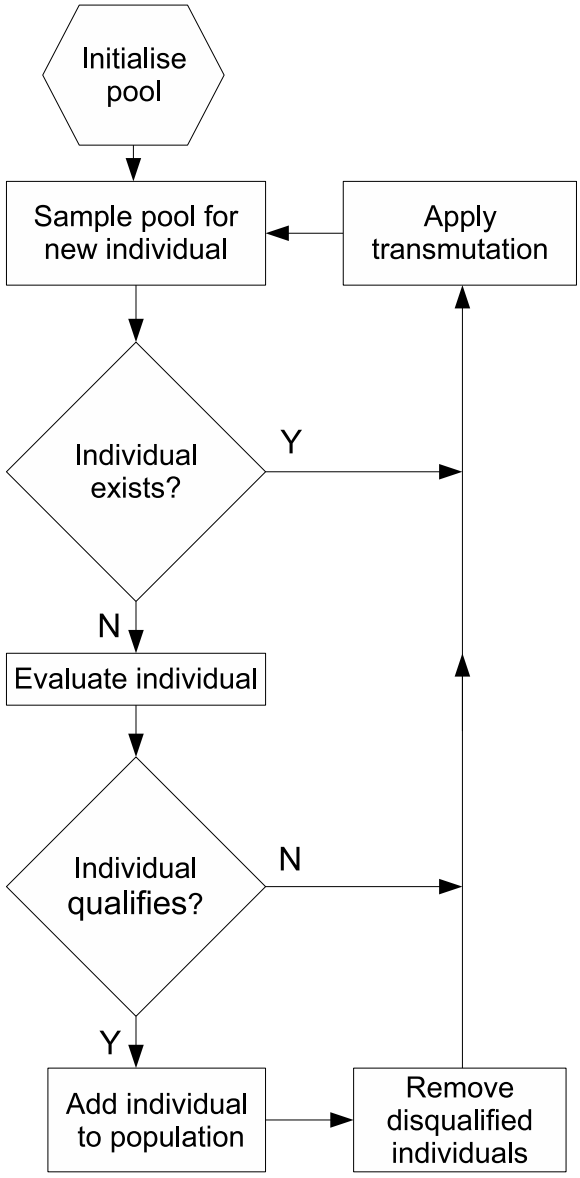

Figure 4.1: Overview of TEA process.

4.2 TEA Model Structure

The search model has multiple levels of structure. At the highest, we have a set of unique solutions, i.e. phenotypes, which qualify by virtue of their fitness.

If we refer to any particular module, allele or individual as m then we denote the fitness as $f(m)$, the frequency as $g(m)$, the number of genes $|m|$ and the weighting as $\omega(m)$.

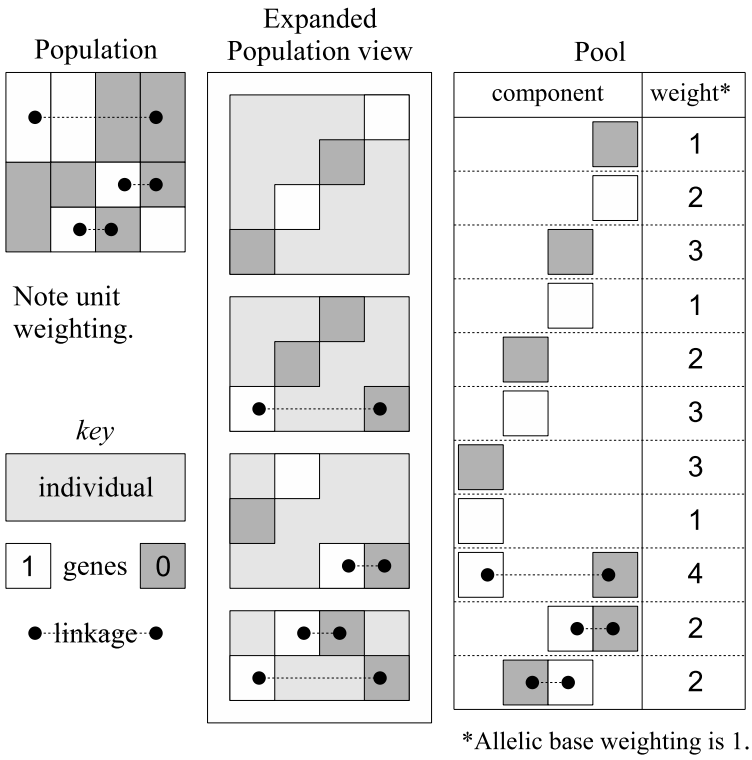

Figure 4.2: Sample TEA model.

The simplest components, i.e. alleles, accumulate weighting from their containing individuals on top of a fixed base weighting, as shown in figure 4.2. Composite components do not benefit from this, relying entirely on their embodiment in the population for weighting. Not shown in figure 4.2 are hierarchical modules resulting from the grouping of composite modules.

4.3 Structural Definitions

The definition of a gene as a value assignment to a variable is straightforward. It represents the simplest genetic structure and, for our purposes, the most primitive module. This means that only one instance of each possible gene is held in the pool but associated with a frequency value and weighting.

Multiple genes may be grouped, via transmutation, to form a module. This module may contain other modules recursively but clearly cannot contain itself, directly or indirectly.

Any pair of modules are identical if they contain an identical set of components[1] or they are identical genes. Two individuals are *phenotypic clones* if they contain within their structures an identical set of genes.

The different levels of structure are characterised by the constraints on them.

- A gene specifies exactly one value for a single variable.
- Modules may contain a mixture of genes and other modules with the constraint that no genes are contained which specify the same variable.
- An individual contains within its modular structure exactly gene for every problem variable.
- A population is a set of individuals with the constraint that no two specify the same values for all the variables, i.e. each is a unique solution.

All the structures above have an associated frequency and weighting, except for the population which we limit to a single, unweighted instance. The definition of this structure gives us a means to encode genetic linkage and provides us with the scaffolding for a feasible building-block process.

[1]The recursive definition means only the top level of components need be checked.

4.4 Pool Initialisation and Sampling

The principle for generating a new individual in TEA is to construct it from components sampled from the current population. Components may be values or groups of values as above. However, no two components may specify the same variable, even if they agree on the value.

The sampling pool refers to all the component modules from which new individuals may be constructed. This auxiliary structure means we only need to store a single instance of any module from the population which can be referenced by the embodying individuals whose frequency we can associate with it. This makes it a significant algorithmic optimisation.

The process for generating a new individual takes the form of a repeated selection from a set of applicable components. Upon each selection, the selected component is added to the individual and its competing (i.e. overlapping) components are removed from the set. This continues until the new individual is fully specified with a value for each of its variables. The process is described algorithmically in figure 4.3.

The selection of a component from a shrinking set of candidates occurs with a probability proportional to each module's relative weighting (given by $\omega(m)$), i.e. for a pool of modules M, the probability of any specific module m being selected is $p(m)$ where:

$$p(m) = \frac{\omega(m)}{\sum_{m\prime \in M} \omega(m\prime)} \tag{4.1}$$

This implies that, since components that specify common variables are mutually exclusive in an individual, the selective probability of a module is determined by its weighting relative to its competitors. This means modules can compete with other modules of the same and higher order in an equivalent manner.

```
i ← fully unspecified individual
c ← set of applicable components from pool
while i has unspecified variables
   sample module m from c proportional to ω(m)
   add m to i
   remove all components from c which intersect m
end while
return i
```

Figure 4.3: Constructing an individual.

Having defined the structures used in TEA we now define how we determine the weightings of these modules. In general, the weightings are assigned according to the frequency of the components in the population and proportionally to their genetic size. The primitive genes additionally have a base weighting β, required to initialise an empty population and avoid suboptimal fixation. The weighting is therefore:

$$\omega(m) = \begin{cases} g(m) + \beta & \text{iff } |m| = 1, \\ g(m)|m| & \text{otherwise.} \end{cases} \tag{4.2}$$

4.5 Transmutation Operation

The origin of new composite modules is the transmutation operation. The structure the individual retains in addition to the set of genes is adapted via the two possible transmutation operations of modularising and demodularising. Each is the inverse operation of the other, as shown in figure 3.9.

Having selected an individual we still have a large number of possible modules arising from the available components. Our first heuristic here is to modularise fewer components. Aiming to embody high order building-blocks in a single step is far harder and may be unnecessary if building blocks can be constructed incrementally. This situation is analogous to mutation; a greater mutation rate may be able to

```
i ← source individual
if uniform_random(0, 1) < ψ then
   modularise(i)
else
   demodularise(i)

modularise(i) :
   if i contains less than k components return i
   m ← k random components from i
   remove components of m from i
   add m to i
   return i

demodularise(i) :
   if i contains no composite modules return i
   m ← k random composite module from i
   add components of m to i
   remove m from i
   return i
```

Figure 4.4: Transmutation pseudocode. ψ is modularisation bias, set to 0.5

take larger steps through the search space but the probability of it being a beneficial adaptation shrinks accordingly.

The modularise operation groups a number of existing components into a new component. The demodularise operation selects a modular component at random and decomposes it to restore its components as selectable individual components. The modularisation bias ψ determines the tendency of the model to increase in structural complexity.

In adding a module to the pool we are potentially increasing the structural complexity of the model. Whichever operation is performed, the pool statistics need to be updated as component frequencies are either incremented or decremented as below.

If this is a modularisation operation then this decrements the frequency of the components in the pool and increases the frequency of the module. A new reference

for this module is created if necessary. The weighting of the components attained from the individual is redistributed to the module.

If this is a demodularisation operation then the inverse is performed. The module is decremented and its components reinstated in the individual with a redistribution of the share of the weighting from the module. This is why the structure of modules is retained in the individual.

4.6 Population Maintenance

Before the new individual is evaluated, its phenotype is compared to the other individuals in the population. If there is already an individual which corresponds to that phenotype then the individual is discarded.

Our population comprises a variable number of individuals. Each of these is unique. A separate record of the weighting of each individual is kept. This set of individuals is a subset of all the individuals generated thus far. Members are added (and persist) based on their fitness. New members are included if they meet a qualifying criterion and existing ones are removed if they do not. We conservatively set the criterion to be for individuals to match the current maximum fitness in our initial definition. Later on, we examine how and why this can be extended.

If an individual is removed then all its components are decremented in the pool.

4.7 Related Algorithms and Concepts

The algorithm to be presented has features which both associate and distinguish it relative to other algorithms. A brief summary follows.

4.7.1 Solution Generation

The process by which new solutions are constructed in TEA, via sampling of components, is akin to the compositional and the probabilistic approaches as discussed earlier.

TEA components are similar to the GA schema, SEAM modules and HGA module settings in that they specify values for variables. They extend this with an explicit weighting value to bias selection. This is one way that TEA aims to advance over the compositional algorithms.

Other probabilistic modelling algorithms employ a variable weighting derived from a population. However, this weighting is extracted from a particular structural model of the population, e.g. a univariate or partitional model. This structure may constrain the form of distributions that can be produced, particularly if the population is a complex distribution. The use of the population is discussed further now.

4.7.2 Population Use

The TEA shares the frugal spirit of HGA with regard to the population in that individuals are retained while they still inform the algorithm on a good search distribution. However, whilst the HGA (along with the GA and most other population-based algorithms) use a fixed-size population, the TEA allows the size of the population to vary as an effect of the qualification criterion. This defines a strict form of elitism. Also, the population members are immutable; there is no recombination or mutation performed on the values of an individual.

Probabilistic models generally discard the population at each iteration, possibly generating (and evaluating) an individual multiple times. GA models may employ a degree of elitism, up to a steady-state level, but also expect to discard and recreate

individuals.

4.7.3 Model Representation and adaptation

Both the ECGA and the HGA partition the solution into non-overlapping sets of variables before extracting or adapting the selective probability of the modules within these. The TEA supports these and also supports overlapping modules; in fact, it requires them in order to allow modules to confirm their viability by competing with their components. The frequencies are taken from the population in which they are embodied. This embodiment of multi-level dependencies makes the population a more expressive model than both the ECGA and the HGA, as figure 3.10 illustrates.

Structural adaptation, i.e. transmutation, in the TEA differs from the join operation of the compositional algorithms is various key respects. It acts locally, specific to an individual, where the join operation is a global operation. Transmutation is provisional with a minor effect and reversible where the compositional join is permanent and with a major effect on the search space.

Whilst HGA and SEAM consider candidate modules in an arbitrary order or from random selection respectively, TEA uses the relative frequencies of components and their correlation in the population to bias their generation probability.

This formulation is expected to solve several of the issues of the compositional approach in a simple and evolutionary-inspired manner. The testing phase is essentially ongoing as modules compete against their components for the niche of the variables. There is no shortage of 'substrate' since components are not removed from the pool on production of a module. Actual modules are those that persist and thrive over time giving an increasing degree of confidence.

4.7.4 Schema Theorem and Building-Block Hypothesis

The essence of TEA follows the motivation of the original building-block hypothesis: the incremental construction and reuse of functional units for solutions of increasing adaptive complexity. However, the recombinative approach hangs on an appropriate encoding and a careful balance of the forces of selection and recombination. The transitional approach seeks to moot these issues by manifesting possible building-blocks as explicit entities under the influence on evolutionary selection. This means these can be processed to avoid disruption without requiring a predefined 'tight-encoding'.

4.8 Summary

In this chapter we have defined the novel Transitional Evolutionary Algorithm (TEA) to an implementable level of detail. This enshrines the principles put forward in the development chapter. Where design decisions have been made, this has been noted and justification has been given for the path taken.

Following this, the detail of the algorithm is also compared and contrasted with related algorithms and concepts.

Chapter 5

Experimental Study

We intend to validate the algorithm arising from the original hypotheses via a targeted empirical examination of its various processes which not only compare the external behaviour and performance of the TEA to related algorithms, but that also probe its internal processes. These experiments are expected to be valuable in explaining the differences in performance between complete algorithms competing head-to-head.

5.1 Overview

The overall strategy for validating the hypothesised method is an incremental one. We introduce transmutation and structured individuals only as we reach the limits of an algorithm without. The use of a population-biased search model with a specialised population maintenance strategy is validated first. We refer to this as Population Biased Sampling (PBS) and it can simply be thought of as TEA with transmutation deactivated. This is intended to bring to light any theoretical flaws or practical difficulties at the appropriate place. With this is mind, the key facets of PBS/TEA are broken down into the following broad areas, related to underlying themes of this

thesis.

5.1.1 adaptation from Population-Biased Sampling

To examine the principle of sampling structures from a population maintained under specific criteria, we employ a general form for a linear problem as described in section 1.4. At this level, with no variable interactions to accommodate, this should be in the realm of the univariate EDAs identified in section 2.3.1. We therefore take the opportunity to obtain a benchmark comparison from these, casting them into a unifying form to support a strong and fair comparison. This also includes a Random Mutation Hill Climber (RMHC) and PBS.

We go on to test the principle of using an adaptively-sized population of elite individuals rather than a population of a fixed size to set the sampling distribution.

5.1.2 Maintaining an Informative Population

A class of problem with variable interactions is defined and examined. We explore the limits for a search that does not support genetic linkage. We follow this with a consideration of the conditions for successful linkage identification, namely a diverse distribution of building-blocks in the population. This is rooted in our discussion in section 2.5.1 where we considered the role of the population.

5.1.3 Structural Selection

Having verified we can maintain a population which is sufficiently informative of the problem optima structure, we extend our search model with genetic linkage to support that structure. Transmutation generates candidate modules and thus provides the mechanism for an adaptive representation as mooted in section 3.5.1. This allows us

to test our hypothesis that modules that overlap but do not cover an identical set of variables can compete in the same way as alleles above. This is crucial if building-blocks are to be identified and amplified by the same selectional process as we propose in section 3.5.

5.2 adaptation from Population-Biased Sampling

A scalable algorithm is one that is competitive on a range of problems, including the simplest ones. Although non-taxing for existing algorithms they can be useful in supporting some of the mechanisms which we will rely upon later. In particular, we are maintaining the population solely as a representative sample of optima and deriving from it a weighted model via frequency only, with no linkage bias.

For problems with no interactions between variables (and therefore no dependencies to be discovered) it is only optimal values that need to be identified and not structure. Bearing in mind that such problems can be solved in linear time[1], we evaluate the performance of population sampling against a background of the most significant models for this kind of operation. This includes the univariate models from section 2.3.1.

We investigate scalability in terms of the mean number of evaluations required to find the optima for problems of increasing size. The outcome of these experiments is an insight into, and verification of, the underlying sampling process that the TEA uses, obtained within the context of a range of related algorithms. This lays the foundation for the high-level dynamics of modelling dependencies.

[1]If the variables are truly independent then they can be optimised independently via enumeration. For instance, binary encoded problems of l independent variables can be optimised in $l+1$ evaluations simply by checking the alternative value for each variable in turn against a base solution.

5.2.1 The Linear Problem

This testing requires a problem for which any solution x has an optimal value for each of its n variables x_i, independent of the other variables. The general form for such a function we take as:

$$f(x) = \sum_{i=1}^{l} f^i(x) \tag{5.1}$$

where $f^i(x)$ is a simple mapping via the variable x_i in x to a fitness contribution value. An instance for the well-known *onemax* problem[2] where:

$$f^i(x) = \begin{cases} 1 \text{ if } x_i = 1, \\ 0 \text{ otherwise} \end{cases} \quad \text{or } f^i(x_i) = x_i$$

This sum-of-functions form is used since it is easily extensible to the more complex problems used later. The univariate form of the nk problem [21] (i.e. where $k = 0$) is simply defined with $f^i(x_i)$ assigned from a uniformly random distribution between 0 and 1. Although the difference in fitness contribution is less than *onemax* (and therefore could take longer to distinguish) the structure is identical. For conventional simplicity, we use *onemax* for this round of experimental validation.

5.2.2 Univariate Algorithm Comparison

A particular role for probabilistic models is to adapt the relative frequencies of competing entities, whether they are individuals (e.g. GAs), alleles (e.g. univariate EDAs) or modules (e.g. ECGA). These stochastic approaches represent the convergence of each variable either implicitly by its relative frequency in a population (as per the

[2]By the conventional definition:

$$f^{onemax}(x) = \sum_{i=1}^{n} x_i$$

Model	Population	Selection	Sampling Vector
BSC	n generational	fitness prop.	models fitness distribution
UMDA	n generational	tournament	frequency modelling
PBIL	n generational	truncation	adapt frequency modelling
cGA	$n = 2$ generational	none	per-locus adaptation
RMHC	$n = 1$ steady-state	elitism	individual modelling bias
EPBS	n adaptive, steady-state	adaptive elite	models pop. distribution

Table 5.1: Univariate modelling algorithms compared.

GA approach) or by an explicit probabilistic distribution (as per the EDA approach).

The presentation of the univariate modelling EDAs in table 5.1 is intended to emphasise their commonalities. They all generate new individuals via the sampling of a probability vector (of length l) which is, in turn, defined from a population of n prior individuals. By casting them into a common form, we strengthen the value of a comparison between them. Having specific aspects of TEA expressed in the same format allows us to directly infer the effect of these on performance for this restricted case. We refer to this reduced form of the TEA as Elitist Population-Biased Sampling (EPBS).

We describe and interpret all these experiments using a fixed base weighting we denoted β which, relative to the weighting vector ω, defines the search distribution. Defining RMHC in these terms clarifies our usage of it. An analysis of the results is shown in figure 5.1 with the leading algorithms shown in greater detail in figure 5.2. Predictably, the more recent algorithms show generally improved performance over the earlier ones, the exception being the RMHC algorithm.

RMHC Random Mutation Hill Climber is an individual-based stochastic search [11], equivalent to an ES type approach. Mutation can be fixed at exactly 1 bit or performed on a per-locus basis with a probability of $1/l$ to give an expected mutation rate of 1 bit from a binomial distribution. The current solution is replaced in the

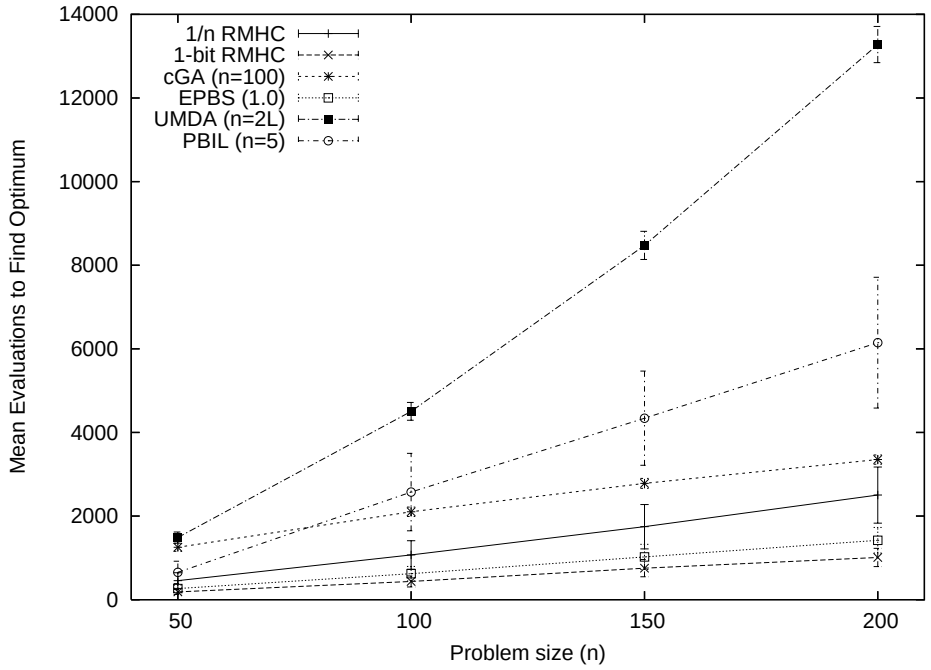

Figure 5.1: Scalability of univariate models (with parameters) on onemax problem. Averaged over 100 trials with standard deviation shown by error bars.

offspring if as good. This becomes significant later where the problem structure requires neutral drift. The weighting vector ω is simply set by the current individual which represents the entire search state, i.e. $\omega_i = x_i$. The inclusion of the base weighting simulates a mutation operator[3] with a per-locus probability of β to give our selective probabilities for the next individual, x':

$$p(x'_i = 1) = \beta/2 + \omega_i(1 - \beta) \quad \text{and} \quad p(x'_i = 0) = 1 - p(x'_i = 1)$$

By further defining the replacement rule for the current individual, e.g. (1+1) or (1, 1) we fully emulate an ES-style search. We use this as a baseline from which to advance our ideas about a better-informed definition of ω than above.

[3]Random substitution rather than bit-flip.

BSC We derive a probability vector P from a population of solutions weighted by fitness [45]. This is sampled to generate the replacement population of constant size.

Without mutation (β is effectively zero) this model easily becomes trapped as the population converges. Trial and error allows a minimum population size to be determined which reliably finds the optima. Unfortunately, the population tends to be several times greater than the problem size and, since this is reevaluated for each generation, it takes many more evaluations than the other algorithms tested. It cannot usefully be put on the same scale as the others.

UMDA This selects a replacement population via tournament selection from which it then derives a probability vector P [33]. This algorithm also requires a large population to avoid convergence to a sub-optimal solution. However, its stronger selective pressure allows it to converge faster, as shown.

PBIL This samples n population vectors (i.e. binary strings) from a probability vector P before adjusting the distribution toward an elite subset [1] of $M = 1$ vectors by the rate $\alpha = 0.15$. Mutation occurs with a probability of $1/n$ and a step size of 0.05 and is essential for larger problem sizes to avoid homogeneity. This slows it down but ensures it does not get stuck. It scales badly but this may be mitigated by its resistance to getting trapped in local optima.

cGA This samples two solutions from its probability vector then compares each corresponding variable [16]. If the values are different then the weighting for that variable is shifted by a fixed amount $1/n$ toward the fitter. The single parameter n represents the simulated population size of the model. There are no clear heuristics how to do this aside from trial-and-error[4]. Setting this value too high results in an

[4]A method for automating this trial-and-error process has been previously presented [27].

overly-slow rate of adaptation. Setting it too low results in the optimum not being found as hitch-hiking non-optimal values are rapidly reinforced. A good compromise was found to set $n = 100$.

EPBS This draws from TEA the features of interest at this level of testing. It maintains a steady-state population of unique solutions which evaluate to the highest current fitness and represent the current known optima. The weighting ω is simply assigned the frequency of instantiating individuals, i.e. if we define a function m of a value v at a locus i for a chromosome x such that:

$$m(x, i, v) = \begin{cases} 1 \text{ if } x_i = v \\ 0 \text{ otherwise} \end{cases}$$

then the weighting vector ω is simply:

$$\omega_i^v = \sum_{x \in P} m(x, i, v) \tag{5.2}$$

This is in line the RMHC formulation given above where ω^1 was used implicitly. Since the base weighting β for EPBS is also set at $1/l$ it can be seen that if we restricted the population to a single member then this is equivalent to the RMHC. Therefore EPBS can be seen as a logical extension of ES.

Other Algorithms The canonical GA was also applied to this problem, with fitness proportionate selection, uniform recombination and a mutation rate of $1/n$ (consistent with algorithms above). It performed too poorly to be included on the same scale as the above for this problem. Once the population converges, mutation becomes the only effective operator.

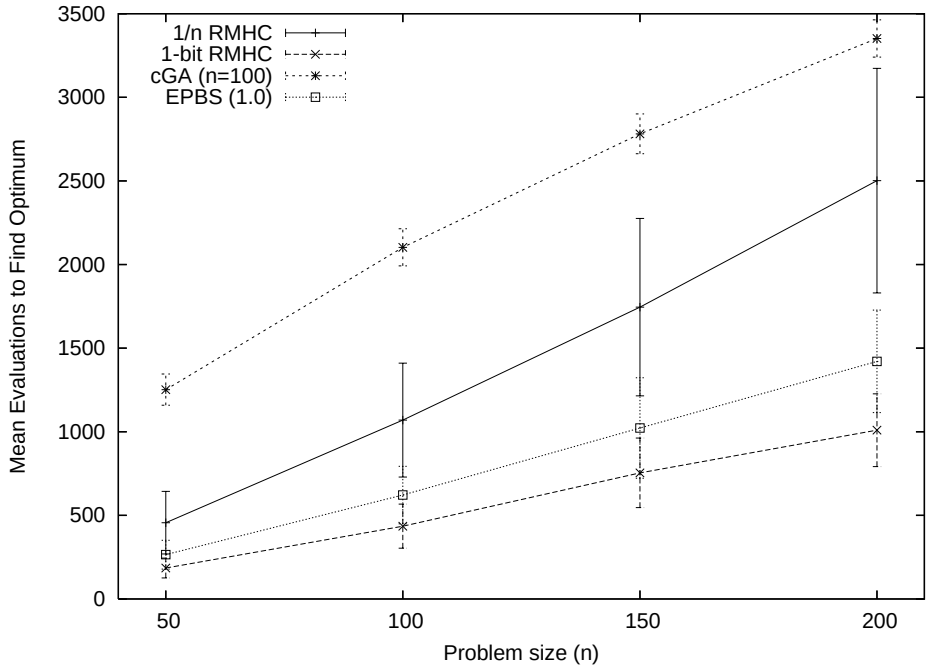

Figure 5.2: Leading univariate models compared on onemax problem. Averaged over 100 trials.

Interpreting RMHC gain over univariate EDAs

Even when the RMHC can be framed as a simple univariate EDA it outperforms more sophisticated ones on the simplest linear problem. We suggest a combination of reasons for this:

Adaptive Period There is a potential for the search state to be adapted after each evaluation when it is represented by a single individual. This contrasts with the generational approach (assuming no elitism) which only adapts after n evaluations.

Bias Strength The search distribution from mutation is generally far narrower than that resulting from a population model, i.e. the range of successive solutions is smaller. This is particularly true earlier on in the run when a population is far from convergence.

Non-Converging Distribution Relying purely on a model of the population inevitably leads to convergence via genetic drift. Either a larger population (to postpone this) or mutation (to counter it) have been offered as remedies [46]. RMHC, having a fixed variance, is immune to this problem, although it may still terminate in a local optima on multi-modal landscapes.

This last point highlights an advantage for a population-based search model highlighted in section 2.5.1: to obtain more information than a single individual can provide. This can include more nuanced sampling probabilities and, later, information regarding possible dependencies.

EPBS gain over RMHC

There are several factors which are likely to have produced the gain of EPBS over RMHC, as well as the other univariate-modelling algorithms.

Firstly, it uses the bias from an elite set of samples rather than a single sample or set of samples with varying fitness. Moreover, this set is not discarded at each generation but is reused and potentially adapted with each new sampling. This produces significant gains, even before we start comparing new solutions to current ones. Finally, the population maintains diversity via the base weighting with uniqueness checking[5]. This allows alleles to compete for representation if there are distinct individuals with equal, or at least comparable, fitness. We aim to produce a similar dynamic for modules later.

The uniqueness checking might be considered an unfair advantage that EPBS has over the other algorithms. In fact, tests show the gain is relatively minor for

[5]This does not imply it cannot become effectively trapped in local optima; we consider problems with such structure shortly.

the additional computation. As we aim to demonstrate later, the real advantage of the uniqueness checking, is in providing a distribution which consistently provides information on epistatic dependencies — information that cannot be exploited by univariate population models.

5.2.3 Comparing Population Maintenance Strategies

We now examine whether there is an optimal population size for PBS on this class of problem, or whether the adaptive-sizing approach described in section 4.6 performs better. This is significant since an adaptively-sized population is a characteristic not shared by many population-based algorithms. The restriction to unique individuals is also implicitly under scrutiny here. We might expect a population with a fixed size to contain more information than one which might frequently reduce to a single individual. However, we might also expect the influence of less fit individuals to slow down the rate of adaptation.

Figure 5.3 shows the general effect of a fixed population size compared to an adaptively-sized population for the *onemax* problem over a wide range of problem sizes. Higher fixed population sizes do not show a significant difference with the result for 2 and 3. EPBS does indicate a gain over these but with a degree of variance overlap that makes a strong result impossible to draw.

Nevertheless, it indicates that an algorithm that supports an adaptively-sized population may have an edge over fixed sizes, with a fixed population size of 2 being closest. In any case, we show later a compelling case of why a strictly elitist population strategy will prove inadequate for more taxing problem forms.

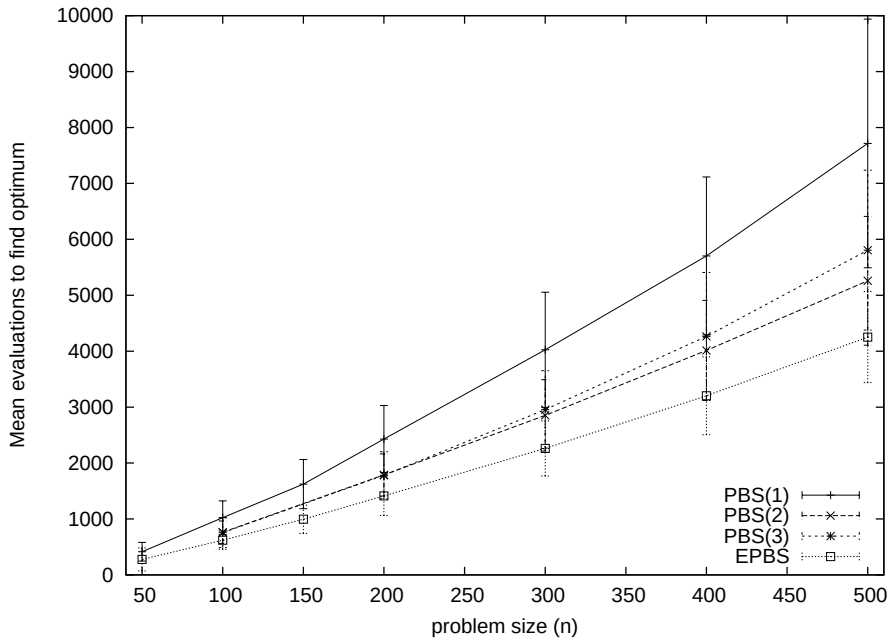

Figure 5.3: Comparison of static with adaptive population sizes. Averaged over 100 trials.

5.2.4 Summary for Univariate Population Modelling

The principle of using a vector model of a population has been shown to be poorly supported by a range of univariate EDA models on *onemax*; they are reliably outperformed by a RMHC algorithm. Of the algorithms tested, only CGA and EPBS were competitive with RMHC. This is shown in more detail in figure 5.2.

It may be that these algorithms are more successful on problems with a small degree of structure, e.g. with a small number of pair-wise dependencies. However, since there is no apparent representational feature for maintaining — let alone validating — genetic linkage within these algorithms, we depart from them at this point.

We have demonstrated the general principle that a population-biased probabilistic model can outperform an individual-biased RMHC[6] via an elite population of unique solutions. This is reasonable if PBS is viewed as a RMHC with a population size

[6] $1/n$ rather than 1-bit RMHC

greater than one. The gain shows some indication of increase if we allow the population size to be set purely by the current set of qualifying individuals, although this is not clear-cut.

This supports our approach to maintaining the population for the simplest class of combinatorial problem. The goal here was not to devise an optimal algorithm for linear problems but to show how a probabilistic model of a population may be most effectively integrated with the processing of that population. In order to gain scalability, we now need to generalise our model to perform on more difficult problems, i.e. those with interactions between the variables.

5.3 Maintaining an Informative Population

With the straightforward sampling of an elite population having proved efficient for linear problems, we now expand our problem class to include those with interactions between the variables (the modular problem class from section 1.4) and examine the implications for the workings of our algorithm.

5.3.1 Representing Fitness Interactions

A *fitness interaction* [49] exists between variables when their resulting fitness contribution cannot be expressed as an additive function (as in equation 5.1). Nevertheless, the problem remains linear between modules despite the combinatorial search within them. This means we can simply generalise the earlier additive fitness function to additionally include specific subsets of the variables in x (the 'chromosome'):

$$f(x) = \sum_{i=1}^{|I|} f^i(x) \tag{5.3}$$

Where i indexes each interaction from a set I which we refer to it as the *interaction network*, in line with biological models [25]. The interaction network comprises the fitness contribution not only for each genetic value, as before, but additionally for specific sets of genes. The additive effect of a specific interaction — the fitness contribution — is applied when specific genes co-occur. We denote these interacting genes as v^i and their corresponding fitness contribution as e^i. The interaction function is then defined by conditionally applying the effect e^i if, and only if, the genome x contains all the values of v^i.

$$f^i(x) = \begin{cases} e^i \text{ iff } x \text{ concurs with all values specified in } v^i, \\ 0 \text{ otherwise} \end{cases} \tag{5.4}$$

A building-block from this viewpoint is then an interaction set with a significant positive fitness effect.

The definition of the interaction network determines the character of the problem. The simplest interaction network we can define is a pairwise one, of which we may specify up to $n/2$ interactions without overlapping, a constraint we apply at this stage. The problem with n variables partitions them into a set I of $n/2$ interactions and evaluates the fitness as the sum over these. Figure 5.4 includes sample interactions taken from standard pairwise sub-functions in a form used by Watson [49].

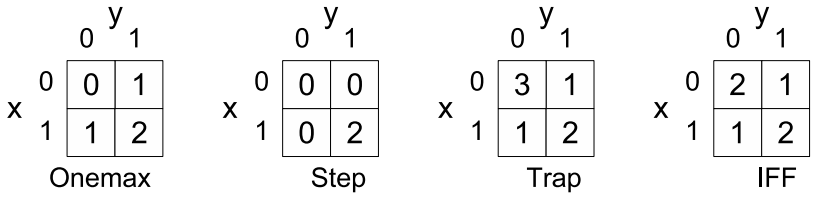

Figure 5.4: Pairwise dependencies for binary variables x and y.

The *onemax* and step functions are qualitatively different from the trap and IFF functions in that they have only a single optima. *Onemax* has a 'gradient' of possible

adaptations to this optima whilst the step function requires the optima to be found at a stroke. The trap function is deceptive in that the 'gradient' leads away from the global optima toward the local optima, making it as least as taxing as the step function. Finally, IFF has two global optima to be found. If the hierarchical form is to be solved then these must both be found and also retained for further combining.

If the interaction reinforces the optimal values then the problem difficulty is unchanged. The problem becomes more difficult where an interaction alters the relative ranking in terms of the fitness of the solutions in the search space. It may do this by changing the global optima or revising local optima. We consider this case later after examining the effect of introduced neutrality.

5.3.2 Neutral Interactions and Population Bloat

Before we go on to apply linkage modelling to problems with variable interactions, it is worth examining the effect of interactions on the population and resultant weighting under the current adaptive-elitist strategy. This strategy allows any individual with a fitness matching the greatest in the population to be included. For the linear problem used earlier where competing genes were assigned differing fitness contributions, the population rarely grew beyond a few individuals before the qualifying fitness was raised and the population reduced to a single optima again.

The dynamics change if the problem involves a degree of neutrality, i.e. different combinations of values (and later, modules) have identical fitness contributions. The function with the most neutrality in figure 5.4 is the step function. This is a general form of the simpler Royal Road function (R_1) [31] where the problem variables are partitioned into equal set, each adding a non-zero fitness contribution for only a single set of values. The order of the partitions is denoted k. If we consider the function

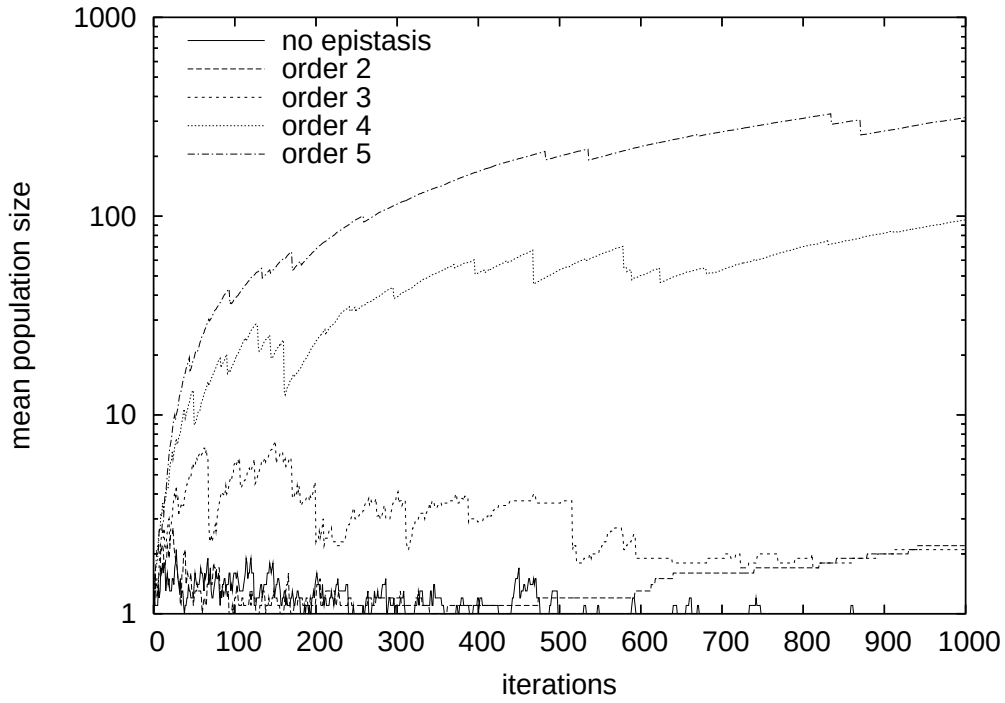

Figure 5.5: Effect of increasing epistatic dependencies on the size of an elite but unbounded population. Selected with a step function of varying order. Averaged over 10 trials.

where $k = 4$, this implies three of the four possible combinations have equal fitness and, all else being equal, there is room for three individuals in the population before the optimal combination is found. Each additional neutral variation, either within the partition or outside, increases this number exponentially.

The general effect on the resulting population size over time for various orders of k is shown in figure 5.5. The strong effect of the increasing epistasis on the population size for the adaptive-elitist population strategy is clearly indicated by the logarithmic scale for population sizes.

The exploration of unique, neutral variants is not necessarily a detrimental path, despite the overhead in maintaining the larger population. However, a serious issue arises as a side-effect of this expansion. The base weighting become relatively negligible compared to the weighting of the population and value sampling becomes

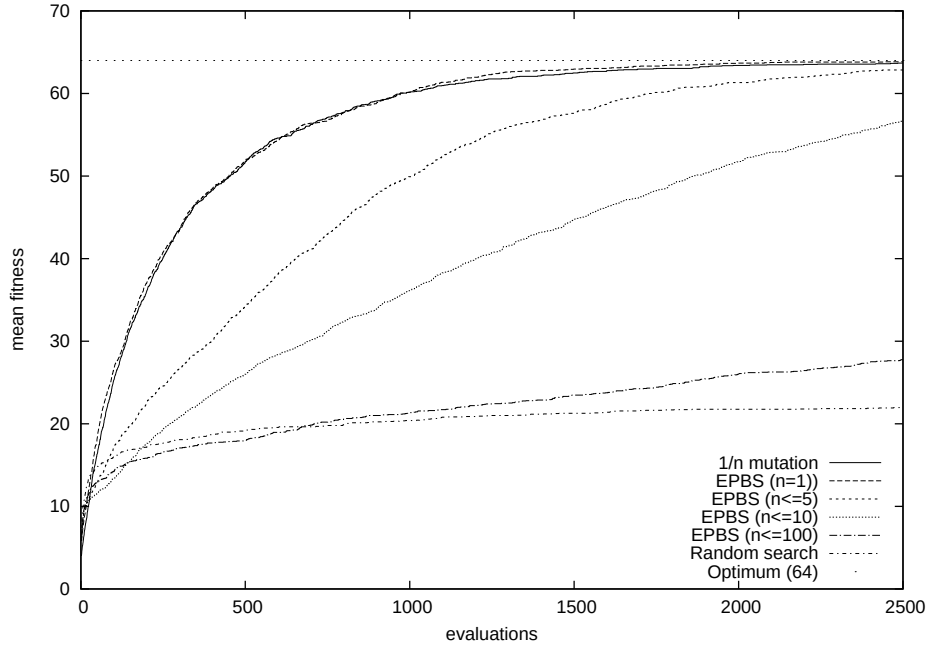

Figure 5.6: Performance degradation on elite population of increasing size bound. Problem is a step function with section length 4. Averaged over 100 trials.

increasingly determined by prior frequencies leading to rapid convergence to certainty in a form of premature fixation. Inspection of the adapting population shows this to be the case. The effect of this on performance is clearly demonstrated in figure 5.6 where, as an upper bound on the population size is increased, the performance degrades from a RMHC level to nearly that of a random search. We appear to have lost the advantage of an adaptive population size. The solution we apply requires us to constrain the bias of the population as follows.

5.3.3 Normalising Population Bias

The solution we take, to restore the effectiveness of population sampling, is a further feature from TEA. We 'fix the mutation rate' in terms of the base sampling rate

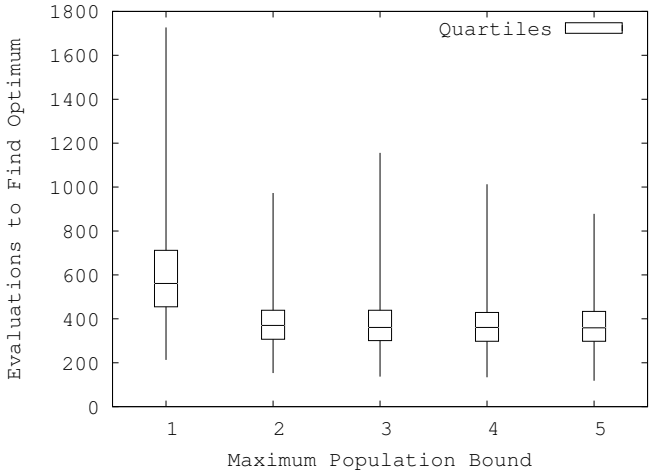

Figure 5.7: Effect of bounding EPBS population size on 64 bit step function (k=1, . equivalent to a one-max function). Averaged over 1000 trials.

relative to the population P (whose size is denoted $|P|$):

$$\omega_i = \frac{\sum_{x \in P} x_i}{|P|} \tag{5.5}$$

For a population of size 1, this is equivalent to equation 5.2. For a larger population, we are effectively distributing the weighting that a single individual would receive amongst all the individuals. We adopt this reasoning and will return to it as we extend our model to incorporate genetic linkage.

We refer to this revision as PBS. Figures 5.7 through 5.10 show the gain from this normalising of the population weighting with varying bounds on the population size.

We continue to use the step function for testing with a step size (k) of the specified order, either 8, 4, 2 or 1 (identical to *onemax*) and is essentially a concatenation of needle-in-the-haystack problems where each section only makes a fitness contribution when all variables have their optimal setting. The sections define contiguous partitions but they need not be since PBS samples values independently of locus.

The figures show a clear effect of increasing the population bound which is con-

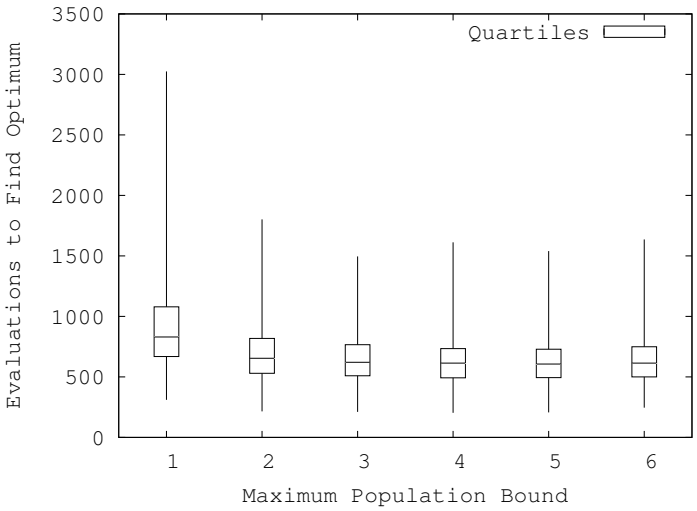

Figure 5.8: Effect of bounding EPBS population size on 64 bit step function, i.e. 32 concatenated 2 bit steps (k=2). Averaged over 1000 trials.

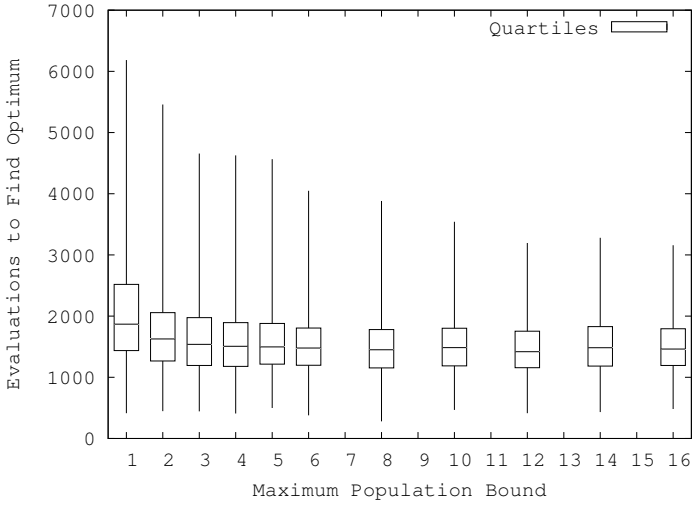

Figure 5.9: Effect of bounding EPBS population size on 64 bit step function, i.e. 16 concatenated 4 bit steps (k=4). Averaged over 1000 trials.

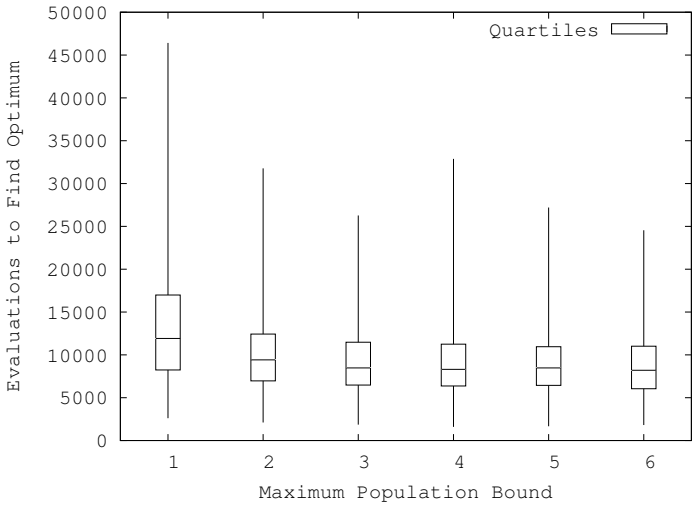

Figure 5.10: Effect of bounding EPBS population size on 64 bit step function, i.e. 8 concatenated 8 bit steps (k=8). Averaged over 500 trials.

sistent, albeit diminishing, for increasing step sizes k. For all values of k there is an overall reduction in the number of evaluations required to find the optima when the population is allowed to grow beyond more than a single individual. This reduction is relatively greater for lower values of k. There is also a marked reduction in the upper bounds of the 'hitting times' for larger populations.

We expect the larger epistasis from increased values for k to make a larger population more likely under our elitist strategy. For low epistasis the population does not grow very far before a fitter individual is found and the population reduced to it. For higher epistasis, where there are more equally-fit variants of an individual, the population grows. The levelling-off of the results for increasing values of k indicates that the artificial bounding is not coming into effect, i.e. the normalised population weighting does not have the runaway effect on population size shown earlier.

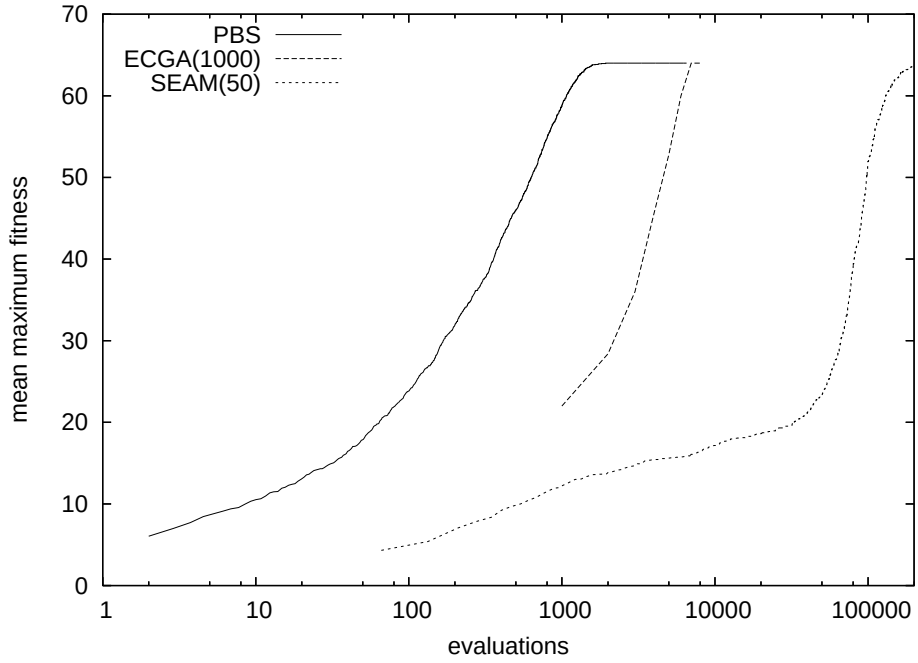

Figure 5.11: SEAM, ECGA and PBS on 64 bit step function ($k = 4$). Averaged over 100 trials.

5.3.4 Complex Models on Easy Modular Problems

We have demonstrated a problem with variable interactions that can still be solved effectively by an algorithm that does not explicitly model those interactions. It is instructive to check the behaviour of advanced dependency-modelling algorithms on this same problem.

Figure 5.11 shows the performance of both ECGA and SEAM against PBS on the step function used earlier. The logarithmic scale emphasises the two order-of-magnitude difference between the SEAM and the other two algorithms.

ECGA and PBS The ECGA is likely to require more evaluations than PBS due to its generational approach which require a sampling of 1000 new solutions for each iteration. After 1000 evaluations — and possible adaptations — PBS has all but reached the optimal solution.

SEAM We expected SEAM to under perform given the way it uses an *ad hoc* population to reduce the search space with a series of occasional joins. A little analysis can help quantify this by considering how many evaluations would be expected to be required to make the first successful join within a 4 bit module of the 64 bit step function.

A module is only created in SEAM if it is tested in a context in which it has a higher fitness than either of its components. For a block of order k, there is only one case for this: when all the other variables in the block are optimally set and the variables in question are not. The probability of sampling such a context is $1/2^k$. Since the probability of sampling an optimal pair of values (i.e. two ones) is $1/4$ and the probability of two variables being from the same k-order set of the n variables is $\frac{k-1}{n-1}$ then the probability of actually finding the first module (prior to testing it) is:

$$p(\textit{initial find}) = \frac{k-1}{4(n-1)}$$

This means the probability of making the first join at any particular attempt, i.e. finding a correct pair of genes and successfully testing them is the product of these independent events:

$$p(\textit{initial join}) = \frac{k-1}{2^k 4(n-1)}$$

For the result shown, $n = 64$ and $k = 4$, the probability of finding the first module from any sampling of the initial pool state is $1/1344$. Having found the first module, successive modules should be increasingly easier to find. However, the simpler algorithm still tends to find the optima *for all the variables* in a little over 1200 evaluations. This marks a clear limitation for the range of competence of SEAM.

5.4 The Requirement for Linkage Modelling

The PBS algorithm appears capable of effectively optimising solutions not just for linear problems but for problems with a significant order of dependency between variables. However, we have only applied this to problems where interaction between variables do not introduce local optima (termed 'easy' epistasis by Watson [49]).

A key hypothesis to be tested in this section is that competing modules comprising combinations of genes can be selectively processed in a population in the same way as genes were successfully selected previously. For this, we must consider problems far less amenable to our current strategy. This entails further examination of variable interactions which, in turn, prompts a modification to our population maintenance strategy to retain competing modules.

5.4.1 Deceptive Problem Testing

For the problems addressed thus far, where the problem variables can be partitioned into independent subsets, there is still only a single optimum for each of these. This is the *set optimum* and can, in principle, be reached by substituting a single variable at a time with no decrease in fitness. The simple biased sampling schemes, along with single hill-climbers, have been shown to be capable of optimising such problems in a reasonable time.

A harder class of problem emerges if the set has multiple optima. Each has a basin of attraction under selective forces and if the greatest optima has a relatively small basin then the problem is termed *deceptive*. Some fitness interaction between genes makes intermediary individuals less fit than the original ones. This interaction is not merely non-linear (as for the step function) but actually creates alternative basins of attraction.

For any set of interacting variables, we can only guarantee optimal values via an exhaustive search; since a fitness interaction may exist which makes any arbitrary combination of values optimal. However, it is sufficient to find combinations which lie within the deceptive basin of the set optima and can be adapted from there to the set optima. For the *trap function* below, the set optima has a basin with a minimal size:

$$f^{trap}(x^b) = \begin{cases} u(x^b) \text{ if } u(x^b) < k \\ k+1 \text{ if } u(x^b) = k \end{cases} \tag{5.6}$$

Where x^b is a specific subset of k interacting variables and $u(x^b)$ is the frequency of ones therein. This is trivially expressed in terms of the simple interaction equation 5.4.

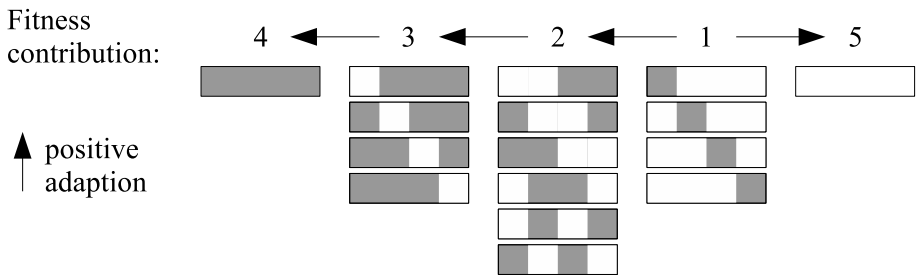

Figure 5.12: The deceptiveness of a 4-bit trap function, normally concatenated.

In the absence of any prior knowledge, a uniform distribution of value combinations would be appropriate. The density of these is an algorithmic parameter related to the likely relative size of the basin for the set optima. Having established a solution in each basin, we need to retain them long enough to allow them to attract solutions to their respective optimum and we can obtain the global optimum.

The problems require multiple alternate modules to be retained for combining at a higher level. Whereas the deceptive problems ultimately have only one global optima which need be retained, hierarchical ones such as H-IFF have multiple optima which can be combined in higher optima. This implies *competition* between sets of genes

and thereby the role for a genetically-enforced association between them.

Test Setup and Algorithmic Configuration

The scalability of the algorithms of interest was tested by concatenating from 4 to 10 independent traps. A trap size of 4 bits was found to be sufficient to tax the different algorithms without being needlessly intractable. The aim is to distinguish between algorithms rather than foil them.

The population size for ECGA was set to 3000 after some preliminary testing. Likewise, the number of contexts for SEAM was set to 120 and the base weighting for PBS was set to 10.

Results for Trap Scaling

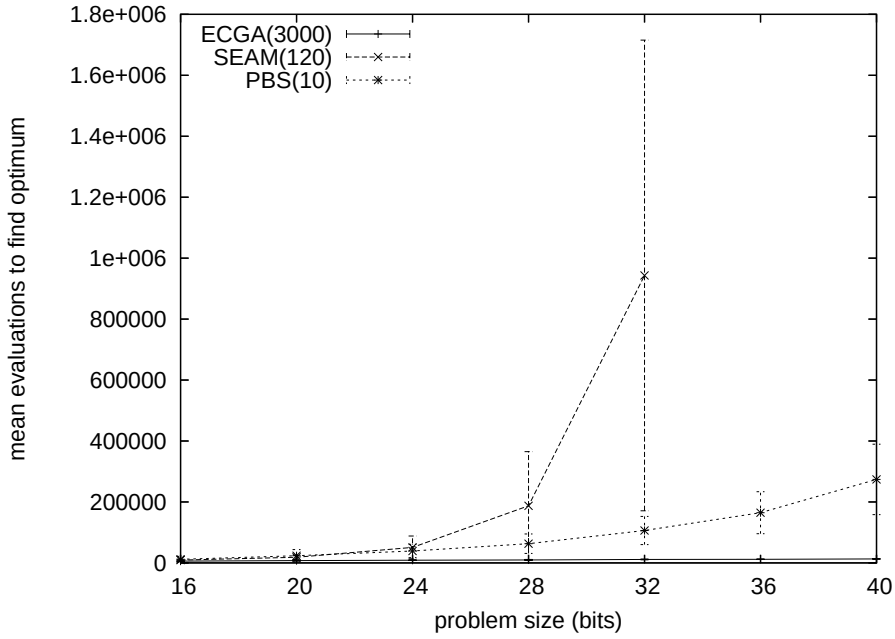

Figure 5.13: Comparison of ECGA, PBS and SEAM scaling, in terms of mean hitting time (with standard deviation) for concatenations of between 4 and 10 independent 4-bit traps. Averaged over 100 trials.

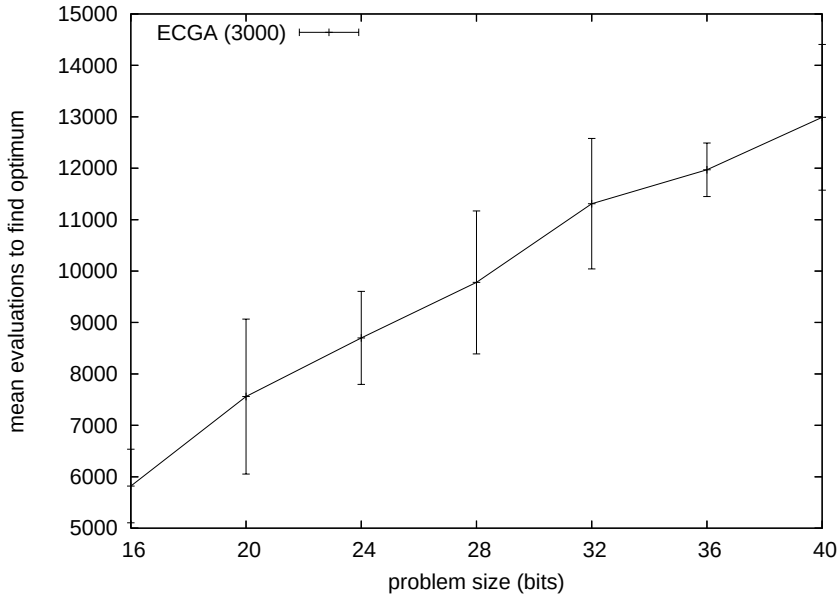

Figure 5.14: Scaling of ECGA in terms of mean hitting time for concatenations of a 4-bit trap of various order (rescaling of figure 5.13). Averaged over 100 trials.

Results are largely as anticipated, shown in figure 5.13. The ECGA, designed especially for this type of problem, scales best. Figure 5.14 shows this is more detail. Where the population was of sufficient size, the ECGA converged on the global optima in all cases.

The SEAM scales particularly poorly, for the same reasons as given for the step problem earlier, to the point where it became impracticable to obtain results for 9 and 10 traps. The PBS scales relatively well, avoiding the premature convergence in all cases, despite no mechanism — as yet — for learning linkage to avoid disruption.

5.4.2 Modifying the Population Maintenance Strategy

At this point, we discover an issue with the purely elitist population maintenance strategy. We find that by restricting the bias to the fittest individuals, we often exclude lesser individuals that nevertheless contain building-blocks. We suffer a loss

of information for a small gain in fitness.

To retain optima for multiple sets of variables requires extending our population maintenance strategy into one that is less exclusive. We wish firstly to acquire a range of different building-blocks, then mix and combine them. We do this by setting a minimum population size. As we asserted above, our initial distribution is uniformly random. From there, we gradually shift the balance toward the bias of our current optima whilst still maintaining an appropriately broad search distribution. It makes sense for this balance to shift over time as we gain confidence that our current optima are globally optimal.

The shift in balance is due to the changing frequency of components in a larger population. The population can still be elitist by retaining individuals so long as they are qualified by fitness. Qualification can also be dependent on the population size such that a minimal set of unique individuals is maintained.

We start with a uniformly distributed, random population. A new individual will then only replace one of the existing members with the lowest fitness if the new individual is at least as fit. This is similar to a conventional, steady-state replacement strategy but with the key difference that there is, as yet, no genetic linkage between values. Conventional genetic operators result in much stronger convergence from implicitly linking values within individuals. We are still only using frequency bias to set values for the new individual and therefore can maintain a far broader and exploratory search distribution.

5.4.3 Results of Population Refinement

The raw population for two typical runs with differing base weightings is shown in figures 5.15 and 5.16 using multiple snapshots. Each row represents one of the

individuals sorted by descending fitness. Shaded cells have the value 1 and clear cells have the value 0. The problem used is 4 concatenated 4-bit deceptive traps. The base weighting of the population is set to 1.0 and then 10.0.

Interpretation of Results

These snapshots show our new population maintenance strategy maintaining a rich distribution of building-blocks in the population without requiring any specialised diversity-maintaining techniques. Just the enforcement of uniqueness in the population can be adequate to retain a set of building-blocks in the population indefinitely, once they have been found. This will be important for providing a basis for adapting a linkage model.

The relative biasing between the population and base weighting determines the tunable balance between a search distribution that is largely independent of the current population and one that fully reflects its relative frequencies. For a problem that is more deceptive, we would set the base weighting higher to increases the probability of sampling the optima for any set of interacting variables. Problems with less deceptive interactions can be biased more toward the current known optima. The population snapshots for two typical runs demonstrates the differing behaviour resulting from low-level background sampling compared to high-level.

For a base rate of 1.0, shown in figure 5.15, the population quickly converges on optima. Most of these are naturally deceptive but as set optima are discovered they are amplified to fixation in the population. This continues until the optimal solution is sampled.

A base rate of 10.0, shown in figure 5.16, makes sampling more 'noisy'. There is less bias toward early deceptive optima and optima for multiple interaction sets appear in the population after just a few iterations. This is to be expected, since

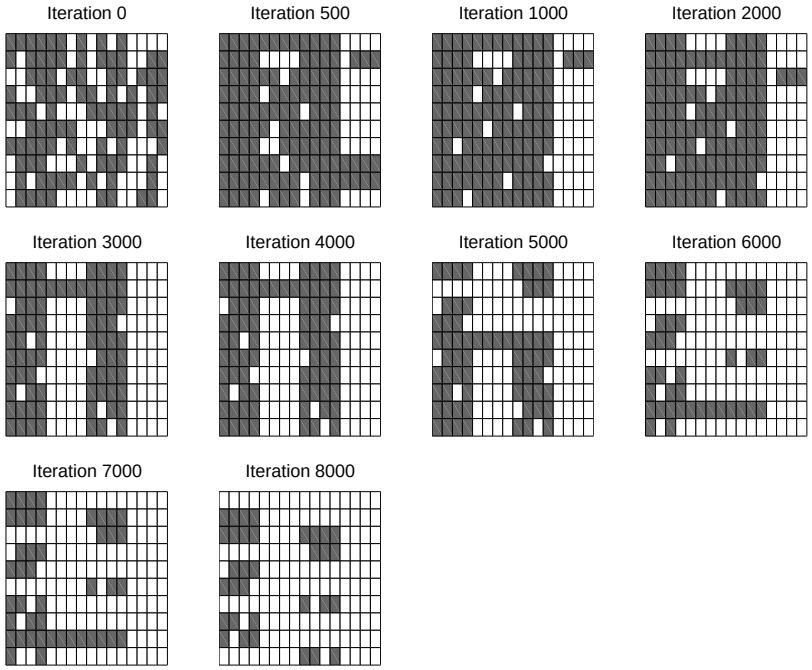

Figure 5.15: Sampling fixed solution set with base rate of 1.0

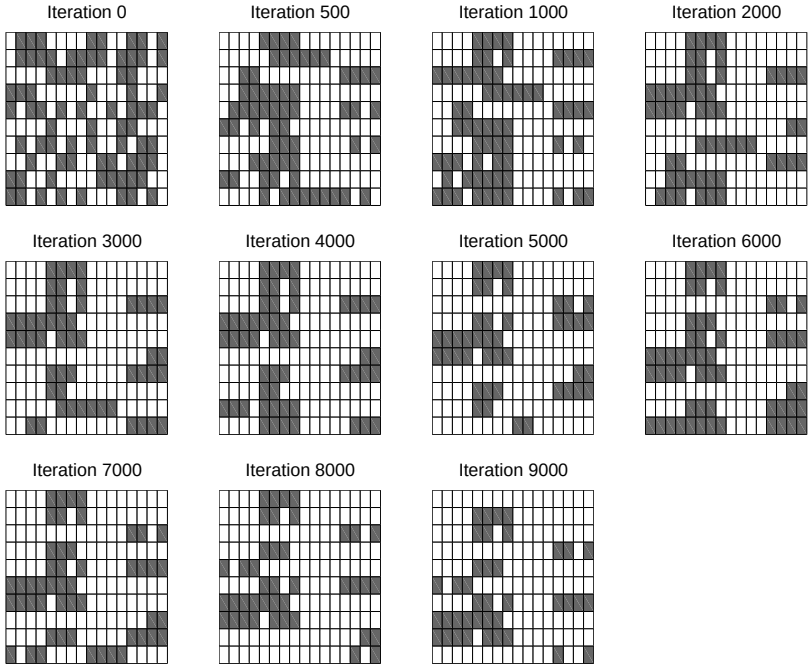

Figure 5.16: Sampling fixed solution set with base rate of 10.0

the probability of a set optima appearing in a random sampling is $1/2^k$ (1/16 in this case). However, to shift the population bias sufficiently to exclude local optima and sample the set optima together requires up to 10000 iterations. Within this time, the ECGA would normally have inferred linkage to reduce disruption and bring together building-blocks. In the worst case of deception, enumeration (or less efficiently, random search[7]) are the only effective methods for finding optimal values for the set of interacting variables. In conjunction with the population maintenance strategy above, this allows us to build up a rich distribution of building-blocks. However, it does not support their mixing. Independent sampling of the values from a heterogeneous population is overwhelmingly disruptive of the building-blocks preserved in individuals. This leads to the testing of our proposed methods for identifying and imposing genetic linkage.

5.4.4 Hierarchical Interaction Testing

In order to test TEA further we need to bridge the gap between modular and hierarchical problems. Although instances are abundant in the real-world domain, there is no generally accepted standard benchmark abstraction of problems with hierarchical dependencies.

A modular problem structure is extended into a hierarchical one by extending our equation for simple interactions into one which allows complex ones. In fact, this is shorthand: a hierarchical problem has at least one interaction set contained within another.

An early candidate is the hierarchical Royal Road function (R2) [31] which takes

[7]The difference between random search and enumeration being enumeration never evaluates the same solution twice. If we imposed the same condition on random search then it could be expected to perform as well.

those fitness interactions of the R1 function[8] and recursively forms new ones from adjacent pairs of interaction sets. The effect is set to the order of the interaction set. Since there is no local optima, the R2 is easily addressable by the prior algorithms and is of limited use for our current interest.

Since the H-IFF defines fitness interactions for homogenous sets of values and R2 defines interactions only for sets of 1s, the interaction network for R2 is generally a subset of that for H-IFF. The effect of the interaction for both functions is assigned the number of values in the set. The key property of the H-IFF that is of interest here is the complete set of competing modules at all levels which make the problem intractable for most algorithms.

Although SEAM has just been shown to perform poorly on modular problems, it was formulated in conjunction with the H-IFF and ought to excel on functions with a very tight hierarchy. We quantify this now, performing a comparison with ECGA and PBS on the H-IFF.

Test Setup and Algorithmic Configuration

We tested the algorithms on H-IFF problems from size 16 through 128 bits. This corresponds to 4 through 7 hierarchical levels. We used the SEAM algorithm for comparison where appropriate. We did not require both optima for the H-IFF to be found, although this may have put SEAM at a further advantage.

The HGA is markedly effective for this problem (for which it has been optimised) but it was not considered to be a useful algorithm for comparison given its limitations demonstrated earlier.

[8] A concatenated step function where $k = 4$.

pop. size	mean evals.	successful trials
5000	45000	6/100
10000	87600	25/100
20000	162000	10/10
30000	249000	10/10
40000	296000	10/10
50000	350000	10/10

Table 5.2: Effect of ECGA population size on capacity to solve HIFF-64 efficiently and reliably, in terms of both number of evaluations and of run time.

SEAM operates by incrementally joining pairs of modules conditional on their performance within n randomly generated contexts.

The SEAM was devised specifically to address the H-IFF problem [49]. Where the problem is of a small scale, i.e. less or equal to 16 bits then SEAM offers little advantage in terms of efficiency. In fact, it lags behind random search.

ECGA The setting of the population size for ECGA is somewhat trial-and-error. Each problem size has a corresponding optimal population size to produce an optimal solution with a given reliability.

We demonstrate this issue in table 5.2 with 64-bit H-IFF. There we illustrate how a minimum population size is required to find the global optima with an acceptable degree of reliability but going beyond this size adds surplus evaluations.

Figure 5.17 shows more clearly how an increasing population size in ECGA results in a roughly proportional increase in mean hitting time. Smaller population sizes potentially show increased efficiency but also entail a higher risk of convergence to a sub-optimal solution. Of 100 trials, a global optimum was found in only 6 for a population size of 5000. For a population size of 10000, this rose to 75.

Our methodology was therefore to increase population size by appropriate increments until all trials converged at an optimum.

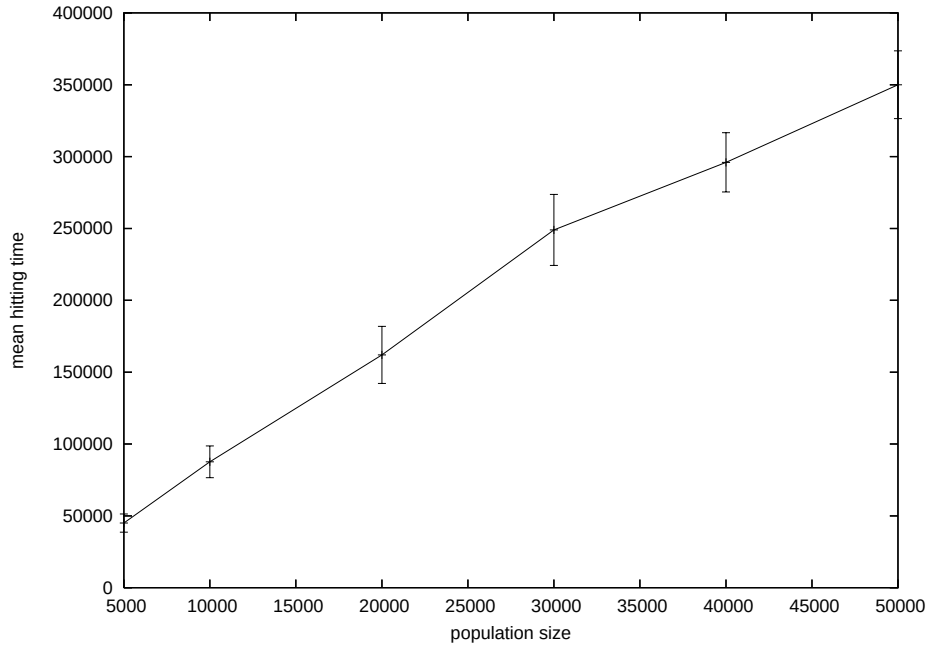

Figure 5.17: Mean hitting time (s.d.) for ECGA with varying population sizes on 64-bit H-IFF. Averaged over 100 trials, with error bars showing standard deviation. Premature convergence occurred for sizes less than 20000 as detailed in table 5.2.

PBS The PBS algorithm was configured with a population of size 10 and a base weighting of 1.0. These parameters were found by trial-and-error to be approximately optimal for this problem.It was compared to SEAM for 16 and 32 bit H-IFF with results shown in figures 5.21 and 5.20 respectively. Discussion follows in the next section.

Hierarchical Scaling Results

ECGA As before, obtaining optimal performance from the ECGA relies on an appropriate population size. An inadequate set of solutions resulted in convergence to a sub-optimal solution. An overly large population size led not only to superfluous evaluations and slower convergences but also hugely increases the work of the model search. This is reflected in the mean actual run time for optimisation, shown in figure

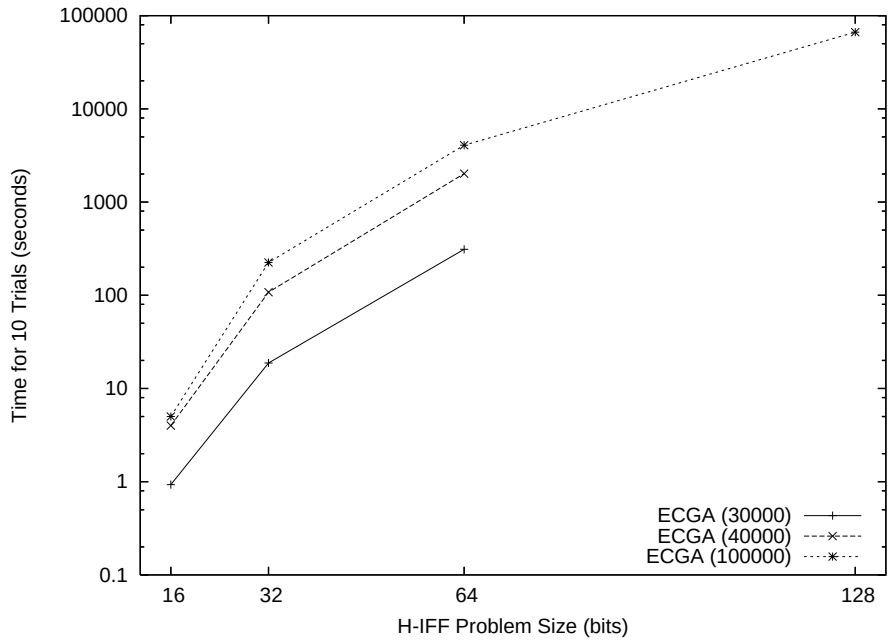

Figure 5.18: Mean convergence times for ECGA with varying population sizes on a range of H-IFF problem lengths. Averaged over 10 trials.

5.18 with the logarithmic scale.

Figure 5.19 shows that, with a sufficiently large population, the number of H-IFF evaluations scale linearly with the problem size. However, with an increasing problem length, the space of MPMs for the ECGA to consider becomes significant and time taken to search it becomes overwhelming.

Setting aside this additional computational effort, the ECGA is shown to have more success with scaling for the hierarchical problem than SEAM does on the modular problem.

PBS PBS is far less successful on H-IFF than for the concatenated trap problem. For 100 trials, PBS only found a global optimum in 26 cases within 300000 evaluations. Figure 5.20 shows it rapidly converging to sub-optimal solutions compared to the SEAM. Actually, convergence as a term needs to be qualified here since the population

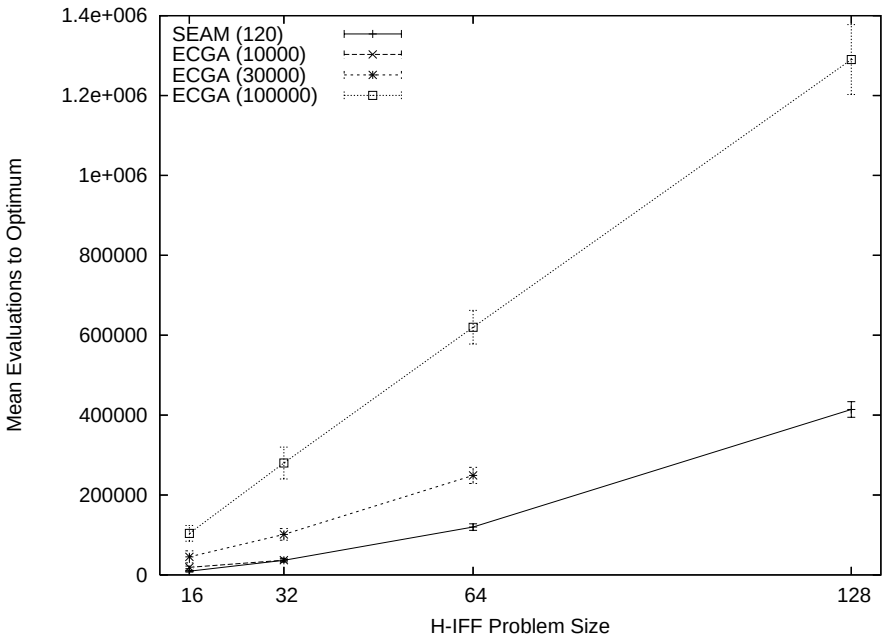

Figure 5.19: Comparison of SEAM against ECGA for scaling on the H-IFF. ECGA tested with different population sizes. Points only plotted where algorithm find optima in all 10 trials.

still maintains a set of unique individuals. Convergence here is restricted to subsets of values rather than entire solutions.

The difficulty that PBS faces is in sampling appropriate values for building-blocks above a certain size. PBS converges to a mean fitness of 160 which corresponds to 5 (rather than the full 6) levels of the hierarchical 32 bit function. PBS is more successful on the 16 bit H-IFF, achieving an optimal fitness within 100000 evaluations for 79 of the 100 trials. Figure 5.21 shows this better relative performance against SEAM, although it should be noted that the search space for 16 bits is only 65536 in size. This highlights the long-anticipated need for linkage-learning and marks the limits for an algorithm with no linkage-learning capability.

Whilst linkage modelling has been shown not to be essential for solving deceptive problems, hierarchical problems such as the H-IFF demand the retention of competing

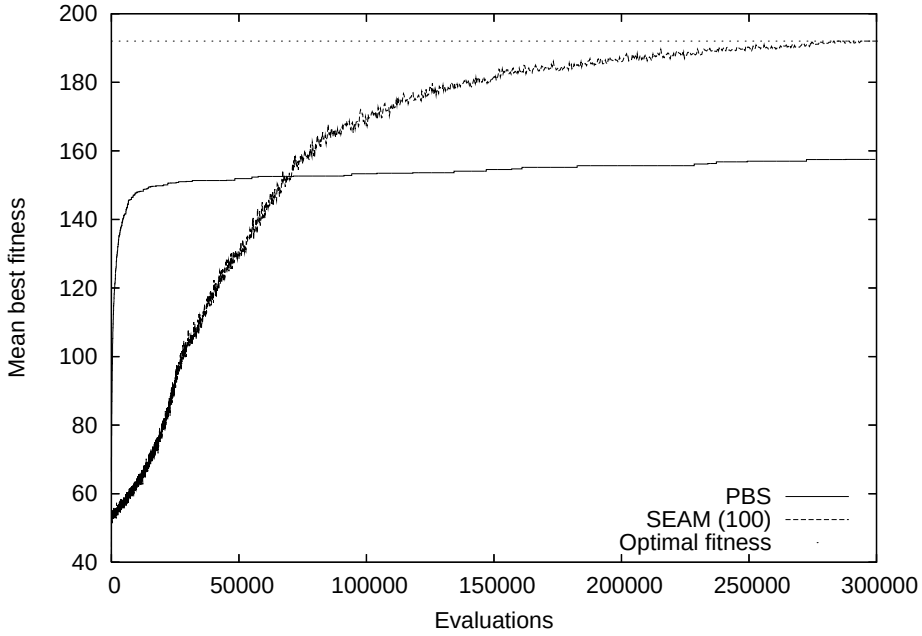

Figure 5.20: PBS convergence compared to SEAM on 32-bit H-IFF. Averaged over 100 trials.

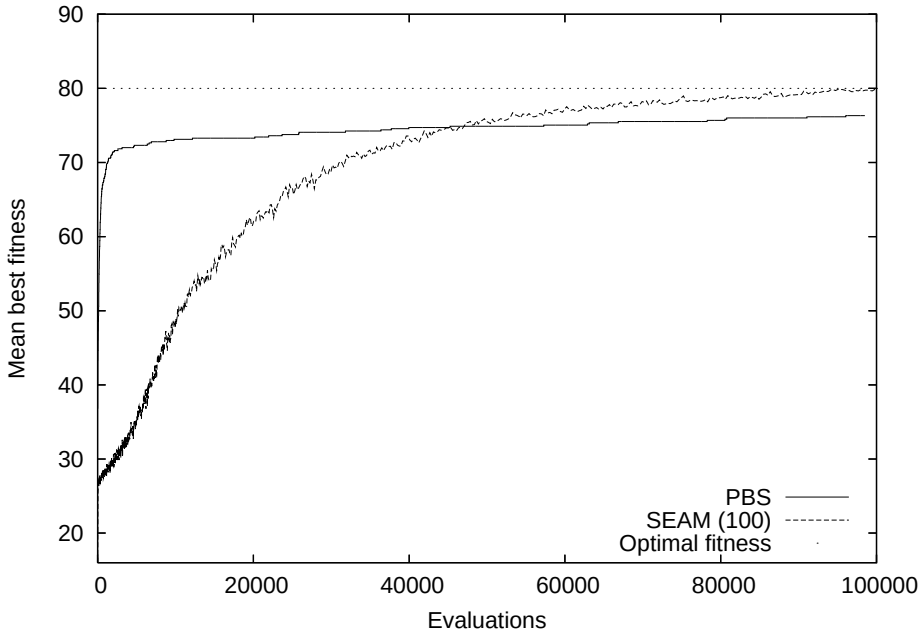

Figure 5.21: PBS convergence compared to SEAM on 16-bit H-IFF. Averaged over 100 trials.

sets of genes. If we bias too strongly on early building-blocks then these will become established in the population to the exclusion of others which compete with them. This may prevent higher-order building-blocks emerging.

We therefore expect linkage support to be crucial where we have competing fitness interactions and a degree of hierarchy. This lead into high-level mixing of potentially competing groups of values motivates our final extension of PBS into the complete TEA.

5.5 Validating Structural Selection

In supporting genetic linkage, we anticipate reducing the disruption caused by independent sampling of interacting variables. Rather than sampling single alleles, we will also allow specific groups of values to be sampled. If these groups, or modules, comprise more optimal combinations of interacting values then we expect them to increase in frequency within the population.

The principle behind this is that the mean fitness of individuals containing a specific module with optimal settings is greater than that of individuals containing any of the components independently[9]. Therefore the individual containing the building-block is more likely to be added to the population.

The difference between our linkage and the limited parentage linkage of the GA is of granularity. We do not assume a gene is linked to every one of the genes it coexists with. For genetic linkage to have a beneficial impact, candidate interaction sets need to be assessed and optimal allele sets amplified. We consider the selection of candidate modules first before examining how they are weighted and used.

[9]This is similarly motivated to the schema theorem [12] and the relation is discussed later in section 6.1.

5.5.1 Testing Transmutation

Before being subject to the rigours of selection to be amplified or discarded, a module must be instantiated for the first time. Transmutation, the source of candidate modules, is a relatively straightforward matter. We apply transmutation uniformly to individuals in the population. Having selected an individual, we can either modularise or demodularise it[10].

According to our hypothesis regarding structural selection, high-quality modules are expected to propagate in the population via the advantaged individuals which contain them. We now show why relying solely on individual selection to process candidate modules might be naive.

Problem Configuration The problem configuration for testing the effect of transmutation is designed to tax the capabilities of a search with no dependency support. The problem is a concatenation of 8 deceptive traps, each of 4 bits. If each converges to the sub-optima then it contributes a fitness of 4. The optima for each trap is 5. We therefore would expect an algorithm that performs a largely local search to converge to a solution of fitness 32. We have seen our population-biased method perform better than this. Although the population may easily find and retain the optimal setting for each trap, without some form of genetic linkage it is more difficult to sample these into a single individual. The deceptive gradient of the function hinders the spread of optima against sub-optima. We now examine the effect of adding candidate modules from transmutation for individual selection to process.

[10]Modularisation would be inapplicable if there is an inadequate supply of components in an individual. Conversely, demodularisation can only operate on an individual containing composite modules.

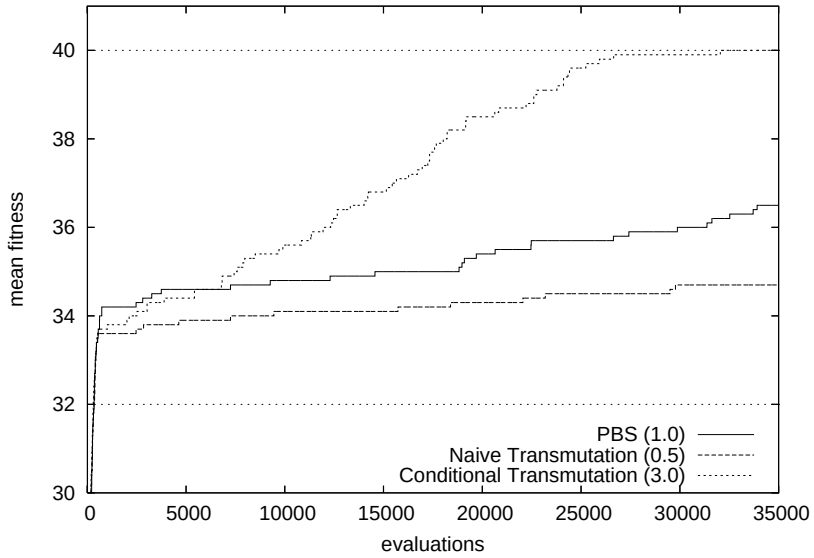

Figure 5.22: Comparison of PBS, 'naive' unconditional (with a modularisation probability of0.5) and conditional transmutation (with a modularising threshold of 3.0) for concatenation of 8×4 bit traps. Dotted lines at 40 and 32 show global and local fitness levels. Averaged over 10 trials.

Algorithmic Configuration All the algorithms used a population size, $n = 20$. The PBS algorithm was assigned a base weight of 1.0. The 'naive' transmutation is applied 10 times at each iteration with an even probability of being a modularisation or demodularisation operation. The conditional transmutation, introduced shortly, uses identical parameters where applicable.

Interpretation of Results The result of this comparison is given in figure 5.22. It demonstrates that naive introduction of transmutation actually degrades performance down when compared to PBS, i.e. no transmutation. Inspection of the adapting population and pool supports the following interpretation and resolution.

Our population maintenance strategy is successful in building up a diverse set of solutions. This diversity can be moderated by varying the population size and the base weighting. However, since the population is designed to retain optimal individuals, the rate at which individuals are replaced (the 'turnover') necessarily decreases over

time. This is an issue as this turnover is the mechanism via which candidate genetic linkage structures are amplified or discarded.

At this point, we come up against a key problem: that the adaptation of the dependency model is limited by the adaptation of the population. The turnover of individuals decreases as the population accumulates fitter individuals. This is despite there being strong linkage disequilibrium information in the population ready to be exploited. This produces a dilemma: whilst the fit but diverse population is retained, it is increasingly difficult for new individuals to qualify. However, if we discard any of the qualifying population then we risk losing information on set optima. Since structural adaptation is dependent on population adaptation, it tends to stagnate as the population settles.

The recombinative approach weakly supports the preservation of building-blocks by restricting parentage to a pair of individuals, i.e. sampling from a small subset of the population. This increases the probability of sampling values from a common individual and thus potentially reducing disruption. However, it will inevitably link values which are non-interacting in the same way. We still want a more finely-grained linkage bias within a generally diverse search distribution. One possible resolution of this issue involves removing this dependency such that structural adaptation can occur within a relatively static population.

5.5.2 Inferring Fitness Interactions

We have shown it is not enough to simply allow selective forces work on the modules produced via transmutation; the transmutation itself must be directed via some adaptive heuristic. The motivation is unchanged; we wish the search distribution to approximate the optima distribution described by the population distribution. The

method for this modifies transmutation to apply it conditionally. A simple inference from population statistics can determine whether or not the modularisation or demodularisation would make the search distribution better approximate the population. This gives us a straightforward way to ensure dependency modelling continues to adapt even as the population composition is static.

We therefore need a mechanism to infer interactions from the population which is independent of adaptation in the population set. The interactions can still be translated into genetic structures via transmutation, which can be executed concurrently with the population adaptation, but we need to make the operation selective. Fortunately, it is a straightforward matter to define a precondition for transmutation based on simple statistical measures.

Consider a pair of components, c_0 and c_1 from an individual x which may be sampled independently, i.e. they are not genetically linked. We wish to know if they co-occur in the population with a higher than expected frequency. If there is a positive fitness interaction between them then population selection should instantiate them together with a higher than expected frequency. The expected joint frequency must take into account the frequency of genetic structures which are exclusive with the independent sampling of the two components. We do this by excluding individuals which contains a component that intersects with both c_0 and c_1.

Within this subset of the population, where the marginals of the components are denoted $O(c_x)$ and the joint distribution $O(c_0, c_1)$, a measure of their linkage is given by:

$$\delta(c_0, c_1) = \frac{O(c_0, c_1)}{O(c_0)O(c_1)} \tag{5.7}$$

Where $\delta(c_o, c_1) > 1$ indicates a positive interaction. By setting an appropriate threshold, θ, on δ we can filter modules to an arbitrary degree.

Result of Filtering Transmutation The application of this condition (using a threshold $\theta = 3$) produces the desired effect, shown in figure 5.22. The modified algorithm rapidly reaches the local optima as PBS and the naive algorithm do. It then steadily discovers each of the trap optima significantly faster than the PBS or naive algorithm. The refinement, in ensuring that unlikely modules are filtered out of the selective process, strongly supports a healthy turnover of individuals. A consequence of this is to make demodularisation effectively obsolete. Inaccurate modules still occasionally enter the pool but are rapidly selected against as genuine ones are amplified. This simplifies the algorithm another degree.

Determining an appropriate threshold for these experiments was based on observation of the pool. Setting θ too low had the effect of bloating the pool with spurious modules akin to the behaviour of the unconditional transmutation. At the other extreme, setting θ too high meant no modules were added to the pool, resulting in a behaviour like the PBS. Run-time inspection of the pool showed that when a balance was struck the pool would stabilise with a dynamic set of known building-blocks. A setting of θ of between 1 and 4 appeared to produce a fairly smooth continuum between these extremes. Whether this would hold for other problems is an open question.

We now go on to compare the refined form of TEA with other algorithms of interest.

5.5.3 Comparing TEA to Linkage-Modelling Algorithms

Having verified the mechanism of structural selection, we now perform a comparison to the prior algorithms of interest to assess the relative performance and identify any final limitations of the TEA.

TEA configuration The TEA was configured with a static population size of 30 individuals with a base weighting of 1.0 and a transmutation threshold of 3.0. This was found by trial and error to be adequate to maintain a diversity of optima in the population and effectively filter modules. These setting were used across the range of problem sizes below.

Behaviour of TEA on Concatenated Trap

The result of TEA applied to the concatenated trap problem is shown in figure 5.23. Prior results for PBS and ECGA are shown for comparison. As TEA is essentially PBS with transmutation deactivated[11], the effectiveness of the operation on improving performance for this problem is clearly demonstrated.

Inspection of the adapting population shows it quickly adapts toward the deceptive optima before finding the global optimum for each partition in turn via base sampling.

When an interaction set is instantiated in a new individual and given a place in the population then there is an increased marginal bias toward those values. Over time these could exponentially increase in frequency. In the PBS, the concatenated trap function is solved via this process. However, when these values are identified as interacting by the TEA and transmutated into a new selectable entity, the pool shows this new selectable unit is amplified far more rapidly by selection, usually to fixation.

Behaviour of TEA on H-IFF

The result of the filtering modification is highly significant and is shown in figure 5.24 in the context of the earlier results. It shows a clear gain over the other algorithms

[11]This effect can be produced by setting the θ threshold prohibitively high.

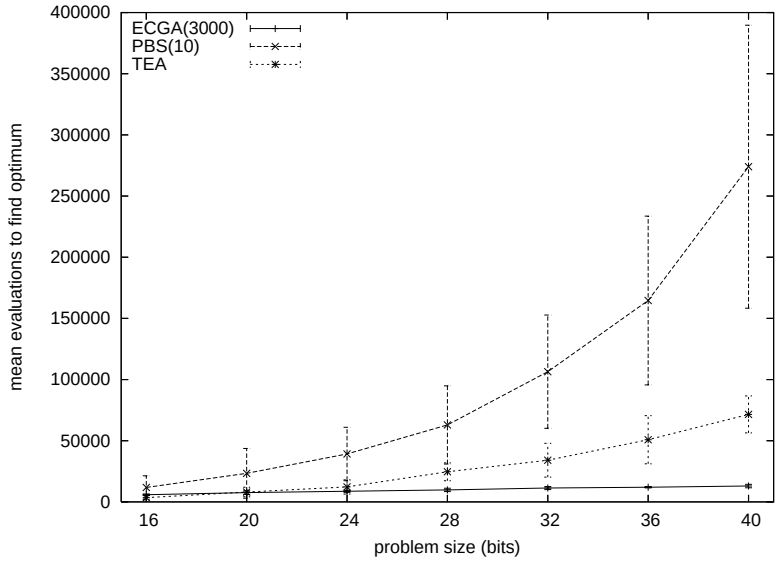

Figure 5.23: TEA scaling against PBS and ECGA on concatenated 4-bit trap function. Averaged over 10 trials.

for H-IFF problems at sizes 16 and 32 bits. At these scales, the problem can only effectively be solved by adapting the representation, i.e. identifying building-blocks, and run-time inspection of the pool confirms this.

Problems of a larger scale encounter an issue which prevents the optima being found in all cases by TEA. Close examination of the pool at this stage suggests that this is due to sets of independent variables fixed in the population which are non-optimal in the context of the rest of the variables. An accurate and diverse decomposition of the other optima invariably occurs but only base sampling is capable of moving the population from this region of local optima. Since the fixed set belong to an interaction of higher order (typically 8 bits) it becomes relatively improbable for base sampling to provide this shift. The consequences of this are discussed in section 6.4.1.

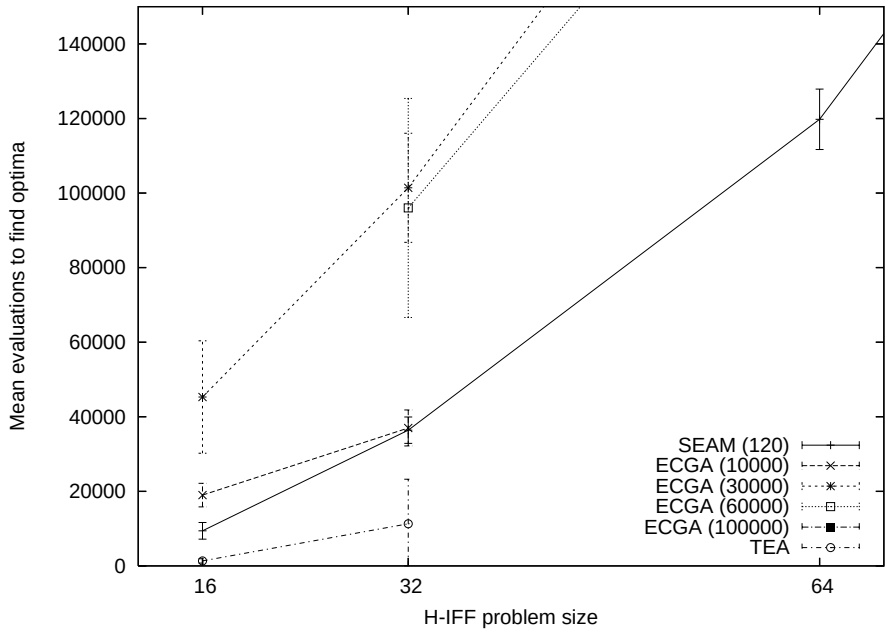

Figure 5.24: TEA scaling against prior algorithms on H-IFF. Averaged over 10 trials.

5.6 A Polyhierarchical Problem

5.6.1 Motivation for a Polyhierarchical Function

There appears to be no reason why modules should be limited to competing with their individual components. If a component can be part of multiple modules then the model is capable of representing polyhierarchical dependencies rather than just hierarchical ones. Overlapping modules are a feature of many real-world problems. The ability to allow overlapping modules to compete is crucial if a wider range of structured problems can be tackled which often have varying degrees of interaction strength to boot. Approaches which partition solutions into strictly non-overlapping modules are ill-equipped to tackle such problems.

5.6.2 Defining a Polyhierarchical Function

The evaluation of a polyhierarchical problem is given by equation 5.4 as before, i.e. as the accumulated fitness contributions from multiple interaction sets. In constructing a polyhierarchical problem, as for the hierarchical problem, we will require that global optima can only feasibly be obtained by successively combining sub-solutions. We describe a simple concrete example of a polyhierachical problem within these terms that is minimal yet hard.

Our problem is composed of two stages. The first step is to define overlapping modules *which need to be mutually exclusive in order for the problem to be genuinely taxing*. This is easily illustrated in figure 5.25 using the familiar schema notation [31] where $k = 2, m = 64$ and $l = 32$.

```
00******************************        *11******************************
**00****************************        ***11****************************
****00**************************        *****11**************************
******00************************        *******11************************
********00**********************        *********11**********************
**********00********************        ***********11********************
************00******************        *************11******************
**************00****************        ***************11****************
****************00**************        *****************11**************
******************00************        *******************11************
********************00**********        *********************11**********
**********************00********        ***********************11********
************************00******        *************************11******
**************************00****        ***************************11****
****************************00**        *****************************11**
******************************00        1*****************************1
```

Figure 5.25: Overlapping interactions produced by a single bit offset. `000...0` and `111...1` are the resultant optima.

Such a set of interactions clearly precludes a useful model which partitions the variables into non-interacting sets, yet it is not generally hard. Since there are no local optima, a local search strategy would be successful in finding one of the two

global optima: `000...0` and `111...1`. This has the effect of making any pair of variables impossible to optimise in isolation; yet not requiring any interaction modelling, via modules, beyond a pair-wise one.

Extending this modular structure into a hierarchical one could easily be performed by adding interactions that combine homogeneous, adjacent pairs of the above interactions, e.g. `*11*****...` and `***11*****...` to add the interaction `*1111***....` However, this does not actually increase the difficulty of the problem structure since this does not add any new optima. For this, we need to define interactions which create new global optima – excluding the previous ones – via alternative combinations.

The second level combines a number of the lower-level interactions to create a set of large-scale interactions which produce correspondingly large fitness effects. These large interactions define new global optima which have a significant number of conflicting values with the lower optima.

```
00*11*00*11*00*11*00*11*00*11***
**00*11*00*11*00*11*00*11*00*11*
*11*00*11*00*11*00*11*00*11*00**
```

Figure 5.26: High-level, high-order interactions. These each produce a fitness contribution of 20.

The definition of the two levels of interaction are intended to separate the algorithm with a polyhierarchy-modelling capability from the simple hill-climber. The first level optima, as has been noted, can be obtained straightforwardly from single bit adaptations. The second level optima requires a significant number of the lower-order modules to be found and then combined. This is a non-trivial challenge that requires a specific assignment to 20 bits of the 32.

Figure 5.27 demonstrates the RMHC search easily locating the lower optima for this problem but never a high order optima. The performance of TEA is compared for a range of population sizes with a base weighting equivalent to an average 1 bit mutation rate and the threshold θ set to 3.

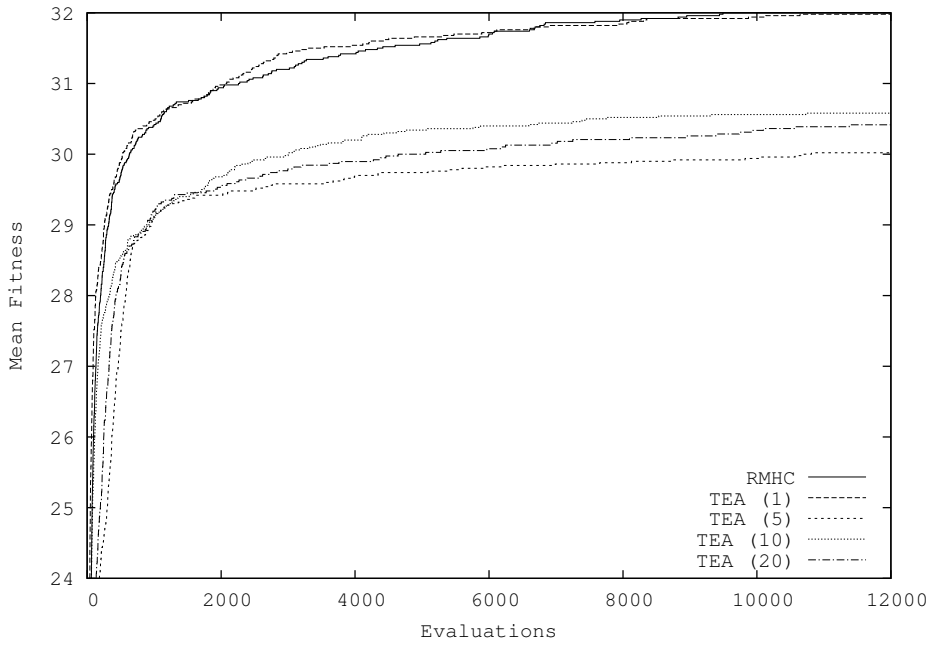

Figure 5.27: TEA on minimal polyhierachical problem. RMHC included for comparison. Averaged over 100 trials.

With a unit population size TEA emulates the RMHC to find the local optima. Increasing the population size raises the potential for it to suggest candidate modules. However, the high-level optima remain out of reach for TEA. Although a number of the lower-level modules are identified — as for the hierarchical problem — not enough are maintained for the higher-level optima to be found. Raising the base weighting of TEA results in a more diverse population distribution and consequently more building-blocks identified. However, the additional noise makes progress slow as the candidate modules have a reduced selection probability.

To enable TEA to succeed on this problem it must be able to reliably build up an adequately rich pool with a weighting that is sufficient to build higher-level structures. Only this will enable them all to be sampled together in a single solution. Progress in this direction is also expected to strengthen its capabilities on the hierarchical problems, if not the simple modular ones.

5.7 Summary

The strategy for these studies were to incrementally validate the hypotheses from earlier chapters, starting with simpler models and problems and building up in a reasoned progression to ones of greater complexity. The result of these studies was the systematic testing and revision of the presented approach to the level where it was competitive with an array of state-of-the-art algorithms on a broad range of problem classes. The course of the experiments can be marked by the following results.

5.7.1 Univariate Model Comparison on Linear Problems

By casting the univariate-modelling EDAs into a common form, we obtained a strong and fair comparison over them on a standard linear problem in section 5.2. This showed that they under perform relative to a model equivalent to a RMHC algorithm. A population-biasing extension to this, EPBS, incorporated features advocated earlier to produce a further performance gain.

This empirically validated the principle of maintaining the population purely as an informative sample from which to weight selectable components with no fitness interactions. We next examined the case where fitness interactions exist.

5.7.2 Interactions Impact Population Strategy

The generalisation in section 5.3.1 of the problem class into one that included fitness interactions revealed byproduct effects that necessitated the population weighting and the maintenance strategy to be revised.

Experiments in section 5.3.2 showed problems with epistatic dependencies causing neutrality and a corresponding increase in population size. This is a consequence of simple population-based sampling and elitist population inclusion.: a straightforward cumulative weighting bias suffers a fixation of non-optimal values.

This prompts the first revision of the weighting scheme in section 5.3.3 which successfully addresses this issue and yields an insight into bounding the weighting bias to avoid undue fixation. Our solution was to normalise the weighting given to the population relative to the background sampling. In essence, this distributed the influence of a single individual amongst a set of unique but equally-fit solutions. We noted that simply bounding the population size was a less effective modification. Also that the problem is still amenable to a univariate search at this stage.

5.7.3 Deceptive Interactions Require A Minimal Diversity

The introduction of deceptive interaction, or hard epistasis, in section 5.4.1 raised additional challenges which were addressed in discussion. The required diversity in the population was shown to be missing, precluding the identification of building-blocks. We showed that PBS was able to solve easy epistatic problems.

We demonstrated the need to retain a population with a fixed minimal size if hard epistatic problems are to be solved. A modified population maintenance strategy was proposed in section 5.4.3 and verified as effective. This led us to the point where we were ready to identify and process building-blocks.

5.7.4 Current Models are Highly Problem Specific

We found in 5.4.1 that the ECGA does not scale well on hierarchical problems in terms of the time required to optimise them. This is a byproduct of the large populations required for optimal convergence.

We also determined in section 5.3.4 that although the compositional algorithms are ultimately capable of solving hierarchical problems that are difficult for other algorithms, they severely under perform on less tightly structured problems, i.e. with larger interaction sets.

5.7.5 Transmutation Effective for Adaptive Search

We found in section 5.5.1 that selection in itself was inadequate for identifying building-blocks from the candidate modules produced by random transmutation. We traced this to the elitist population strategy producing a diminishing replacement rate as it becomes better adapted, and failing to provide the constant turnover of individuals required for effective discrimination between competing modules.

A simple condition was introduced in section 5.5.2 to validate prospective and existing modules, making pool adaptation independent of population-level selection. This proved successful at identifying building-blocks given our informative population. Once the building-blocks were identified they were rapidly exploited by selection, as hypothesised. There remains an issue of maintaining adequate diversity across all variables for the most highly structured problems.

5.7.6 Issues in Maintaining a Balanced Pool

In order to probe the limits of TEA a minimal instance of a polyhierarchical problem was devised in section 5.6. In principle this problem should be equally amenable to the TEA method. In practice it reiterated areas where TEA may need more analysis.

Although able to reliably identify some building-blocks, maintaining a large set of these is not straightforward. This issue of balancing a base weighting with the population bias so that a diverse population can be maintained without making the search too noisy is a top candidate for future analysis and development.

Chapter 6

Discussion and Conclusions

In this chapter we finally address the hypotheses in light of the experimental studies to determine the contribution. This leads to comments on the problem form employed. The conclusion summarises the contribution of the thesis before going on to propose future directions.

6.1 Review of Motivation

Different algorithms draw their efficiency from making presumptions (informed or otherwise) regarding the structure of a problem. Simple mutation-based searches have been shown to work well on problems with a low degree of structure but falter where significant competing dependent sets are required to be respected. Such problems, with a modular or hierarchical structure, have motivated algorithms which aim to infer the dependency structure from a set of solutions into an explicitly probabilistic model.

This disjointedness is unsatisfactory; we would like to carry our evolutionary inspiration throughout. Yet we have seen the limitations of a model based purely upon mutation and recombination in constructing and maintaining sets of values for dependent variables. The schema theorem was presented to theoretically support the canonical GA's capacity to

execute the building-block hypothesis. From the discussion given in chapter 2 there are various strong caveats inherent to this theorem, to wit, a static and favourable variable linkage representation implicit to the ordering and set by the user from the outset.

Fortunately, another biological abstraction of adaptive complexity provides a prospective paradigm: the evolutionary transition. This has allowed us to present an approach which makes no presumptions regarding the structure of a problem, but uses competitive performance to validate candidate representations alongside the search for optimal solutions.

The principle of TEA, like the idealised GA, rests on the propagation of building-blocks through the population. However, the population itself has a more specialised role: as a persistent set of unique, well-adapted samples. More significant is the lack of implicit linkage in TEA. Building blocks are explicitly encoded at run-time and processed as units of selection. This effectively makes TEA a closer adherent to the building-block hypothesis than the canonical GA.

Interesting commonalities can also be found with Learning Classifier Systems (LCS). For a concrete example, we take the Michigan-style XCS [51] which is designed to produce classifiers with increasing generality and accuracy.

In this system, a population of classifiers is maintained using a steady-state, niched GA. Each classifier includes a condition part which will match inputs with a certain specification for a subset of its variables. These classifiers are initialised with a high degree of specificity which may then generalise to match more inputs and map out the problem structure. Classifiers are assigned a fitness derived from the performance of the classifier relative to classifiers with which it overlaps and therefore competes with. This may include the parents of newly-generated classifiers.

The parallels between the idealised GA schema, LCS conditions and the modules of TEA are clear. They embody, in a ternary form, entities competing for representation within a limited medium. Their associated degree of weighting, either implicit or explicit, influences their current selection and ongoing persistence.

There are also key differences, not least of which is the difference between the task of classification and function optimisation.

Recent work has attempted to address the inadequacy of the standard GA operators for constructing and mixing building-blocks by taking an EDA approach [2]. This work investigates substituting mutation and recombination with adapted versions of the ECGA and BOA for overlapping and hierarchical problems respectively.

6.2 Hypotheses and Contributions

In this section, we consider the validity of the original motivation and hypotheses in light of the experimental studies. This allows us to confirm the key contributions from the thesis which have both a practical and a theoretical part.

From a practical standpoint we address the originality, significance and applicability of the TEA. For this, we recap on the novel combination of processes that TEA employs and seek to explain how these suit it to the classes of problems laid out. For the theoretical part, we outline possible lessons that can be learnt about an alternative paradigm for evolutionary optimisation based on variable units of selection.

We now restate the original hypotheses, concluding each one and noting its contribution.

6.2.1 Population as a Structured Sampling Model

Hypothesis *An elitist population of solutions, composed from multivariable components, can be sampled as an expressive and compact search model.*

Conclusions We investigated different strategies for population maintenance. Whilst a purely elitist strategy was adequate for unimodal problems, maintaining a population with a minimum size was required to ensure multiple optima could be retained indefinitely. A simple form of this was shown to be effective. Other population maintenance strategies are suggested in future work later in this chapter.

Sampling an individual from a large, diverse population can be highly exploratory but it can also be disruptive. We found that when the population has adapted to a set of optima based on large-order interactions, it becomes increasingly difficult to sample new viable individuals which preserve these. At this point, with relatively little turnover in the population, we find that adaptation of the genetic structure cannot efficiently be performed by individual selection.

Contribution The population comprises structured individuals as a self-contained model of positive fitness interactions, as well as fit elements. The weighting it assigns to all selectable components makes it capable of representing all ECGA distributions as well as those of the SEAM. Furthermore, TEA is capable of representing the competition of overlapping (or polyhierarchical) functions. A valuable by-product of explicit linkage modelling is that the resulting knowledge of the problem is in a form that is amenable for human interpretation and possible exporting to more customised search algorithms.

6.2.2 Transmutation for Structural adaptation

Hypothesis *Structural mutation acting at the individual level can, in principle, adapt the search model to reflect the genetic interactions of a problem.*

Conclusions By successfully adapting to represent the structure of modular problems (with a significant order) and hierarchical problems with a significant number of levels, the transmutation operator has shown itself to be capable of making the adaptations to the complexity from an entirely unstructured population model to one which reflects that of the problem. This is without any prior knowledge regarding the structure of the problem.

Contribution Adaptive representation has been identified as a requirement for scalable evolution but existing models are hindered by either computational complexity, overspecialisation or sensitivity to parameters. Transmutation provides a simple, intuitive yet effective

means for adapting the genetic structure when used in conjunction with the selective process of the next hypothesis.

The presented algorithm shares motivation and features with various other works, notably the weighted sampling with EDAs and the adapting pool of incremental partial solutions from the CSEAs. Nevertheless, the approach is self-supporting.

6.2.3 Structural Validation via Individual Selection

Hypothesis *Selection can efficiently amplify and distinguish modules, in parallel, which encapsulate valid building-blocks, and demote and remove invalid modules.*

Conclusions To be a reliable source of modules, an additional filter was required involving a simple statistical test; unconditional acceptance of transmutated modules was found to be incompatible for selective processing.

Once we raised the likelihood of candidate modules being genuine building-blocks then we found that we could accept modules with a relatively low degree of confidence and allow selection to perform the final validation. The filtered supply of candidate modules meant that individuals of a higher quality would be sampled and this maintained the turnover of individuals in the population to ensured selective processes were able to efficiently amplify or discard competing representations. The original demodularisation operation was found to be superfluous once the selective process was refined; low-quality modules were effectively selected against.

Contribution The extension of selection to entities within the individual is a powerful one given the conditions for an effective individual turnover. It allows structures resulting from transmutation to be reinforced and exploited in a straightforward way without the use of complex statistical testing and model building techniques. Such techniques would likely be able to enhance the algorithm further but they have been shown to be unnecessary for the desired modelling capabilities.

6.2.4 Intersection of Compositional and Probabilistic Models

Hypothesis *There exists a model which unifies the most significant elements of the CSEAs with those of the EDAs to obtain key advantages over both of these.*

Conclusions In chapter 3 we described a view of GA and EDA models that could employ explicit frequency and be extended to embody modular structures. A bridge is then built to encompass compositional models with a unified 'meta-algorithm'.

Staged testing in section 5.5 empirically supports generalised applicability of this algorithm.

Contribution Arbitrary hybrids of existing algorithms are not difficult to manufacture. The contribution of the TEA is in identifying the key properties required to solve the target problems and bringing them together without extraneous elements. The result is an algorithm that is simpler than its predecessors in many regards.

6.2.5 Notes on the Problem Form

The problem form that has been used is a general one, encompassing linear functions (e.g. ones-max), modular (e.g. NK networks, Royal Road, concatenated trap), hierarchical functions (e.g. H-IFF) and finally, a polyhierarchical function. We believe it to be a useful form for characterising problems, particularly as it makes explicit the structure of the problem.

It follows that we can have a search algorithm that is more efficient than enumeration when the fitness function can be decomposed to a set of independent sub-functions. Rather than use a lookup table for each possible set of values, we can linearly combine functions over subsets of the variables. This is the structure in the fitness function that allows us to concurrently optimise independent sets of variables.

It is important to note that problems do not need to expressed in the form used here in order to be tractable; they merely need to be implicitly expressible in such a form. We

believe that many real-world problems do fit into this category.

The similarity between the proposed problem representation and the representation in TEA is neither incidental nor restrictive. The framework is created to represent a difficult class of real-world problem and TEA to tackle this. The problem form is general enough to represent other problems and test other algorithms. Likewise, TEA is capable of solving other problems. Both however are capable of representing polyhierarchical structures. TEA has no special access to the structure of the dependencies within the black box function, just as the form is not presented with the genetic linkage information of TEA solutions. The challenge for TEA, as for all applicable algorithms, is to infer and exploit the problem settings and interactions.

6.3 Conclusion

This work addresses a fundamental issue in evolutionary optimisation, that of scalability, and presents a novel approach to tackling the issue in combinatorial optimisation problems. Previous analysis has concluded that an adaptive representation is key to allowing scalable adaptation in the form originally envisaged by the building-block hypothesis to proceed.

Inspiration for this work, which we aimed to apply to these classes of increasingly complex problems, was the evolutionary transition. Although inspired by complex natural systems and contemporary theories for their development and adaptation, the primary motivation is practical optimisation and does not extend to claims that the search model presented herein is an actual model of real adaptive systems. Nevertheless, the model is a highly abstract one with no features that are irreconcilable with natural system. If it therefore serves as a reciprocal inspiration to natural science models this would be a bonus.

Unlike other work which constructed the algorithm around a biologically-informed model of this phenomenon, we aimed to show how this could be abstracted into a generally applicable algorithm. This general model has aspects of probabilistic models with a frequency-

biased weighting, and compositional algorithms in an explicit, adaptive modular representation. This enables this compound model to be effective on problems which do not necessarily have a particular form.

The initial formation of a novel approach was guided by a reasoned set of principles distilled from a critical analysis and comparison of the relevant algorithms and concepts. This included a clear description of the type of problem we would address, considering complexity from a natural inspiration to a form fit for optimisation. The Transitional Evolutionary Algorithm then implemented the principle of frequency-biased selection of units on varying complexity.

The pertinent elements of TEA were verified on appropriate problems against the standard algorithms. Various insights were gained and used to revise the TEA as it was applied to state-of-the-art algorithms on their native problems. This structured set of experiments delineated the range of competence for these algorithms. Chapter 5 shows a reasoned and validated investigation from the simplest optimisation algorithm to one that addresses classes of problems known to be challenging for state-of-the-art algorithms.

TEA proved to be not only able to reliably identify and process modules, but also to do this efficiently enough to be competitive with the leading algorithms. This was without requiring separate model-building or tight hierarchy. It therefore demonstrates a prospective approach for adaptive scaling inspired by evolution.

Our search identifies not only sets of correlated variables but specific sets of values for those which appear to produce a significant positive fitness interaction. We are not interested in the distribution of non-optimal values or modules; the population is used solely to map out known optima. Also, we are not attempting to infer conditional dependencies but mutual positive associations.

The significance of this approach is broad. EAs are an established technique applied across a range of problem domains. The underlying processes of mutation, recombination with selection are taken as read for extended models of complex interactions, e.g. Artificial

Life. The model here only concerns the genetic representation and is entirely independent of any phenotypic expression. Therefore, any insight or refinement into these fundamental processes have consequences for any application built on them.

GAs are often adopted for their simplicity and EDAs for their expressiveness. TEA is significant in that it combines the advantage of both and is therefore valuable from a practical perspective. From a theoretical perspective, we have shown that the phenomenon of the evolutionary transition is a more promising nature-inspired approach than recombination across a range of optimisation problem types.

We now reflect on two questions, largely unaddressed in this thesis, which may be appropriate for further development.

6.3.1 Incorporation of Prior Knowledge

We take some time in section 1.4 to identify various types of prior domain knowledge that may exist for a problem function. Our reason for doing this is to ensure that they are not implicitly introduced into the search. Rather, the biases of genetic frequency and linkage are automatically acquired by the search as it progresses.

In actual applications, we may well have access to prior domain knowledge that we wish to exploit. The TEA model makes this application straightforward and robust as the pool can be primed with weightings and linkages. Modules embodying linkage could be added to the initial pool to genetically encode known building-blocks. Non-uniform weightings can be applied to these, along with with the initial set of alleles, according to their perceived quality.

Significantly, these seeding entities are subject to the same selective scrutiny that validates the algorithm-generated ones. This means that even weakly-supported linkages can be encoded and their viability directly tested by their persistence in the population. Despite this, some general problem classes may not be naturally suited to the TEA approach and are considered below.

6.3.2 General Applicability of TEA

The population maintenance strategy of the TEA is intended to retain a diverse set of the fittest solutions so far encountered. This is reliant on a relatively consistent fitness evaluation of its members. A noisy function that samples the fitness of a solution from a distribution might well invalidate this presumption. Likewise, a dynamic function, perhaps arising from coevolutionary dynamics, may impact the performance of the TEA. In both these case, some individuals can be accepted on the basis of a fortunate evaluation as others are rejected via unfortunate ones. Generational population models are less prone to such effects with their lower levels of persistence.

6.4 Future Work

There are several ways this model could be extended, many of which have already been cursorily investigated. The algorithm described represents a base model which is simple to understand, implement and apply and is ripe for further development. A number of the more promising ones are outlined below.

6.4.1 Delineating Limitations of TEA

We encountered a possible pitfall with the TEA for higher order H-IFF problems where some sets of interaction variables became fixated at non-context-optimal values. It is not clear as yet whether a larger population size or alternative diversity-emphasising population maintenance strategy would resolve this — although this is believed to be the case. Useful further work would be to analyse the bounds of successful TEA operation in more detail and possibly demonstrate way to extend them.

6.4.2 Nature-Inspired Interaction Inference

Transmutation is currently conditional on a simple statistical test. The criterion for transmutation is statistical but simple enough that a nature-inspired metaphor based on competition could be conceived to perform the same role using a niching method.

6.4.3 Fitness-weighted Modelling

There is currently no effect of a fitness-based weighting and we straightforwardly model the population frequencies. As an extension, we also consider alternative variations to the population strategy. Specifically, if we allow a population with a range of fitness values (rather than strict elitism) then we can weight via these and not solely on frequency.

6.4.4 Implementation Optimisations

The current implementation evolved to be flexible and general for easier inspection of algorithmic variants and comparative tests with other algorithms. A fresh implementation, written with efficiency as a priority, would be expected to be able to make significant gains in that respect. This might include a caching of previously computed properties of the population for periods where it is unchanged. The ECGA employs such a system as part of its model-building phase and this is believed to be responsible making it an effective algorithm.

The search model used here, being represented by a population of structured individuals, is also highly amenable to being split across multiple systems and later merged. This would be valuable for problems for which the population tends to converge to local optima.

6.4.5 Other Representations

We are in no way restricted to selecting bits or bitsets as entities. We could instead, for example, be selecting subroutes (i.e. TSP). In fact, a particular rule for weighting update

is believed to result in something like the ACO technique [52].

6.4.6 Other Interpretations

The motivation for this work has been scalable evolutionary optimisation and the inspiration the evolutionary transition. However, the algorithm that has been shaped has, in many respects, taken on the form of a dynamic probabilistic network model. A reinterpretation of TEA, or something similar, exclusively in these terms might well prove worthwhile.

Bibliography

[1] S. Baluja. Population-Based Incremental Learning: A method for integrating genetic search based function optimization and competitive learning. Technical report, School of Computer Science, Carnegie Mellon University, Pittsburgh, PA, 1994.

[2] M.V. Butz, M. Pelikan, X. Llorà, and D.E. Goldberg. Automated global structure extraction for effective local building block processing in XCS. Technical Report 2005011, Illinois Genetic Algorithms Laboratory (IlliGAL), 2005.

[3] Charles Darwin. *The life of Erasmus Darwin.* Cambridge University Press, first unabridged edition, 2003. Edited by Desmond King-Hele.

[4] R. Dawkins. *The Extended Phenotype: The Long Reach of the Gene.* Oxford University Press, 1983.

[5] E.D. De Jong, Dirk Thierens, and Richard A. Watson. Hierarchical genetic algorithms. In *Proceedings of the 8th International Conference on Parallel Problem Solving from Nature*, pages 232–241, 2004.

[6] E.D. De Jong, R.A. Watson, and D. Thierens. A generator for Hierarchical Problems. In *Proceedings of the GECCO Workshop on the Theory of Representations.* ACM Press, 2005.

[7] K.A. De Jong and W.M. Spears. A formal analysis of the role of multi-point crossover in genetic algorithms. *Annals of Mathematics and Artificial Intelligence*, 5(1):1–26, 1992.

[8] A. Defaweux. *Evolutionary Transitions as a Metaphor for Compositional Search: Definition and Evaluation of a New Optimisation Algorithm.* PhD thesis, Vrije Universiteit Brussel, Brussel, April 2006.

[9] Gerald Edelman. *Bright Air, Brilliant Fire.* Penguin, 1994.

[10] R. Etxeberria and P. Larrañaga. Global optimization using bayesian networks. In *Second Symposium on Artificial Intelligence (CIMAF-99)*, pages 332–339, 1999.

[11] S. Forrest and M. Mitchell. Relative building-block fitness and the building block hypothesis. In D. Whitley, editor, *Foundations of Genetic Algorithms 2*, pages 109–126, San Mateo, CA, 1993. Morgan Kaufmann.

[12] D. Goldberg. *Genetic Algorithms in Search Optimization and Machine Learning.* Addison Wesley, Reading, MA, 1989.

[13] D.E. Goldberg, B. Korb, and K. Deb. Messy genetic algorithms: Motivation, analysis, and first results. *Complex Systems*, 5(3):493–530, October 1989.

[14] G. Harik. *Learning linkage to efficiently solve problems of bounded difficulty using genetic algorithms.* PhD thesis, Dept. Computer Science, University of Michigan, Ann Arbour, 1997.

[15] G. Harik. Linkage learning via probabilistic modeling in the eCGA. Technical Report 99010, Illinois Genetic Algorithms Laboratory, January 1999.

[16] G.R. Harik, F.G. Lobo, and D.E. Goldberg. The compact genetic algorithm. *IEEE Transactions on Evolutionary Computation*, 3(4):287–297, 1999.

[17] Hinterding, Michalewicz, and Eiben. Adaptation in evolutionary computation: A survey. In *IEEECEP: Proceedings of The IEEE Conference on Evolutionary Computation, IEEE World Congress on Computational Intelligence*, 1997.

[18] J.H. Holland. *Adaptation in Natural and Artificial Systems.* University of Michigan Press, 1975.

[19] E.D. De Jong, Richard A. Watson, and Dirk Thierens. On the complexity of hierarchical problem solving. In *Proceedings of the Genetic and Evolutionary Computation Conference.* IEEE, 2005.

[20] H. Kargupta. Gene expression: The missing link in evolutionary computation. In D. Quagliarella, J. Périaux, C. Poloni, and G. Winter, editors, *Genetic Algorithms in Engineering and Computer Science*, chapter 4. John Wiley & Sons Ltd, 1997.

[21] S.A. Kauffmann. Adaptation on rugged fitness landscapes. In D. Sein, editor, *Lectures in the Sciences of Complexity*, volume I of *SFI Studies in the Sciences of Complexity*, pages 527–618. Addison-Wesley, Redwood City, 1989.

[22] S. Kullback and R.A. Leibler. On information and sufficiency. *Annals of Mathematical Statistics*, 22:79–86, 1951.

[23] P. Larrañaga and J.A. Lozano. *Estimation of Distribution Algorithms: A New Tool for Evolutionary Computation.* Kluwer Academic, Boston, MA, 2001.

[24] T. Lenaerts. *Different Levels of Selection in Artificial Evolutionary Systems: Analysis and Simulation of Selection Dynamics.* PhD thesis, Department of Computer Science, Vrije Universiteit Brussel, Pleinlaan 2, 1050 Brussel, 2003.

[25] Richard Lewontin. *The Triple Helix: Gene, Organism and Environment.* Harvard University Press, 2001.

[26] X. Llorà, K. Sastry, D.E. Goldberg, and L. de la Ossa. The χ-ary extended compact classifier system: Linkage learning in Pittsburgh LCS. Technical Report 2006015, Illinois Genetic Algorithms Laboratory (IlliGAL), April 2006.

[27] F.G. Lobo. The parameter-less genetic algorithm: Rational and automated parameter selection for simplified genetic algorithm operation. Technical report, University of Illinois at Urbana-Champaign, July 2000.

[28] S. Mahfoud. *Niching Methods for Genetic Algorithms.* PhD thesis, Dept. General Engineering, University of Illinois, 1995.

[29] L. Margulis. *Origin of Eukaryotic Cells.* Yale University Press, 1970.

[30] J.M. Maynard-Smith and E. Szethmary. *The Major Transitions in Evolution.* WH Freeman, 1995.

[31] M. Mitchell and S. Forrest. Fitness landscapes: Royal road functions. In T. Bäck, D. Fogel, and Z. Michalewicz, editors, *Handbook of Evolutionary Computation.* Oxford University Press, 1997.

[32] M. Mitchell, S. Forrest, and J.H. Holland. The royal road for genetic algorithms: Fitness landscapes and GA performance. In F.J. Varela and P. Bourgine, editors, *Toward a Practice of Autonomous Systems: Proceedings of the First European Conference on Artificial Life*, pages 245–254, Cambridge, MA, 1992. MIT Press.

[33] H. Mülenbein, J. Bendisch, and H.-M. Voigt. From recombination of genes to the estimation of distributions. i. binary parameters. In *Proceedings of PPSN-I, First International Conference on Parallel Problem Solving from Nature*, pages 178–187, Berlin, Germany, 1996. Springer.

[34] M. Pelikan, D. Goldberg, and E. Cantú-Paz. BOA: The Bayesian optimization algorithm. In Wolfgang Banzhaf, Jason Daida, Agoston E. Eiben, Max H. Garzon, Vasant

Honavar, Mark Jakiela, and Robert E. Smith, editors, *Proceedings of the Genetic and Evolutionary Computation Conference GECCO-99*, volume I, pages 525–532, Orlando, FL, 13-17 1999. Morgan Kaufmann Publishers, San Fransisco, CA.

[35] M. Pelikan, D.E. Goldberg, J. Ocenasek, and S. Trebst. Robust and scalable black-box optimization, hierarchy, and ising spin glasses. Unpublished.

[36] M. Pelikan and H. Mühlenbein. The bivariate marginal distribution algorithm. In R. Roy, t. Furuhashi, and P.K. Chawdhry, editors, *Advances in soft computing - Engineering Design and manufacturing*, pages 521–535, London, 1999. Springer-Verlag.

[37] Martin Pelikan. *Hierarchical Bayesian Optimization Algorithm: Toward a new generation of evolutionary algorithms*. Springer-Verlag, 2005.

[38] Martin Pelikan, David Goldberg, and Fernando Lobo. A survey of optimization by building and using probabilistic models. Technical Report 99018, IlliGAL, September 1999.

[39] A. Penn. Steps towards a quantitive analysis of individuality and its maintenance: A case study with multi-agent systems. In D. Polani, J. Kim, and T. Martinez, editors, *Fifth German Workshop on Artificial Life: Abstracting and Synthesizing the Principles of Living Systems*, pages 125–134. IOS Press, 2002.

[40] I. Rechenberg. *Evolutionnstrategie: Optimierung technischer Syseme nach Prinzipien der biologischen Evolution*. Friedrich Frommann Verlag, Stuttgart, 1973.

[41] H.P. Schwefel. *Evolutionasstrategie undnumerische optimierung*. PhD thesis, Technical University of Berlin, Department of Process Engineering, 1975.

[42] H.A. Simon. *The sciences of the artificial*. MIT Press, Cambridge, MA, 1968.

[43] W. Spears and K. De Jong. On the virtues of parameterized uniform crossover. In Rick Belew and Lashon Booker, editors, *Proceedings of the Fourth International Conference on Genetic Algorithms*, pages 230–236, San Mateo, CA, 1991. Morgan Kaufman.

[44] G. Syswerda. Uniform crossover in genetic algorithms. In *The Proceedings of the Third International Conference on Genetic Algorithms(ICGA-89)*, pages 2–9, San Mateo, CA, 1989. Morgan Kaufmann.

[45] G. Syswerda. Simulated crossover in genetic algorithms. In *Foundations of Genetic Algorithms*, volume 2, pages 889–914. Morgan Kaufmann, San Mateo, CA, 1993.

[46] D. Thierens. Scalability problems of simple genetic algorithms. *Evolutionary Computation*, 7(4):331–352, 1999.

[47] R. Watson and B. Pollack. Modular interdependency in complex dynamical systems. *Artificial Life*, 11(4):445–457, 2005.

[48] R. Watson and J. Pollack. A computational model of symbiotic composition in evolutionary transactions. *Biosystems, Special Issue on Evolvability*, 2002.

[49] R.A. Watson. *Compositional Evolution: Interdisciplinary Investigations in Evolvability, Modularity, and Symbiosis.* PhD thesis, Brandeis University, 2002.

[50] R.A. Watson, G.S. Hornby, and J.B. Pollack. Modelling building-block interdependency. In *PPSN-V*, 1998.

[51] S.W. Wilson. Classifier fitness based on accuracy. *Evolutionary Computation*, 3(2):149–175, 1995.

[52] M. Zlochin, M. Birattari, N. Meuleau, and M. Dorigo. Model-based search for combinatorial optimization: A critical survey. *Annals of Operations Research*, 131:373–395, 2004.