

---

# Empirical Modelling of Genetic Algorithms

**Richard Myers**

Department of Computer Science, University of York, York, Y01 5DD, UK.

rich@cs.york.ac.uk

**Edwin R. Hancock**

Department of Computer Science, University of York, York, Y01 5DD, UK.

erh@cs.york.ac.uk

---

## Abstract

This paper addresses the problem of reliably setting genetic algorithm parameters for consistent labelling problems. Genetic algorithm parameters are notoriously difficult to determine. This paper proposes a robust empirical framework, based on the analysis of factorial experiments. The use of a graeco-latin square permits an initial study of a wide range of parameter settings. This is followed by fully crossed factorial experiments with narrower ranges, which allow detailed analysis by logistic regression. The empirical models thus derived can be used first to determine optimal algorithm parameters, and second to shed light on interactions between the parameters and their relative importance. The initial models do not extrapolate well. However, an advantage of this approach is that the modelling process is under the control of the experimenter, and is hence very flexible. Refined models are produced, which are shown to be robust under extrapolation to up to triple the problem size.

## Keywords

Genetic algorithms Empirical models Factorial experiments Constraint satisfaction Line labelling

## 1 Introduction

Genetic algorithms are stochastic global optimisation methods based on the concept of Darwinian evolution in populations (HollandHolland1975; GoldbergGoldberg1989; FogelFogel1994; MitchellMitchell1996). Evolutionary algorithms have been proposed independently by several authors (FraserFraser1957; BremermannBremermann1958; Reed, Toombs, and BarricelliReed et al.1967; HollandHolland1975; RechenbergRechenberg1973; SchwefelSchwefel1981). It is Holland's formulation in (HollandHolland1975) which is regarded as the standard for genetic algorithms, although there are others, e.g. the evolutionary strategies (SchwefelSchwefel1981). Such algorithms have applications in many disciplines: among the more famous are artificial life (Jefferson, Collins, Cooper, Dyer, Flowers, Korf, Taylor, and WangJefferson et al.1991) and evolutionary programming (KozaKoza1992). In the field of computer vision, genetic algorithms have been used for a host of applications, including image segmentation (Andrey and TarrouxAndrey and Tarroux1994), object recognition (TsangTsang1997), stereo matching (Saito and MoriSaito and Mori1995), edge extraction (Bhandarkar, Zhang, and PotterBhandarkar et al.1994), and graph matching (Cross, Wilson, and HancockCross et al.1997).

There are many factors influencing the behaviour of the algorithm. These can be classified as problem specific or algorithm specific. The problem specific factors are:

- the nature of the problem itself,

- the form of the fitness function, and
- whether or not there exists an effective local search procedure with which to augment the algorithm.  
Algorithm specific factors include:
  - the particular (genetic) algorithm used,
  - whether or not it has a local search step,
  - which crossover operator is used,
  - which selection operator is used,
  - the population size,
  - the terminating criterion,
  - the crossover rate, and
  - the mutation rate.

Any additional procedures, such as fitness sharing (Goldberg and RichardsonGoldberg and Richardson1987) or otherwise constraining the selection operator (GoldbergGoldberg1989), may increase the number of parameters. The result of this complexity is that applying a genetic algorithm to a particular problem, whilst simple in principle, can be very difficult in practice (DavisDavis1991). This problem can be addressed by theoretical modelling, empirical modelling, or ad-hoc experimental approaches to parameter setting, as described in the following paragraphs.

Holland's original text (HollandHolland1975) proposed the "schema theorem", which predicted the rapid convergence of the algorithm. Despite its weaknesses (GrefenstetteGrefenstette1993; MühlenbeinMühlenbein1994), it is still widely appealed to as evidenced by the paper of (Srinivas and PatnaikSrinivas and Patnaik1994). More rigorous analyses have been attempted by, among others, Rudolph, Vose, and Qi and Palmieri (RudolphRudolph1997; VoseVose1995; Qi and PalmieriQi and Palmieri1994a; Qi and PalmieriQi and Palmieri1994b). These models usually require simplifying assumptions such as infinite population sizes or infinite time, or involve large matrix calculations (MitchellMitchell1996). There has also been interest in applying mathematical models used in the study of biological genetics to evolutionary optimisation. Examples include the work of Mühlenbein in (MühlenbeinMühlenbein1994; Mühlenbein and Schlierkamp-VoosenMühlenbein and Schlierkamp-Voosen1995) and Bedau in (BedauBedau1995). While general theoretical models are beneficial in understanding the processes at play in genetic algorithms, their use in practical settings is limited. Since these models impose few constraints on the algorithm they are not particularly informative as to what the best algorithm configuration might be for a given problem. In view of the factors listed in the previous paragraph, any theoretical model which predicts parameter values should also take the nature of the problem into account. By taking advantage of specific features of the problem to be solved, Cross, Myers and Hancock (Cross, Myers, and HancockCross et al.2000), and Prügel-Bennett and Shapiro (Prügel-Bennett and ShapiroPrügel-Bennett and Shapiro1994) have recently been able to make better quantitative predictions about the behaviour of the algorithm.

The alternative to theoretical modelling is empirical modelling. Possibly because of the computational expense, very few substantial experimental studies of genetic algorithms have been undertaken. The earliest was by DeJong (DeJongDeJong1975),

who considered a suite of five numerical optimisation problems, at least three of which are relatively easy (Whitley, Beveridge, Graves, and MathiasWhitley et al.1995). Problems from DeJong's test suite are still used to test and compare genetic algorithms, an approach strongly criticised by Whitley and others in (Whitley, Beveridge, Graves, and MathiasWhitley et al.1995). In (Schaffer, Caruna, Eshelman, and DasSchaffer et al.1989), Schaffer et al. presented what appears to be the only significant large scale experimental study of genetic algorithms to date. However, their experiments were mostly based on numerical test problems and have been shown by Mühlenbein in (MühlenbeinMühlenbein1994) to be unreliable under extrapolation. This is not surprising since Schaffer et al. (Schaffer, Caruna, Eshelman, and DasSchaffer et al.1989) based their models on problems drawn from different domains including both numerical and combinatoric optimisation. It is important to stress that the extrapolation of a regression model within the same problem domain should be undertaken only with caution. Moreover, the extrapolation of the model outside the problem domain is often totally meaningless.

There have been several attempts to determine genetic algorithm parameters experimentally without modelling (GrefenstetteGrefenstette1986; BramletteBramlette1991; DavisDavis1989; DavisDavis1991). These attempts have all used genetic algorithms to optimise the parameters in a bootstrapping framework. For instance, Grefenstette have cast parameter-setting as combinatoric optimisation, and have used a meta-level genetic algorithm (GrefenstetteGrefenstette1986). Unfortunately, this approach begs two questions. The first is what control parameters would be appropriate for such a "control parameter meta-optimiser"? The second is how can the results be generalised so that they apply to more than one problem instance? Grefenstette's study has the additional weakness of comparing the outcomes of several random processes (the genetic algorithms) on the basis of only very small sample sizes (1 sample in most cases, 5 samples at best). Davis's work involved adapting the probabilities of the various genetic operators on the basis of how successful previous applications of each operator had been in the program run (DavisDavis1989; DavisDavis1991). However, it is not clear whether the parameter settings can respond fast enough to the changing population (MitchellMitchell1996). Davis's approach also requires a formula for operator fitness, which implies additional parameterisation of the algorithm. One area which does not appear to have received much attention in the literature is the use of supervised or unsupervised learning techniques to set algorithm parameters. Such "black-box" approaches are likely to present problems when attempts to generalise them are made. In this paper, we take the view that there is no substitute for empirical modelling when seeking to determine appropriate values for algorithm parameters. However, there is no particular reason to suppose that a genetic algorithm will behave in the same way for different problem classes, e.g. continuous numerical optimisation vs. (discrete) combinatoric optimisation. Moreover, there is also no reason to suppose that parameters which work for numerical optimisation problems will also work for symbolic ones.

The main novel contribution in this paper is the development of specific, empirical models of genetic algorithm performance for consistent labelling problems. The labelling problem studied in detail is line labelling, which has attracted research interest in computer vision for the last twenty-five years (HuffmanHuffman1971; ClowesClowes1971; WaltzWaltz1975; Lipson and ShpitalniLipson and Shpitalni1996). The empirical models are based on factor analysis. This is a mature statistical technique which has been applied to difficult problems in many disciplines over the last 50

to 70 years. Our specific aim is to use factor analysis to determine the best combination of genetic algorithm parameters for consistent labelling problems. We fit generalised linear models (McCullagh and NelderMcCullagh and Nelder1989) to the data. The advantage of generalised linear models is not only that they can be used to determine optimal parameter settings, but also that they shed light on the relative importance of the different parameters and their interactions. Another useful feature is that the experimenter has total control over the form of the model. This is not generally true of black-box methods. The empirical models which result from our factor analysis (Cochran and CoxCochran and Cox1957) summarise the outcome of some 500,000 program runs. The empirical models are shown to interpolate and extrapolate well. As a result, we argue for the use of factorial experiments as a general tool for genetic algorithm parameter selection. The experimental protocol described in this paper should be extensible to other constraint satisfaction problems.

An additional motivation for this paper is that consistent labelling encompasses problems ranging from the trivial to the intractable. Ackley's ONEMAX problem (AckleyAckley1987) would be solved by the hybrid genetic algorithm used in this study in a single iteration with a population size of 1. Line labelling is harder, but still relatively easy. Graph matching is more difficult still. General satisfiability would present a serious challenge to the algorithm. It is therefore not unreasonable to suppose that, because labelling problems have been so extensively studied (MackworthMackworth1977; Haralick and ShapiroHaralick and Shapiro1979; Haralick and ShapiroHaralick and Shapiro1980; Haralick, Davis, and RosenfeldHaralick et al.1978; NudelNudel1983; JeavonsJeavons1998), they might provide a realistic framework in which to further investigate genetic algorithm behaviour.

The outline of this paper is as follows. The next section discusses consistent labelling, and introduces and formulates the line labelling problem. Section 3 describes the experimental design and methods. Section 4 analyses the results of full factorial experiments by fitting generalised linear models. Finally, section 5 presents some conclusions and suggests future work.

## 2 Consistent Labelling

The consistent labelling problem is a discrete constraint satisfaction problem in which the goal is to make an assignment from a set of labels to a set of objects, subject to constraints which exist between (subsets of) these objects. It is formally equivalent to the general satisfiability problem and is NP-complete in its general form (NudelNudel1983; Garey and JohnsonGarey and Johnson1979; CookCook1971; Haralick, Davis, and RosenfeldHaralick et al.1978), although many special cases exist. Consistent labelling problems were studied intensively in the computer vision literature of the mid to late 1970s (MackworthMackworth1977; Haralick and ShapiroHaralick and Shapiro1979; Haralick and ShapiroHaralick and Shapiro1980). Examples of consistent labelling problems include graph colouring, subgraph isomorphism, the Boolean satisfiability problem and scene labelling.

Early solutions to this problem involved searching a configuration space, which can be pruned using discrete relaxation operators (MackworthMackworth1977; Mackworth and FreuderMackworth and Freuder1985; WaltzWaltz1975). However, these methods did not use evidence derived from the scene, relying merely on pre-defined constraint relations. To overcome this problem, Rosenfeld, Hummel and Zucker formulated probabilistic relaxation in (Rosenfeld, Hummel, and ZuckerRosenfeld et al.1976). Much of the work involving consistent labelling has adopted Hummel and Zucker's

optimisation paradigm (Hummel and ZuckerHummel and Zucker1983; Faugeras and BerthodFaugeras and Berthod1981; LloydLloyd1983; Mohammed, Hummel, and ZuckerMohammed et al.1983; Wilson and HancockWilson and Hancock1997). The problem is to find a set of label assignments which optimises some global consistency measure. This is typically done by iteratively applying a local operator to the solution until no further improvement can be made. The most straightforward optimisation technique is gradient ascent, in which the update operator is required to monotonically increase the quality of the solution: this was the approach used by Hummel and Zucker in their original work (Hummel and ZuckerHummel and Zucker1983), and by many others (Hancock and KittlerHancock and Kittler1990a; Faugeras and Berthod-Faugeras and Berthod1981; LloydLloyd1983; Mohammed, Hummel, and ZuckerMohammed et al.1983; Wilson and HancockWilson and Hancock1997). Gradient ascent is appropriate when an initial guess can be made, which is close to the final solution in the sense that there are no intervening local optima. This is not always the case, so it is often preferable to use global optimisation techniques such as simulated annealing (Kirkpatrick, Gelatt, and VecchiKirkpatrick et al.1983; Geman and GemanGeman and Geman1984), mean field annealing (Geiger and GirosiGeiger and Girosi1991; Yuille and KosowskyYuille and Kosowsky1994) or genetic search (HollandHolland1975).

## 2.1 Line Labelling

Line drawing interpretation has been an active area of investigation in machine vision for over twenty-five years. It was independently formulated by Huffman and by Clowes in the mid 1970s (HuffmanHuffman1971; ClowesClowes1971). In fact it was this work which led Waltz to his seminal discrete relaxation algorithm (HuffmanHuffman1971; ClowesClowes1971; WaltzWaltz1975). Waltz's contribution was to show how a dictionary of geometrically permissible junction labellings could be used in an efficient search for consistent interpretations of polyhedral objects. The interpretation of line drawings has applications in, among other areas, document analysis, processing architects' sketches, and automatic interpretation of engineering drawings. A good example is provided in the paper of (Lipson and ShpitalniLipson and Shpitalni1996).

Trihedral polyhedra are solid shapes whose vertices may involve up to three faces. Thus drawings representing scenes composed of trihedral polyhedra only contain four types of line junction. The junctions can be classified according to whether they have ELL, TEE, FORK or ARROW shaped topologies. Each line in the drawing must be labelled according to whether it represents the concave intersection of two surfaces, (−), the convex intersection of two surfaces, (+), or an occluding boundary, (← and →) (the occluding surface is always to the right as one follows the arrow). An example is given in figure 1. In (WaltzWaltz1975), Waltz associated a dictionary of consistent label configurations with each junction type. These dictionaries are derived from the geometric constraints on the projection of three-dimensional scenes onto two-dimensional planes. Thus, the problem of finding a plausible three-dimensional interpretation of a line drawing can be solved by finding a consistent labelling of the lines in the drawing.

## 2.2 Formulation

The consistent labelling problem can be formulated as follows. Given a set of objects,  $\mathbf{V}$ , and a set of labels,  $\mathbf{A}$ , a set of neighbourhoods,  $\mathbf{C}$ , can be constructed. Each element,  $C_j$ , of  $\mathbf{C}$  is defined as  $C_j = \langle i \mid i \in \mathbf{V} \wedge \mathbf{Conn}(j, i) \rangle$ , where  $\mathbf{Conn}$  is the connectivity relation which is reflexive but not necessarily symmetric or transitive. A labelling,  $\gamma_j$ , of the  $j^{\text{th}}$  neighbourhood is a list of labels applied to its constituent objects,  $\gamma_j \subset C_j \times \mathbf{A}$ .

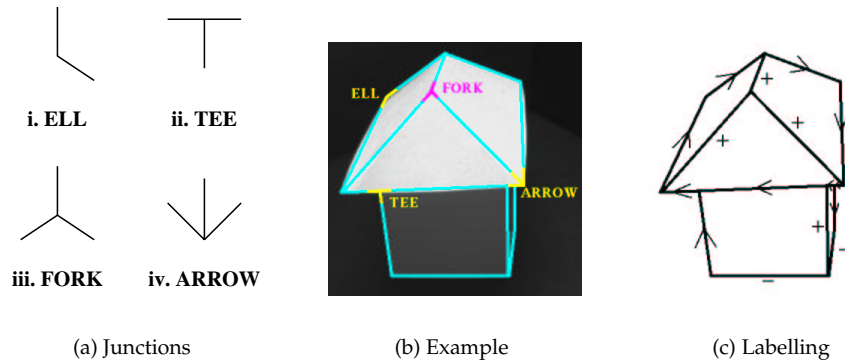


Figure 1: The Four Junction Types Defined by Huffman in (HuffmanHuffman1971).

Each neighbourhood has a dictionary,  $\Theta_j$ , of legal labellings. The dictionaries capture the structure of the problem.<sup>1</sup> Waltz filtering (WaltzWaltz1975) in this context would remove globally inconsistent labellings from the dictionaries,  $\Theta_j$ . The consistency of a labelling of all the objects, i.e. the global configuration of line-label assignments  $\Gamma \subset V \times \Lambda$ , can be determined by considering separately the consistency of the labellings of the neighbourhoods,  $\Gamma_j$ . Table 1 illustrates this formulation in the case of line labelling.

Consistent Labelling	Line Labelling
$V$	The set of lines in the drawing
$\Lambda$	The labels, +, -, $\leftarrow$ and $\rightarrow$
$C$	The set of junctions in the drawing
$\gamma_j$	A labelling of the $j^{\text{th}}$ junction
$\Theta_j$	The (Waltz) dictionary of the $j^{\text{th}}$ junction
$\Gamma$	A labelling of the entire drawing

Table 1: The formulation as it applies to line labelling.

### 2.2.1 Probabilistic Labelling Criterion

Hancock and Kittler have built on the work of Faugeras and Berthod (Faugeras and BerthodFaugeras and Berthod1981) and Hummel and Zucker (Hummel and ZuckerHummel and Zucker1983) by developing a probabilistic framework for measuring consistency (Hancock and KittlerHancock and Kittler1990a). This framework couples an explicit dictionary representation of constraints, as adopted by Waltz (WaltzWaltz1975), with a probabilistic model of the label corruption process. The corruption model is based on the premise that an initially consistent labelling is subject to the action of a memoryless stochastic label corruption process, which yields the current label configuration. The resulting consistency criterion expresses the quality of a labelling as a probability, which can be combined naturally with measurements made on the scene. This

<sup>1</sup>If all the neighbourhoods are the same size, this is equivalent to Haralick and Shapiro’s original formulation of the consistent labelling problem as a network constraint satisfaction problem in (Haralick and ShapiroHaralick and Shapiro1979; Haralick and ShapiroHaralick and Shapiro1980). The neighbourhoods are the “unit-constraint relation” and the dictionaries are the “unit-label constraint relation”.

allows both statistical and structural considerations to influence the labelling process. Scene interpretation is achieved by finding the label configuration which optimises the combined probability criterion. This was originally done in (Hancock and KittlerHancock and Kittler1990a) by gradient ascent. This approach has been applied to scene labelling (Hancock and KittlerHancock and Kittler1990a), edge detection (Hancock and KittlerHancock and Kittler1990b), graph matching (Wilson and HancockWilson and Hancock1997), and line labelling (HancockHancock1994).

Hancock and Kittler have developed a label-error process which can be used to compute the probability of an observed label configuration  $\gamma_j$  given a dictionary of permissible labellings of the same configuration of objects. Suppose that  $S_i$  is the  $i$ th configuration of permissible labels in the dictionary  $\Theta_j$  of the junction indexed  $j$ . The model commences from the assumption that the label errors occur at different line sites in a junction are independent of one-another and occur with a memoryless error probability  $P_e$ . As a result of this assumed model, the similarity of the label configuration  $\gamma_j$  and the dictionary item  $S_i$  is gauged by their Hamming distance  $H(\gamma_j, S_i)$ . The Hamming distance counts the number of label differences at the corresponding line-site for the current label configuration  $\gamma_j$  and the junction dictionary item  $S_i$ . Using the Bayes theorem, Hancock and Kittler show that the probability of observing the label configuration  $\gamma_j$  for the junction with label-configuration dictionary  $\Theta_j$  is given by

$$P(\gamma_j) = \frac{K_{C_j}}{|\Theta_j|} \sum_{S_i \in \Theta_j} \exp[-k_e H(\gamma_j, S_i)] \quad (1)$$

where  $K_{C_j} = (1 - P_e)^{|\gamma_j|}$  and  $k_e = \ln\left(\frac{1-P_e}{P_e}\right)$ . When  $P_e = 0.5$ ,  $k_e = 0$  and all labellings are equiprobable. As  $P_e \rightarrow 0$ ,  $k_e \rightarrow \infty$ , and only consistent configurations will have non-zero probabilities. Thus the constraints can be iteratively hardened by decreasing the label error probability,  $P_e$ , as the labelling algorithm (gradient ascent) proceeds (Hancock and KittlerHancock and Kittler1990a; Wilson and HancockWilson and Hancock1997). This is analogous to the annealing of the temperature parameter in simulated annealing. The best strategy has been found to reduce  $P_e$  by a constant factor with time.

To obtain a global measure of label consistency,  $\Gamma$ , Hancock and Kittler compute the arithmetic mean of junction label configuration probabilities appearing in equation 1 over all the junctions in the scene (i.e. the local line neighbourhoods). The resulting measure of label consistency is

$$P_A(\Gamma) = \frac{1}{|\mathbf{V}|} \sum_{j \in \mathbf{V}} P(\gamma_j) \quad (2)$$

### 2.2.2 Minimum-Distance Criterion

An alternative to Hancock and Kittler's approach is to take the geometric mean of the probabilities of the local label configuration residing on the different junctions as a measure of the consistency of the global label configuration. Taking the geometric mean of  $P(\gamma_j)$  over the different junctions gives the following measure of label consistency:

$$P_G(\Gamma) = \left( \prod_{j \in \mathbf{V}} P(\gamma_j) \right)^{\frac{1}{|\mathbf{V}|}} \quad (3)$$

For small values of  $P_e$ , the sum of exponentials appearing in Equation 1 is dominated by the dictionary item of minimum Hamming distance. As a result we can make the approximation

$$P(\gamma_j) \simeq \frac{K_{C_j}}{|\Theta_j|} \exp \left[ -k_e \min_{S_i \in \Theta_j} H(\gamma_j, S_i) \right] \quad (4)$$

With this approximation, the geometric mean of the junction label-configuration probability is given by

$$P_G(\Gamma) \approx k_v \exp \left[ -\frac{k_e}{|\mathbf{V}|} E(\Gamma) \right] \quad (5)$$

where  $k_v = \left( \prod_{j \in \mathbf{V}} \frac{K_{C_j}}{|\Theta_j|} \right)^{\frac{1}{|\mathbf{V}|}}$ , and  $E(\Gamma)$  is the sum of the minimum Hamming distances over the different junctions, i.e.

$$E(\Gamma) = \sum_{j \in \mathbf{V}} \min_{S_i \in \Theta_j} H(\gamma_j, S_i) \quad (6)$$

This label consistency criterion is appealing since it depends only on the closest dictionary item to the current label configuration. The reason for this is that  $E(\Gamma)$  is the number of local label errors or inconsistencies in the global label configuration  $\Gamma$  and therefore provides a more direct measure of consistency than the arithmetic mean of the junction label configuration probability appearing in equation 2. The approximation leading to the geometric mean label consistency criterion appearing in equation 5 will be poor when both  $k_e$  and  $H(\gamma_j, S_i)$  are small. Fortunately, this is unlikely to be the case: at the start of the labelling process,  $k_e$  may be small but  $H(\gamma_j, S_i)$  is likely to be large since the initial labelling will probably have many inconsistencies. By the end of the labelling process, when  $H(\gamma_j, S_i)$  becomes small,  $k_e$  will be large because of the annealing of the parameter  $P_e$ .

### 3 Experimental Protocol

To avoid confusion, the term “parameter” is used in the statistical sense for the remainder of this paper. Genetic algorithm “parameters” will be referred to as “control variables”. In addition to the four “standard” genetic algorithm control variables, population size, crossover rate, mutation rate and iteration limit, this section considers the effects of the following factors on the algorithm performance for several different line labelling problems

- probabilistic vs. minimum-distance labelling criteria (equations 2 and 6 respectively),
- uniform, one point, two point and half-uniform (EshelmanEshelman1991) crossovers, and
- the presence of a local search step

The genetic algorithm with a local search step will hereafter be referred to as the “hybrid algorithm”. The local search step consisted of forcing each individual in the population to a local optimum using a deterministic update procedure prior to performing selection. These effects cannot be studied in isolation: two-point crossover might work best in a standard genetic algorithm, but uniform might be preferable when there is a local search step. In short, parameters which are likely to interact should be

studied together in a factorial framework. This study will develop analytical empirical models relating algorithm performance to the control variable settings and other factors. One is generally more interested in inverting these models to determine what control variable settings will lead to a given outcome. This is by no means an easy problem in this case, since the models will be multidimensional. Nevertheless, given an analytical model, numerical optimisation techniques can be brought to bear. The goals of this study are:

1. to establish a model of the genetic algorithm's optimal performance.
2. to establish how useful the model is in setting control variables, i.e. how robust it is with respect to interpolation and extrapolation.
3. to examine the relationship between algorithm behaviour and the control variable settings.

### 3.1 Design

This study uses experiments in which all relevant explanatory variables are examined simultaneously. Such experimental designs are called "factorial". Factorial designs have two major advantages over other designs (see pages 150 and 151 of (Cochran and CoxCochran and Cox1957)). First, they allow one to investigate quantitatively the interactions between explanatory variables. Second, they permit the quantification of the relationship between the outcome and the explanatory variables with a high degree of precision in a single experiment.

To enjoy these advantages, each possible combination of variable levels must appear once. Such a design is said to be fully crossed and balanced. If each explanatory variable has  $l_j$  levels, the number of combinations to be tested is  $\prod_{j=1}^J l_j$  for  $J$  explanatory variables, which can rapidly become large, making both the conduct and analysis of the experiment difficult. Some statistical texts recommend that the number of factors be reduced using preliminary experiments, e.g. Cochran and Cox (Cochran and CoxCochran and Cox1957). Other authors argue that a factorial design can serve as a summary of the data, e.g. Hays (HaysHays1994). In his paper we take the view that it is worth using the largest feasible design and then simplifying. However, the fact remains that for a large number of factors, the number of levels per factor must be kept small for the experiment to be feasible. For example, a design with five factors each having two levels, requires that 32 combinations be tested. If there are four levels per factor, there will be 1024 combinations to consider. This is a problem because the genetic algorithm control variables and problem size can take very many different values and it would be best to look at as many of these as possible. It would also be convenient to do some preliminary experiments to determine the interesting ranges of these variables.

A cheaper alternative to a fully crossed balanced factorial design is a fractionally replicated factorial or graeco-latin square design, such as that shown in figure 2. It is conventional to denote the levels of each factor using a different alphabet. (In this case the first factor has levels 1, 2 and 3; the second I, II and III; the third A, B and C; and the fourth  $\alpha$ ,  $\beta$ , and  $\gamma$ .) This design has the property that no level of any explanatory variable appears more than once with any level of any other explanatory variable: there are no pairwise correlations between the explanatory variables. This allows the investigation of individual effects of explanatory variables using fewer combinations. For example, four factors of nine levels each require  $9^4 = 6561$  combinations in a full factorial design, but only  $9^2 = 81$  combinations in a graeco-latin square. The price to be paid

for reducing the dimensionality in this way is an inability to consider interactions since the co-occurrence of, say, 2 and A cannot be distinguished from the co-occurrences of 2 and III, 2 and  $\beta$ , III and  $\beta$ , or A and  $\beta$ . However, the reduced dimensionality of the graeco-latin square allows one to consider more levels per factor than the corresponding full factorial design for the same computational effort.

It is worth emphasising that the purpose of the graeco-latin square arrangement is to explore wider ranges of factor levels. Detailed analysis cannot be based on graeco-latin square experiments if interactions between the factors are highly likely to occur, as we suspect is the case with genetic algorithm control variables. However, this design can be used to find suitable ranges of the control variables and check that the response is continuous over the domain of interest. The full analysis can then be based on a fully crossed design with a narrower focus.

	1	2	3
I	$A_\alpha$	$B_\gamma$	$C_\beta$
II	$B_\beta$	$C_\alpha$	$A_\gamma$
III	$C_\gamma$	$A_\beta$	$B_\alpha$

Figure 2: A 3x3 Graeco-Latin Square. The first factor has levels 1, 2 and 3; the second I, II and III; the third A, B and C; and the fourth  $\alpha, \beta$ , and  $\gamma$ .

### 3.2 Method

The genetic algorithm to be studied is a simple generational algorithm, c.f. (GoldbergGoldberg1989). The encoding will use the labellings directly. That is, each individual will be represented as a string of labels. A binary encoding was not chosen since this would have enabled the crossover operator to change individual labels<sup>2</sup> in addition to transferring them between parents. Selection will be by the standard roulette method. The fitness function will be based on the label consistency criteria appearing in equations 2 and 6. We use superscripts to index the different global label configurations appearing in the population of solutions. Accordingly, we denote the  $i^{\text{th}}$  global label configuration appearing in the population by  $\Gamma^{<i>}$ . The population size is  $x_p$ . The fitness function is:

$$\text{Fitness}^{<i>} = \begin{cases} P(\Gamma^{<i>}) & \text{for the probabilistic labelling criterion (equation 2)} \\ \exp[-E(\Gamma^{<i>})] & \text{for the minimum-distance criterion (equation 6)} \end{cases} \quad (7)$$

In these experiments, the idea is to use graeco-latin squares in a preliminary study to identify suitable ranges of population size, iteration limit, crossover rate and mutation rate. Then, fully crossed designs will be used to derive statistical models of the effects of these control variables on the algorithm's performance. Thus the genetic algorithm control variables will be in either graeco-latin square or fully crossed configurations, and this arrangement will in turn be embedded in a fully crossed design for criterion, crossover type and problem size. The resulting experimental design matrix has 7 dimensions in both cases. These arrangements are shown in figure 3.

Since exact solutions exist to the line labelling problem, it is possible to determine whether or not the algorithm has located a global optimum - i.e. one of the consistent

<sup>2</sup>For example, given  $(0, 0) \mapsto +$  and  $(0, 1) \mapsto -$ , a crossover could transform a + to a -.

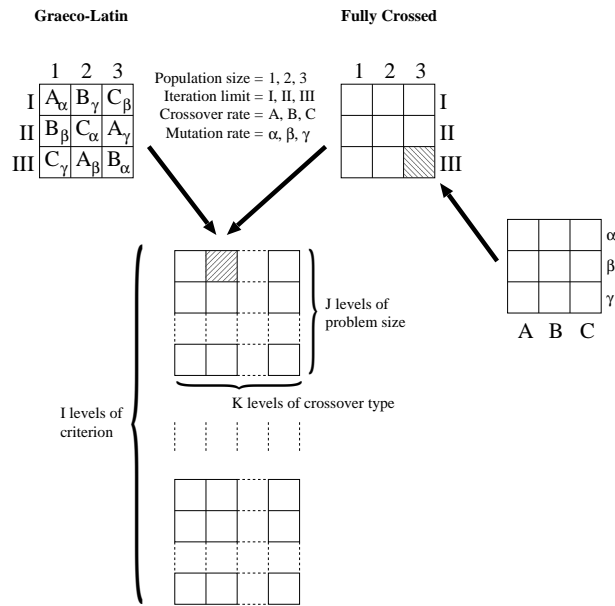


Figure 3: Experimental Design. The genetic algorithm control variables are shown with 3 levels each for simplicity. The arrows represent embeddings. In practice, there were 9 levels for the graeco-latin square configuration (81 combinations) and 4 levels for the fully crossed configuration (256 combinations). There were  $J = 10$  levels of problem size,  $K = 4$  levels of crossover type and  $I = 2$  levels of criterion.

labellings for which  $E(\Gamma) = 0$ . Thus, algorithm performance can be measured by a binary variable which takes the value 1 if the algorithm has located a global optimum (the algorithm is stopped when this happens, so the number of iterations is really an upper limit). Analysing binary outcome variables is inconvenient when attempting to estimate the probability that the algorithm will find a global optimum. So 20 program runs per factor-level combination will be performed, and the number of successful runs will be treated as having a binomial distribution with a sample size of 20. This quantity is the “success rate” of the algorithm. The experiments were conducted on a SGI Origin2000 computer with 32 180MHz MIPS R10000/R10010 processors. Timings vary, but each experiment required between four and eight days of computer time.

### 3.3 Statistical Models

Generalised linear models (McCullagh and NelderMcCullagh and Nelder1989) were fitted to the data using NAG’s GLIM Version 4 update 8 (Francis, Green, and Payne-Francis et al.1993). These models are of the general form:

$$r = g^{-1}(\eta + \epsilon) \tag{8}$$

where  $r$  is the outcome variable (proportion of successful runs in this case),  $g$  is the “link function” (q.v.), which transforms  $r$  so that it matches the range of the “linear predictor”,  $\eta$ , and  $\epsilon$  is the additive noise resulting from random observation errors. The

errors are assumed to have zero mean. As a result, the expected value of  $g(r)$  depends only on the linear predictor,  $\eta$ , which in turn depends on the explanatory variables as follows:

$$\eta = \sum_{0 \leq j \leq J} \beta_j x_j \tag{9}$$

where  $J$  is the number of explanatory variables, and  $\beta_j$  is the coefficient of the  $j^{\text{th}}$  explanatory variable,  $x_j$ . It is conventional to take  $x_0 = 1$  so that  $\beta_0$  represents a constant offset. Such models are known as “generalised linear models” (McCullagh and NelderMcCullagh and Nelder1989) because they are linear in the parameters  $\beta$ , but not necessarily in the variables  $x_j$ . Indeed, any transformation of  $x_j$  may be substituted, such as its logarithm or reciprocal, or even  $K$  polynomials of degrees  $K, K - 1$ , and so-on down to 1.

The link function,  $g$ , is used to map the domain of the response variable,  $r$ , onto the range of the linear predictor,  $\eta$ , which is taken to be  $(-\infty, +\infty)$ . In the particular case when  $r$  is a proportion, the link function transforms the variables from the interval  $[0, 1]$  onto the set of reals  $(-\infty, +\infty)$ . There are four such link functions of interest and which are listed below

- The logistic or logit link,  $\text{logit}(r) = \ln\left(\frac{r}{1-r}\right)$ , which has a natural interpretation as the log-odds of the event associated with  $r$ .
- The probit link,  $\text{probit}(r) = \Phi^{-1}(r)$ , where  $\Phi(a)$  is the standardised normal integral from  $-\infty$  to  $a$ .
- The complementary log-log link,  $\text{cll}(r) = \ln[-\ln(1 - r)]$
- The parameterised link function introduced by Aranda-Ordaz in (Aranda-OrdazAranda-Ordaz1981) which is  $g(r; \alpha) = \ln\left[\frac{(1-r)^{-\alpha}-1}{\alpha}\right]$  for  $\alpha > 0$ . This has two important special cases. First, when  $\alpha = 1$ , the function is equivalent to the logistic link,  $g(r; 1) \equiv \text{logit}(r)$ . Second, when  $\lim_{\alpha \rightarrow 0} g(r; \alpha) = \text{cll}(r)$ , the complementary log-log link function results.

The model algebra developed by Wilkinson and Rogers (Wilkinson and RogersWilkinson and Rogers1973) will be used to describe the form of the linear predictor. The notation is shown in table 2.

Notation	Interpretation
A+B	denotes the main effects $\beta_A A + \beta_B B$
A . B	denotes the interaction $\beta_{AB} AB$
A * B	is equivalent to $A + B + A . B$
A <k>	denotes the polynomials $\beta_{A1} X_A^1 + \dots + \beta_{Ak} X_A^k$
<b>M</b> * * n	represents the cartesian product $\mathbf{M}^n$

Table 2: Summary of Model Algebra. A and B are individual terms and **M** is a set of terms.

Thus the notation

$$1+A*B+C<2> \tag{10}$$

corresponds to the model

$$\left. \begin{aligned} r &= g^{-1}(\eta) \\ \eta &= \beta_0 + \beta_A A + \beta_B B + \beta_{AB} AB + \\ &\quad \beta_{C1}(\gamma_{C10} + \gamma_{C11} C) + \\ &\quad \beta_{C2}(\gamma_{C20} + \gamma_{C21} C + \gamma_{C22} C^2) \end{aligned} \right\} \quad (11)$$

where the coefficients  $\gamma$  are chosen to take into account the correlation between  $C$  and  $C^2$  (such polynomials are termed “orthogonal” for this reason).

### 3.3.1 The Modelling Process

In general, the experimenter chooses the link function,  $g$ , and the form of the linear predictor, i.e. which (combinations of) explanatory variables to include. The model is then fitted by computing the maximum likelihood estimates of the model coefficients,  $\beta$  (and  $\gamma$ ), from the data. Thus the choice of models has an inevitable subjective element. This has the advantage that the experimenter can choose a model form appropriate for the task at hand. The implication of this is that modelling is an iterative process in which many different models will have to be constructed and evaluated. When comparing different models, the experimenter should be guided by statistical significance criteria. Where the outcome of the experiment is a proportion, as is the case in this study, logistic regression is used (CollettCollett1991).

In logistic regression models are compared using the deviance, which is a summary measure of goodness of fit derived from the likelihood ratio of the current model to that of an ideal model with a parameter for every observation. It is approximately distributed as  $\chi^2$  on the residual degrees of freedom of the current model. See (CollettCollett1991) and (Francis, Green, and PayneFrancis et al.1993) for a detailed discussion. The general method adopted in this study will be to start with relatively complex models, and then simplify them.

The standard method for simplifying models in logistic regression is to observe that for binomial data, if two models **M1** and **M2** are nested, i.e. the explanatory variables in **M2** are a subset of those in **M1**, the difference in their deviances is approximately distributed as  $\chi^2$  on the difference in their d.f. (CollettCollett1991; Francis, Green, and PayneFrancis et al.1993). This approximation is good when the total number of binary observations is high (as it is here being 163840). Thus, terms may be removed or added to a model, their significance being judged on the basis of  $\chi^2$ -tests in a procedure analogous to the analysis of variance.

Having fitted and simplified a model it is then necessary to determine its adequacy. The adequacy of the link function,  $g$ , can be assessed informally using the goodness of link test described in (CollettCollett1991). In this test the link function with the lowest deviance gives the best fit. For the Aranda-Ordaz link function, the constructed variable technique described in (CollettCollett1991) was used to estimate the parameter of the link function. The adequacy of the model as a whole is assessed by comparing its predictions with the original data. The differences are the residuals, which should have well-defined distributions. For modelling binomial data, Anscombe residuals are constructed so as to have a standard Normal distribution (AnscombeAnscombe1953; CollettCollett1991; Francis, Green, and PayneFrancis et al.1993). An Anscombe residual of magnitude 1.96 is therefore significantly large at the 5% confidence level. Further details of model fitting and logistic regression can be found in (CollettCollett1991) and (Francis, Green, and PayneFrancis et al.1993).

## 4 Experimental Results and Analysis

This section presents and analyses the results of the modelling experiments described in the previous section. First, section 4.1 describes a preliminary study using graeco-latin squares. Then section 4.2 gives brief details of the full factorial experiments. These experiments are analysed in sections 4.3 (optimal performance), 4.4 (robustness) and 4.5 (algorithm behaviour). Finally, section 4.6 demonstrates that the modelling process can be extended to the harder labelling problem of graph matching.

### 4.1 Preliminary Study

A preliminary study was performed using an embedded 9x9 graeco-latin square configuration. Thus the total number of observations was  $2 \times 4 \times 10 \times 9 \times 9 = 6480$ . Since each observation corresponded to 20 trials, the total number of program runs was 129600. The values for the control variables are given in table 3. For clarity, the factor names, e.g. "LINES", will be used in the text, and the variable names, e.g. " $x_L$ ", will be used in the formulae. For non-ordinal variables, switch variables such as " $w_X$ " will be defined in formulae.

Variable	Notation	Values
Criterion type	COST, $w_C$	minimum-distance, probabilistic
Crossover type	CROSS, $w_X$	uniform, two point, half-uniform, geometric
Problem size	LINES, $x_L$	6, 7, 8, 9, 14, 20, 21, 28, 40, 42
Population size	POP, $x_P$	20, 40, 60, 80, 100, 120, 140, 160, 180
Iteration limit	GEN, $x_G$	10, 20, 100, 200, 250, 300, 350, 400, 500
Crossover rate	$P_x$ , $x_X$	0.1, 0.2, 0.35, 0.45, 0.6, 0.65, 0.75, 0.85, 0.9
Mutation rate	$P_m$ , $x_M$	0.001, 0.005, 0.01, 0.02, 0.05, 0.075, 0.1, 0.2, 0.3

Table 3: Control Variables for Preliminary Study.

The results of the experiments must be analysed graphically, since this design does not give access to interactions between all the variables. Figure 4 shows the main effects of population size, iteration limit, crossover rate and mutation rate. Panel (a) shows the effect of increasing population size on the success rate (proportion of successful program runs). Performance improves initially with problem size, but the rate of improvement seems to fall off. The performance reaches a plateau as the iteration limit increases, as shown in panel (b). The crossover rate, shown in panel (c), does not appear to be as important as one might have expected. Higher crossover rates do seem to improve success rate up to some upper limit beyond which there is a falloff. The effects appear slight and it is possible say only that the optimal crossover rate will probably occur in the range [0.6,0.9] for each problem. Since like mutation rate, crossover rate is defined per gene, its effects might be expected to scale with the problem size. However, the most striking feature of these plots is the sensitivity to mutation rate shown in panel (d). Although not shown here, the shape of the mutation rate plot was the same for all problem sizes. The optimal mutation rate seems to be around 0.02. The

scalability of mutation rate effects is unsurprising since mutation is genic (i.e. more lines, more mutations). The fact that low but non-zero values of the mutation rate are beneficial agrees with a general view in the genetic algorithm literature that mutation is a necessary source of background noise, allowing the exploration of new regions of the search space, but that it is not the primary mechanism of algorithm convergence. It should be stressed that local search was not used. As shown in section 4.3.1, hybrid genetic algorithms tolerate much higher mutation rates.

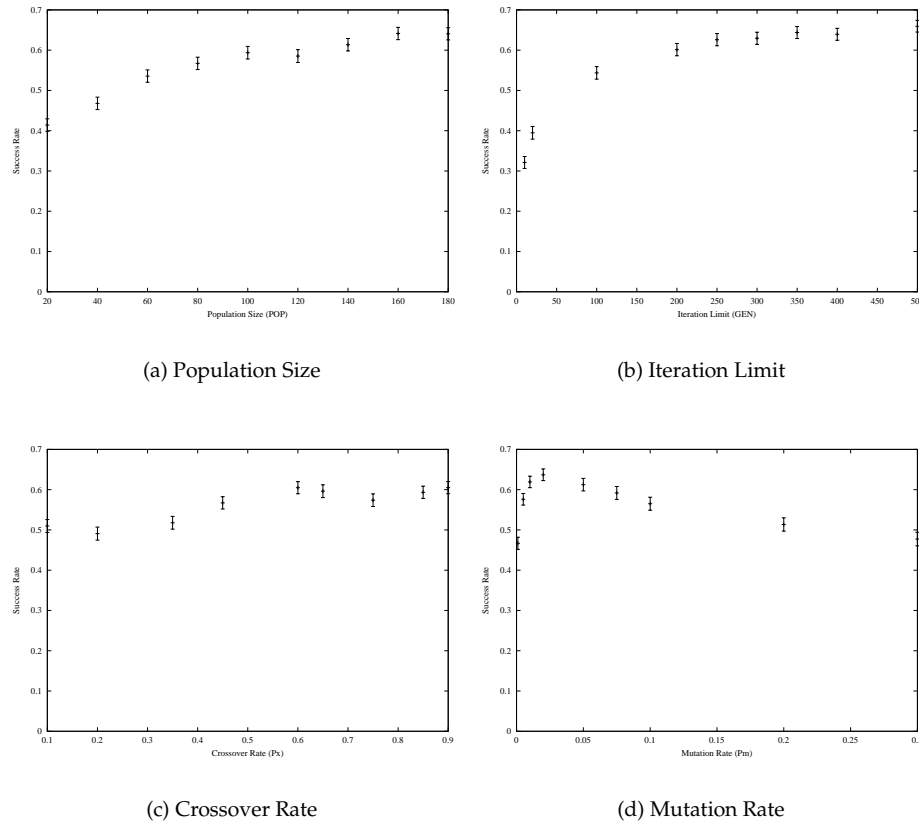


Figure 4: Main Effects of Genetic Algorithm Control Variables.

## 4.2 Factorial Experiments

In all, three factorial experiments were performed: one without local search and two with local search. Each experiment used a fully crossed factorial design and produced 8192 observations (see table 4) over 163840 program runs (= 8192 × 20 observations). Although the computational effort is roughly the same as for the preliminary study, this design permits interactions to be considered, but at the price of only having four levels of each explanatory variable. The explanatory variables are summarised in table 4.

Variable	Values
Criterion type	minimum-distance, probabilistic
Crossover type	uniform, two point, half-uniform, geometric
Problem size	9, 21, 28, 40
Population size	50, 100, 150, 200 (without local search) 10, 20, 30, 40 (with local search)
Iteration limit	50, 150, 250, 350 (without local search) 2, 4, 6, 8 (with local search)
Crossover rate	0.2, 0.4, 0.6, 0.8
Mutation rate	0.01, 0.02, 0.03, 0.04

Table 4: Factors Included in Line Labelling Experiments. There are  $2 \times 4 \times 4 \times 4 \times 4 \times 4 \times 4 = 8192$  combinations.

### 4.3 Modelling Optimal Performance

#### 4.3.1 Standard versus Hybrid Algorithm

In these experiments, the addition of a local search step was found to greatly improve algorithm performance: it achieved a higher success rate with one fifth of the population size and one twenty-fifth of the iterations when compared to the plain algorithm. This section does not dwell on the plain algorithm, but considers some contrasts between it and the hybrid version. Perhaps the most interesting differences between the hybrid and non-hybrid algorithms concern the label consistency criterion type and the mutation rate.

Figure 5 shows the average success rates for each label consistency criterion with the plain and hybrid algorithms. The most likely explanation for this is that local search does better with the arithmetic mean probabilistic labelling criterion. This is to be expected since the arithmetic mean probabilistic labelling criterion is smoother, so local search is less likely to get stuck in local optima with the this criterion than with the minimum-distance one. Another important difference is the effect of mutation rate on success rate, shown in figure 6, where the hybrid algorithm appears much less sensitive to changes in mutation rate than the plain one. Indeed, to observe any interesting variation, the range of mutation rates had to be extended to  $[0.1, 0.7]$ . This is presumably because local search will tend to correct small disturbances caused by mutation, hence the hybrid algorithm will tolerate higher mutation rates.

The histograms of success rate, shown in figure 7, are also very different. The histogram for the plain algorithm is bimodal, indicating that some problems were very hard for this algorithm while others were very easy. The histogram for the hybrid algorithm has only one mode, suggesting that all the problems were quite easy for the algorithm. For the hybrid algorithm, about 75% of the observations coincide with the mode, indicating that the algorithm located the global optimum in almost all cases. Since the hybrid algorithm almost never failed to solve a problem in all 20 trials, it would appear that this algorithm scales better with problem size than the plain one. The hybrid algorithm is now considered in some detail.

#### 4.3.2 Local Search Hybrid

As the histogram in panel (b) of figure 7 indicates, about 75% of the program runs succeeded in each of the 20 trials. A pertinent question at this point is whether there is any variation at all over some of the explanatory variables. Table 5 shows the proportion

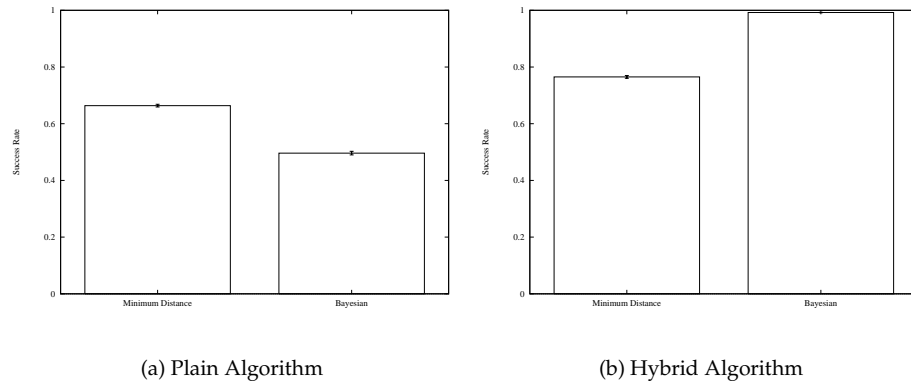
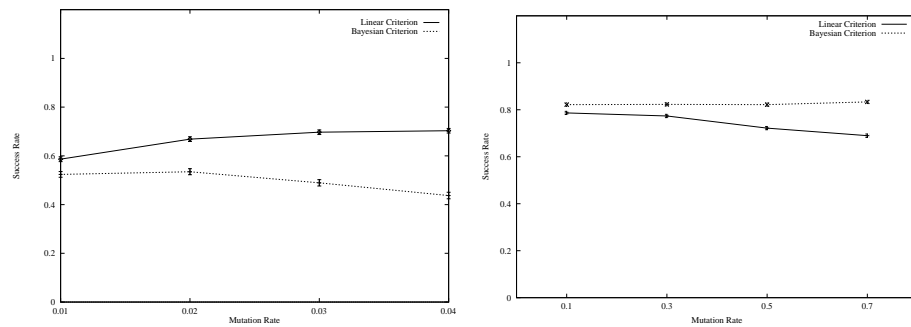


Figure 5: Effect of Criterion on Success Rate. The improvement of the hybrid algorithm over the standard one for the arithmetic mean probabilistic labelling criterion greatly outstrips that for the minimum Hamming distance criterion. The error bars in this and subsequent figures refer to the distributions of the means rather than the distributions of the underlying data.



(a) Plain Algorithm (mutation rate in [0.01,0.04]) (b) Hybrid Algorithm (mutation rate in [0.1,0.7])

Figure 6: Effect of Mutation Rate on Success Rate. Note that the range of mutation rates is about 20 times larger for the hybrid algorithm. With local search, the algorithm appears relatively insensitive to mutation rate.

of cells for which the observed success rate was 100% for each level of the explanatory variables. Tabulating the pairwise interactions between each pair of explanatory variables (COST.LINES is shown in table 6 as an example) shows that only the following combinations were saturated: COST = probabilistic and LINES = 9 or 28; POP = 30 or 40 and LINES = 9; GEN = 8 and LINES = 9. There may therefore still be scope for modelling, especially with the minimum Hamming distance criterion.

The initial statistical model fitted was the quadratic response surface,

$$1 + (COST + CROSS + LINES + POP + GEN + P_x + P_m) ** 2 + (COST + CROSS) * (LINES < 2 > + POP < 2 > + GEN < 2 > + P_x < 2 > + P_m < 2 >) \tag{12}$$

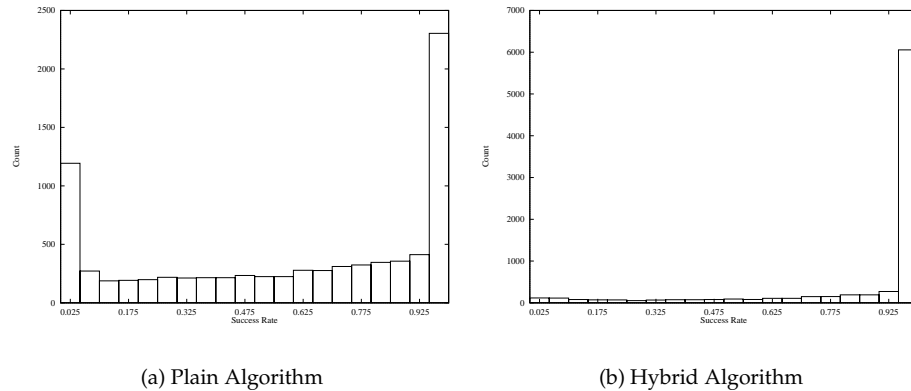


Figure 7: Histograms of Success Rate. Counts are summed over all variables. None of the test problems seemed hard for the hybrid algorithm. The plain algorithm clearly did not find the global optimum in some cases.

Factor / Level	1	2	3	4
COST	1794 (0.44)	3831 (0.94)	-	-
CROSS	1369 (0.67)	1422 (0.69)	1405 (0.69)	1429 (0.70)
LINES	2011 (0.98)	1358 (0.66)	1492 (0.73)	764 (0.37)
POP	1105 (0.54)	1350 (0.66)	1530 (0.75)	1640 (0.80)
GEN	1123 (0.55)	1422 (0.69)	1524 (0.74)	1556 (0.76)
$P_x$	1311 (0.64)	1384 (0.68)	1449 (0.71)	1481 (0.72)
$P_m$	1373 (0.67)	1395 (0.68)	1420 (0.69)	1437 (0.70)

Table 5: Saturated Factor Levels for Hybrid Algorithm. The values are summarised over all but one factor in each row. The total number of cells at each level is 4096 for COST and 2048 for the other factors. The table shows counts (proportions) of cells which had 20 out of 20 successes for each factor level.

which has a deviance<sup>3</sup> of 6776.9 on 8124 degrees of freedom (d.f.). The model contains 68 terms, which suggests that it is somewhat over-fitted.

COST	LINES			
	9	21	28	40
Minimum-distance	987 (0.96)	336 (0.33)	468 (0.46)	3 (0.00)
Probabilistic	1024 (1.00)	1022 (1.00)	1024 (1.00)	761 (0.74)

Table 6: Saturated Cells Between COST and LINES. The maximum possible number of saturated cells is 1024. Only the 40-line problem presents problems to the arithmetic mean probabilistic labelling criterion.

The analysis of deviance for the interactions of COST in model 12 is given in table 7 as an example. There is insufficient space to give full analyses, so only the results will be quoted. Table 7 shows that in addition to an obvious main effect, four out of

<sup>3</sup>See section 3.3.

six pairwise interactions are significant. This means that there are really two different models, one for each label consistency criterion with COST effectively acting as a switch variable. These would be cumbersome to consider simultaneously. Therefore each label consistency criterion will be modelled separately, partitioning the data on COST.

Interaction	Deviance change	d.f. change	p-value
COST . CROSS	24.05	3	0.00
COST . LINES	136.7	2	0.00
COST . POP	132.2	2	0.00
COST . GEN	0.1460	2	0.93
COST . $P_x$	19.58	2	0.00
COST . $P_m$	0.9683	2	0.62

Table 7: Analysis of Deviance for Interactions of COST. The p-value indicates the probability that such a large deviance change could have arisen by chance if the term in question really were insignificant.

### 4.3.3 Probabilistic Labelling Criterion

As shown in table 5, 94% of cells with the arithmetic mean probabilistic labelling criterion have 100% success rates. The interaction tabulated in table 6 indicates that this label consistency criterion is almost always successful with the 9 and 28-line problems, given the other explanatory variables. Because there is relatively little variation in the data, it is only sensible to consider models which are linear in the explanatory variables: there is insufficient variation in the data set to support more complex models. The first model is:

$$1 + \text{CROSS} + \text{LINES} + \text{POP} + \text{GEN} + P_x + P_m \tag{13}$$

which has a deviance of just 675.72 on 4087 d.f. from 4096 observations (half the data). The part of the linear predictor involving CROSS is shown in table 8.

Term	$\hat{\beta}$	s.e. ( $\hat{\beta}$ )	t-value	p-value
CROSS = two point	-0.2850	0.1214	-2.348	0.02
CROSS = geometric	-0.1964	0.1231	-1.595	0.11
CROSS = half-uniform	$4.700 \times 10^{-14}$	0.1275	$3.687 \times 10^{-13}$	1.00

Table 8: Parameter Estimates for CROSS. The t-value is the ratio of the estimate to its standard error,  $\hat{\beta}/s.e.(\hat{\beta})$ . For binomial data, the t-value has a standard Normal distribution under the null hypothesis,  $\beta = 0$  (CollettCollett1991). The significance test is one-tailed.

From this table, only two point crossover has a parameter significantly different from zero, so uniform, geometric and half-uniform crossovers may be amalgamated, and an offset for two point crossover included, in future models. An analysis of the link function, given in table 9, showed that the probit link gave the best fit. The shape of the curves in panels (a) and (b) of figure 4 suggested that there might be a logarithmic relationship between POP and GEN and the success rate. Taking the logarithms of POP and GEN improved the fit. This leads to the model

$$\left. \begin{aligned} r &= \text{probit}^{-1}(\eta) \\ \eta &= 3.659 - 0.1786x_L + 1.176 \ln x_P + 1.080 \ln x_G + \\ &\quad 1.010x_X + 6.066x_M + 0.1243w_X \\ w_X &= \begin{cases} 0 & \text{for two point crossover} \\ 1 & \text{otherwise} \end{cases} \end{aligned} \right\} \quad (14)$$

This model will help achieve the primary goal of finding optimal conditions for the algorithm since the arithmetic mean probabilistic labelling criterion is the better of the two alternatives. The extraction of control variable settings from this model is discussed in the next section, where its robustness to interpolation and extrapolation are investigated.

Link function	Deviance
Logit	679.15
Ideal Aranda-Ordaz	675.64
Complementary log-log	689.47
Probit	674.11

Table 9: Goodness of Link Test. The link function with the lowest deviance gives the best fit, which in this case is the probit link.

#### 4.4 Robustness of the Model

The second goal of this study is to establish how suitable the model from the previous section is for the purpose of setting control variables. From equation 14, the optimal crossover is non-two point. If the crossover type is fixed at non-two point (uniform), equation 14 becomes

$$\left. \begin{aligned} r &= \text{probit}^{-1}(\eta) \\ \eta &= 3.783 - 0.1786x_L + 1.176 \ln x_P + 1.080 \ln x_G + 1.010x_X + 6.066x_M \end{aligned} \right\} \quad (15)$$

The task is now to solve the inverse problem: for a given value of LINES, what values of POP, GEN,  $P_x$  and  $P_m$  will give an acceptably high success probability,  $p$ ? This is generally not an easy problem to solve. However, if the surface is simple enough, numerical optimisation techniques such as the downhill simplex method can be used (see pages 408-412 of (Press, Teukolsky, Vetterling, and FlanneryPress et al.1992) for a good description and implementation). Table 10 shows how well the model predicts control variables for the four problems in the data set (top 4 rows), new problems requiring interpolation (middle 3 rows), and problems requiring extrapolation (bottom 4 rows). For each problem, the simplex optimum is given together with a prediction of the success probability based on equation 15. The last column shows the actual success rate from 100 trials.

The model seems reasonable for interpolation, which is not really surprising since the process of model-fitting is essentially one of interpolation. However, the model's performance degrades very rapidly on extrapolation, giving excessively large estimates for POP, GEN, and in the last case  $P_m$ . The required population sizes are at odds with both the predictions of equation 21 and one's experience with genetic algorithms. On

LINES	Simplex Optimum					Predicted 95% Conf. Int. for $p$	Actual $p$ (N=100)
	POP	GEN	$P_x$	$P_m$	$\hat{p}$		
9	5	2	0.008	0.0003	1.00	[1.00, 1.00]	1.00
21	6	2	0.005	0.001	0.99	[0.99, 1.00]	1.00
28	6	3	0.38	0.050	0.99	[0.99, 1.00]	1.00
40	7	13	0.77	0.040	0.99	[0.99, 1.00]	1.00
26	6	3	0.29	0.048	0.99	[0.99, 1.00]	0.94
33	6	5	0.69	0.043	0.99	[0.99, 1.00]	1.00
36	6	8	0.79	0.041	0.99	[0.99, 1.00]	1.00
41	8	14	0.69	0.039	0.99	[0.99, 1.00]	1.00
53	22	32	0.78	0.025	0.99	[0.98, 1.00]	1.00
65	61	59	0.71	0.076	0.99	[0.96, 1.00]	1.00
130*	1039	793	0.98	0.93	0.99	[0.07, 1.00]	1.00

Table 10: Performance of Model 15. The upper rows are the original test problems, the middle rows are interpolation problems and the bottom rows are extrapolation problems. The simplex optimum vector is shown, together with a 95% confidence interval for the outcome based on the model, and the actual outcome over 100 trials of the algorithm. The initial guess for the simplex method was POP=40, GEN=8,  $P_x=0.8$  and  $P_m=0.03$ , except for the row marked (\*), which was POP=800, GEN=800,  $P_x=0.9$  and  $P_m=0.3$ .

the other hand, the predictions for crossover and mutation rates appear quite reasonable, except in the last case. This suggests that the problem is with the form of the model.

Although there are good reasons to believe that logarithmic terms in POP and GEN explain the algorithm’s performance well, a little re-arrangement of equation 15 will indicate that, all other things being equal, the model is of the form:  $\ln x_P \propto x_L + k$ . This suggests that the population size should increase exponentially with the number of lines to be labelled, exactly the opposite of the previous conclusion.

The implication of equation 21 and figure 13, is that a model should be used which is logarithmic in LINES rather than in POP and GEN. Such a model was fitted, and its functional form is given in equation 16, and also includes interaction terms which further increase the estimated success rate. Since it is generally inadvisable to extrapolate with polynomial models, only linear terms are included. Table 11 shows the results for this model. It still appears to overestimate the requirements for POP and GEN but nowhere near as badly as model 15. On the other hand, it is not very good for small problems. However, recalling figure 9, the original model was over-fitted to the small problems, which may partly explain its poor performance under extrapolation.

$$\left. \begin{aligned}
 r &= \text{probit}^{-1}(\eta) \\
 \eta &= 20.92 - 5.597 \ln x_L + 0.02215x_P - 0.1781x_G + 0.6277x_X - 11.64x_M + \\
 &\quad 0.01597x_Px_G + 0.1229x_gx_X + 5.506x_Gx_M
 \end{aligned} \right\} \tag{16}$$

The surface can be visualised by fixing  $P_x$  and  $P_m$  as before, and requiring that  $r = 0.99$ , hence  $\text{probit}(r) = 2.326$ : it is shown in figure 8. The figure shows that the model extrapolates stably as LINES and POP increase well beyond the range of the data set. Whether or not the predicted value of GEN is correct is another matter: it is decreasingly likely to be accurate as one departs further from the original data.

LINES	Simplex Optimum					Predicted 95% Conf. Int. for $p$	Actual $p$ (N=100)
	POP	GEN	$P_x$	$P_m$	$\hat{p}$		
9	6	2	0.006	0.001	1.00	[1.00, 1.00]	0.99
21	5	1	0.005	0.001	1.00	[1.00, 1.00]	0.57
28	5	1	0.19	0.00	1.00	[0.97, 1.00]	1.00
40	6	6	0.74	0.05	0.99	[0.98, 1.00]	0.90
26	5	1	0.00	0.00	1.00	[0.98, 1.00]	0.82
33	6	3	0.72	0.038	0.99	[0.98, 0.99]	1.00
36	6	5	0.52	0.044	0.99	[0.98, 0.99]	1.00
41	6	7	0.60	0.042	0.99	[0.98, 0.99]	0.98
53	7	10	0.72	0.044	0.99	[0.96, 1.00]	1.00
65	16	8	0.79	0.048	0.99	[0.93, 1.00]	1.00
130	32	11	0.74	0.044	0.99	[0.88, 1.00]	1.00

Table 11: Performance of Model 16. The upper rows are the original test problems, the middle rows are interpolation problems and the bottom rows are extrapolation problems. The simplex optimum vector is shown, together with a 95% confidence interval for the outcome based on the model, and the actual outcome over 100 trials of the algorithm. Again, the initial guess for the simplex method was POP=40, GEN=8,  $P_x=0.8$  and  $P_m=0.03$ .

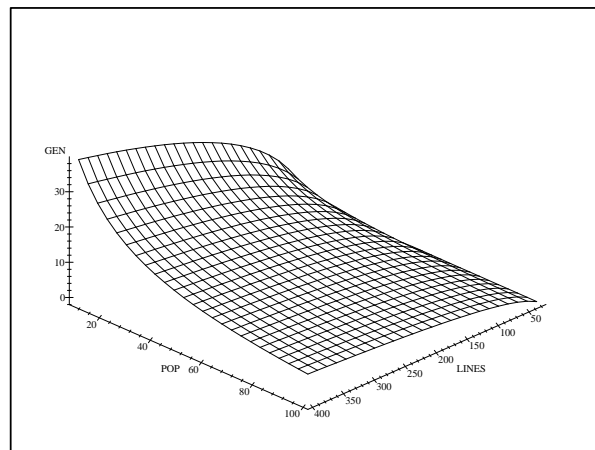


Figure 8: Plot of Model 16. Number of iterations (GEN) required to achieve a success rate of 0.99 is plotted against LINES and POP.  $P_x$  and  $P_m$  are fixed at 0.8 and 0.03.

#### 4.5 Algorithm Behaviour

In this Section, we aim to shed some light on role of the explanatory variables. The arithmetic mean probabilistic labelling criterion worked so well that there was little scope for such investigation in the previous section, so the minimum Hamming distance criterion was used for this purpose. Initial analysis suggested that the complementary log-log link function was the most appropriate. Again, it appears on the basis of t-values that uniform, geometric and half-uniform crossovers could not be distinguished, so CROSS will be redefined with only two levels: two point crossover and

non-two point crossover.

Since there are only four levels of LINES, POP, GEN,  $P_x$  and  $P_m$ , the most complex response surface supported is cubic:

$$\begin{aligned}
 &1 + \text{CROSS} * (\text{LINES} < 3 > + \ln(\text{POP}) < 3 > + \ln(\text{GEN}) < 3 > + P_x < 3 > + P_m < 3 > \\
 &+ (\text{LINES} + \ln(\text{POP}) + \ln(\text{GEN}) + P_x + P_m) ** 2 \\
 &+ (\text{LINES} + \ln(\text{POP}) + \ln(\text{GEN}) + P_x + P_m) ** 3
 \end{aligned} \tag{17}$$

This model has a deviance of 3140.6 on 4034 d.f. The model can be simplified using repeated analysis of deviance to

$$\begin{aligned}
 &1 + \text{CROSS} * (\text{LINES} < 3 > + \ln(\text{GEN}) < 2 >) + \ln(\text{POP}) + P_x < 2 > + P_m < 2 > \\
 &+ (\text{LINES} + \ln(\text{POP}) + \ln(\text{GEN}) + P_x + P_m) ** 2 - \ln(\text{POP}) . P_x \\
 &+ \text{LINES} . \ln(\text{POP}) . \ln(\text{GEN}) + \text{LINES} . \ln(\text{GEN}) . P_m
 \end{aligned} \tag{18}$$

The resulting fit has deviance 3179.2 on 4068 d.f. Since these models are nested, the deviance change of 38.6 can be compared to  $\chi^2$  on 34 d.f., and found to be insignificant. This is the simplest model which adequately describes the data. A plot of the Anscombe residuals for this model is given in figure 9. The residuals have no clear pattern apart from some over-fitting to the smallest problem, and only about 2% of them are significantly large at the 5% confidence level. Although a significantly large residual implies a poor fit, one would expect about 5% of the residuals to be large purely by chance.

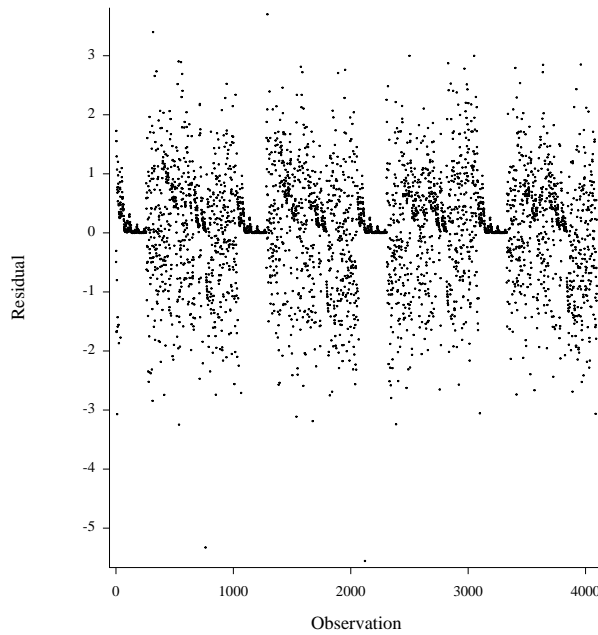


Figure 9: Anscombe Residuals for Model 18. The periodic dense regions in the plot shows that the model over-fits the 9-line problem. Apart from that there is no pattern to the residuals.

The functional form of model 18 contains 28 terms and is given by

$$\begin{aligned}
 r &= \text{cfl}^{-1}(\eta) \\
 \eta &= \left. \begin{aligned}
 &2.401 - 1.213x_L + 0.05144x_L^2 - 0.0007183x_L^3 + 1.943 \ln x_P + \\
 &2.811 \ln x_G - 0.4411(\ln x_G)^2 + 2.325x_X - 0.9407x_X^2 + \\
 &42.46x_M - 184.9x_M^2 - 0.01940x_L \ln x_P - 0.6367 \ln x_P \ln x_G - \\
 &0.003289x_L \ln x_G + 0.01239x_Lx_X - 0.3246x_X \ln x_G - \\
 &5.948x_M \ln x_P - 0.3023x_Lx_M - 9.499x_M \ln x_G - \\
 &10.03x_Xx_M + 0.01970x_L \ln x_P \ln x_G + 0.4675x_Lx_M \ln x_G \\
 &w_X(-1.116 + 0.1529x_L - 0.007309x_L^2 + 0.0001018x_L^3 + \\
 &0.7089 \ln x_G - 0.2670(\ln x_G)^2)
 \end{aligned} \right\} \quad (19) \\
 w_X &= \begin{cases} 0 & \text{for two point crossover} \\ 1 & \text{otherwise} \end{cases}
 \end{aligned}$$

This is a very complex six-dimensional function. The only way to visualise it is to hold some of the explanatory variables constant. Figure 10 shows the success rates for the factor levels of LINES, POP, GEN,  $P_x$  and  $P_m$ . The least important variables appear to be  $P_x$  and  $P_m$ . The coefficients in the model in equation 19 for interactions involving  $P_x$  and  $P_m$  are smaller than those for the main effects, suggesting that the interactions are not as important as the main effects. If three of POP, GEN  $P_x$  and  $P_m$  are fixed, the success rate can be plotted as a function of LINES and the fourth variable. These are shown in panels (a) and (b) of figure 11 for POP and GEN.

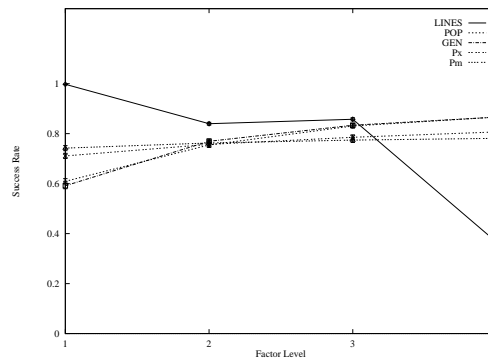
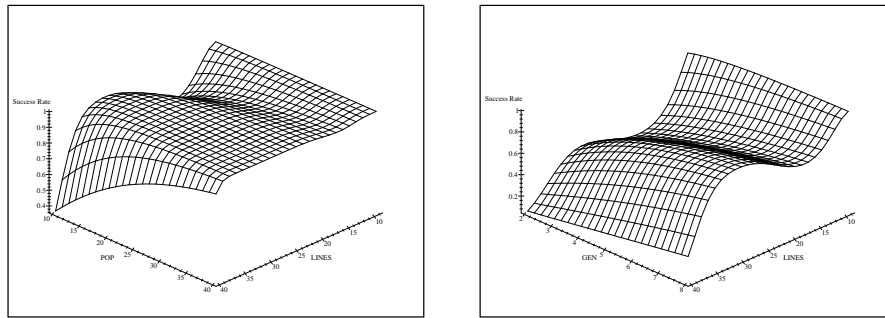


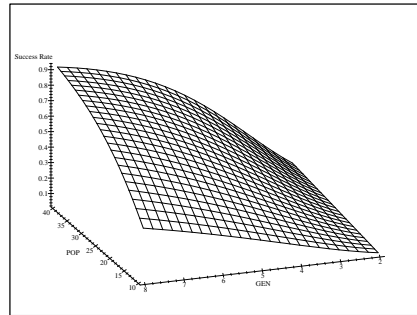
Figure 10: Success Rate Plots. A larger slope indicates that a particular variable has a greater effect on the success rate. The ranges of the variables must also be taken into account - for example, only  $\frac{1}{25}$ <sup>th</sup> of the range of mutation rate is explored.

Although this model provides a good statistical explanation of the data, there remain some question as to its validity. The 28-line problem seems to be easier for the algorithm than the 21-line problem. This difference is not very great but it does account for the cubic polynomial in LINES which appears in equation 19 ( $\dots -1.213x_L + 0.05144x_L^2 - 0.0007183x_L^3 + \dots$ ). Any curve fitted to these points must have two local optima and must therefore be cubic or of higher order. There must be additional aspects of the problems which make them easy or hard, apart from the number of lines in the drawing. These aspects are probably structural - neither the numbers nor ratios of the different junction types are any better at explaining the data than drawing size. These structural differences between line drawings mean that one must be careful about fitting models polynomial in LINES since they are likely to be responding



(a) Success vs. LINES and POP

(b) Success vs. LINES and GEN



(c) Success vs. POP and GEN

Figure 11: Plots of Model 19. In panel (a) success rate is plotted as a function of LINES and POP, with GEN fixed at 8. In panel (b) success rate is plotted against LINES and GEN with POP fixed at 40. In panel (c) success rate is plotted as a function of POP and GEN for the 40-line problem. In all panels,  $P_x$  is fixed at 0.8 and  $P_m$  fixed at 0.04.

as much to structural variations among problems as to problem size. In view of this, attention should perhaps be paid to the relationships to POP and GEN, shown in panel (c) of figure 11.

The success rate increases with the logarithm of the population size confirming the observation made in section 4.1, that there is a limit to the benefit of increasing the population size. This is an interesting phenomenon because the size of the search spaces for these problems are so large that it is very unlikely that a global optimum will be found in the initial population (which is generated at random). Of course, all that is really necessary is that a sufficiently good initial guess for local search is present in the population: this often occurs with the 9-line problem but rarely with the other problems. This consideration leads to a lower bound on population size in terms of the quality of the guesses furnished by the initial population, as follows.

The quality of the initial population can be assessed in terms of the number of loci at which all labels appear. For example, an initial population in which every individual

incorrectly assigned the label '+' to the same line is unlikely to locate an optimal solution quickly. A much better initial population would contain all possible labels for each line. There are only four labels, which are equiprobable, so the probability of a label not appearing at a particular locus is  $p_a = 4 \times (3/4)^{x_P}$ . The probability of this happening at least once is

$$\begin{aligned} P(\text{at least 1 missing label}) &= 1 - P(\text{no missing labels}) \\ &= 1 - (1 - p_a)^{x_L} \end{aligned} \tag{20}$$

This is shown in figure 12, together with the expected number of missing labels ( $\text{LINES} \times p_a$ ), as a function of LINES and POP. For smaller problems, these are effectively zero for population sizes larger than 30. This agrees with the results of the preliminary study and suggests that POP should increase with LINES so that the probability of missing labels in the initial population is very small. If this probability is to be no greater than some threshold,  $P^*$ , manipulation of equation 20 gives:

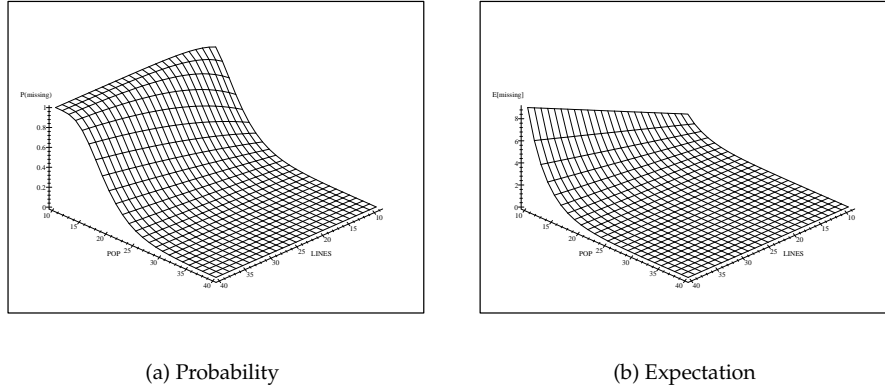


Figure 12: Missing Labels vs. LINES and POP. (a) Probability that at least one label is missing over all loci. (b) Expected number of missing labels over all loci.

$$x_P \geq \frac{\ln \left[ 1 - (1 - P^*)^{\frac{1}{x_L}} \right] - \ln |\mathbf{\Lambda}|}{\ln (|\mathbf{\Lambda}| - 1) - \ln |\mathbf{\Lambda}|} \tag{21}$$

where  $|\mathbf{\Lambda}|$  is the size of the label set (4 in this case). The optimal population size clearly increases with LINES with no upper bound. This formula is not amenable to direct manipulation, but it is shown graphically in figure 13 that this lower bound on population size increases at a lower rate for larger problems.

Under the assumption that the population is large enough to “guarantee” that all labels appear at all loci, all that remains is for the genetic algorithm to assemble a good initial guess for local search. Following this argument, it would seem that high mutation and crossover rates are grist to the mill for local search. There must be some point beyond which increasing the mutation rate is counterproductive, since mutation indiscriminately perturbs the population. Such a danger does not exist for crossover,

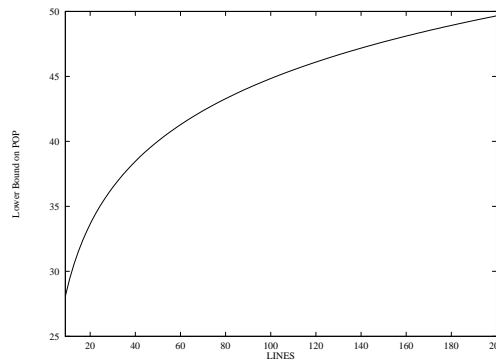


Figure 13: Lower Bound on Population Size. Equation 21 is plotted as a function of LINES with  $P^*$  set to 0.1.

since it has its disruptive effect only at those loci where there is disagreement as to the labelling.

To summarise, the behaviour of the hybrid genetic algorithm can be understood in terms of local search transforming a good initial guess into an optimal solution. This happens over the course of a single algorithm iteration. So, provided the population is large enough to furnish a good selection of labels for all loci, crossover and mutation can between them construct a suitable initial guess within a few iterations. Clearly, the more iterations, the larger the chance of this occurring, provided selection pressure is not too great.

#### 4.6 Graph Matching

This section describes the extension of the modelling results of sections 4.3, 4.4 and 4.5 to matching attributed relational graphs. Like line labelling, graph matching is a classical consistent labelling problem. The variant of concern here is attributed relational graph matching (FuFu1983). A detailed probabilistic formulation of this problem is given in (Wilson and HancockWilson and Hancock1997). Essentially, a mapping between the nodes of the two graphs is equivalent to a using the nodes in a model graph to label those in a data graph. Table 12 illustrates how the formulation of consistent labelling given in section 2.2 can be adapted for graph matching.

Consistent Labelling	Graph Matching
$\mathbf{V}$	The set of nodes in the data graph
$\mathbf{\Lambda}$	The set of nodes in the model graph
$\mathbf{C}$	The set of neighbourhoods in the data graph
$\Gamma_j$	A labelling of the $j^{\text{th}}$ neighbourhood
$\Theta_j$	The dictionary of the $j^{\text{th}}$ neighbourhood
$\Gamma$	A labelling of the entire data graph

Table 12: The formulation as it applies to graph matching.

The algorithm was tested on 4 pairs of synthetic graphs each containing 30-nodes. An example of the graphs used in our experiments is shown in figure 14. The graphs were synthesized by generating random configurations points on a plane and then

computing their six nearest neighbour graphs. By randomly perturbing the positions of the original points, we have introduced differences into the edge-sets of the graphs. This simulates the type of structural corruption experienced when graphs are extracted from real world images.

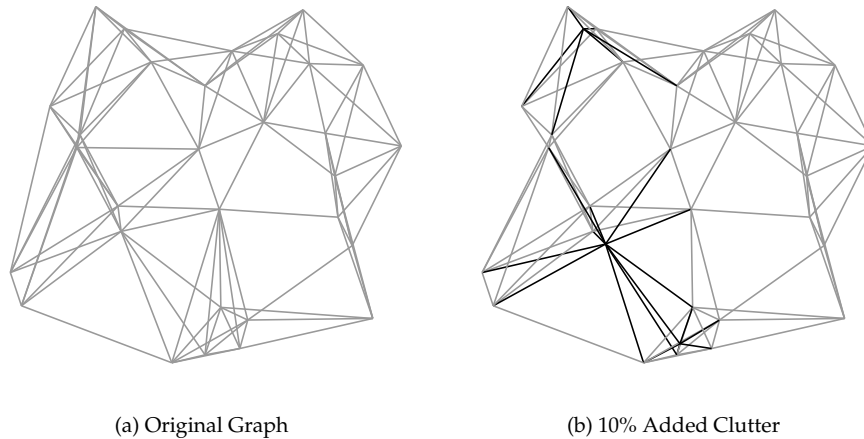


Figure 14: Synthetic Graph with Corruption. Clutter is indicated by dark edges.

#### 4.6.1 Control Variables

Cross et al. have already shown that the local search step is essential when matching graphs (Cross, Wilson, and HancockCross et al.1997). Combining Wilson and Hancock’s observation that the local search step can recover from significant initialisation error (Wilson and HancockWilson and Hancock1997) with a “missing labels” argument similar to that given in section 4.5 suggested that the population size should be around 10. The optimal crossover type (uniform), crossover rate (0.9) and mutation rate (0.3) were determined using the modelling approach from section 4.3. This set of control variables was compared with two sets recommended in the literature (DeJongDeJong1975; GrefenstetteGrefenstette1986). The number of iterations was chosen so that there were 700000 fitness evaluations for each control variable set. The three sets are shown in table 13.

	Set A (Modelling)	Set L1 (DeJongDeJong1975)	Set L2 (GrefenstetteGrefenstette1986)
Population	12	100	30
Iterations	20	3	8
Crossover	Uniform	2 point	Uniform
Cross rate	0.9	0.6	0.9
Mutate rate	0.3	0.001	0.01

Table 13: Control Variable Sets for Graph Matching. Each set required approximately 700,000 fitness evaluations.

Each control variable set was run 100 times on each of the 4 30-node graphs. The

results were pooled to give 400 observations per control variable set. The algorithm's performance was assessed by calculating the average proportion of correct labels in the labellings from the final population. The results are shown in table 14.

Parameter Set	Average Fraction Correct
A (Modelling)	0.93 (0.0041)
L1 (DeJongDeJong1975)	0.73 (0.0049)
L2 (GrefenstetteGrefenstette1986)	0.91 (0.0047)

Table 14: Algorithm Performance. Standard errors are given in parentheses.

The control variables derived by modelling gave the best performance, followed by those suggested in (GrefenstetteGrefenstette1986).

#### 4.6.2 Real Images

Figure 15 shows an office scene taken with a low quality web-cam. Regions were extracted from the grey-scale image pair (panels (a) and (b)) using a simple thresholding technique. Each image contained 50 regions. Graphs were extracted from the scenes by constructing the Delaunay triangulations of the region centroids using the Triangle algorithm (ShewchukShewchuk1996). Examples of the Delaunay graphs are shown in panels (c) and (d) of Figure 15. The Delaunay triangulations were matched using a hybrid genetic algorithm. Panel (e) shows an initial guess in which none of the mappings is correct. Panel (f) shows one the solutions found in the final population. There were 50 regions in the left image of which 42 had feasible correspondences in the right. The amount of relational corruption between the two triangulations was estimated at around 35% by counting the number of inconsistent supercliques given the ground truth match. Despite the significant relational corruption, the final match had 98% correct mappings.

## 5 Conclusion

This paper has addressed the problem of setting genetic algorithm control variables using statistical models fitted to the results of large factorial experiments. The many different factors controlling algorithm behaviour cannot be considered in isolation because they may interact. For example, in section 4.3, it was shown that whether or not the arithmetic mean probabilistic label consistency criterion was better depended on whether or not the algorithm had a local search step.

Unfortunately, the size of the design matrix in a factorial design grows exponentially with the number of levels of each variable included in the experiment. This means that in complex designs only a few levels of each variable can be considered. However,  $n$  levels are sufficient to permit polynomial models up to degree  $n - 1$  to be fitted to the data. In general, polynomial models are of limited use when the model must be extrapolated, so in practice it is not unreasonable to restrict the number of levels considered. The approach adopted in this paper was to focus the factorial experiments on factors whose roles were unclear. Factors such as local search were known *a priori* to have such an important influence that their inclusion in the models would have served little purpose. Conversely, factors which at the outset are thought unlikely to have any substantial effect on the outcome, can also be omitted from the initial models. Although it is important to consider such factors, they need not be explicitly included in models.

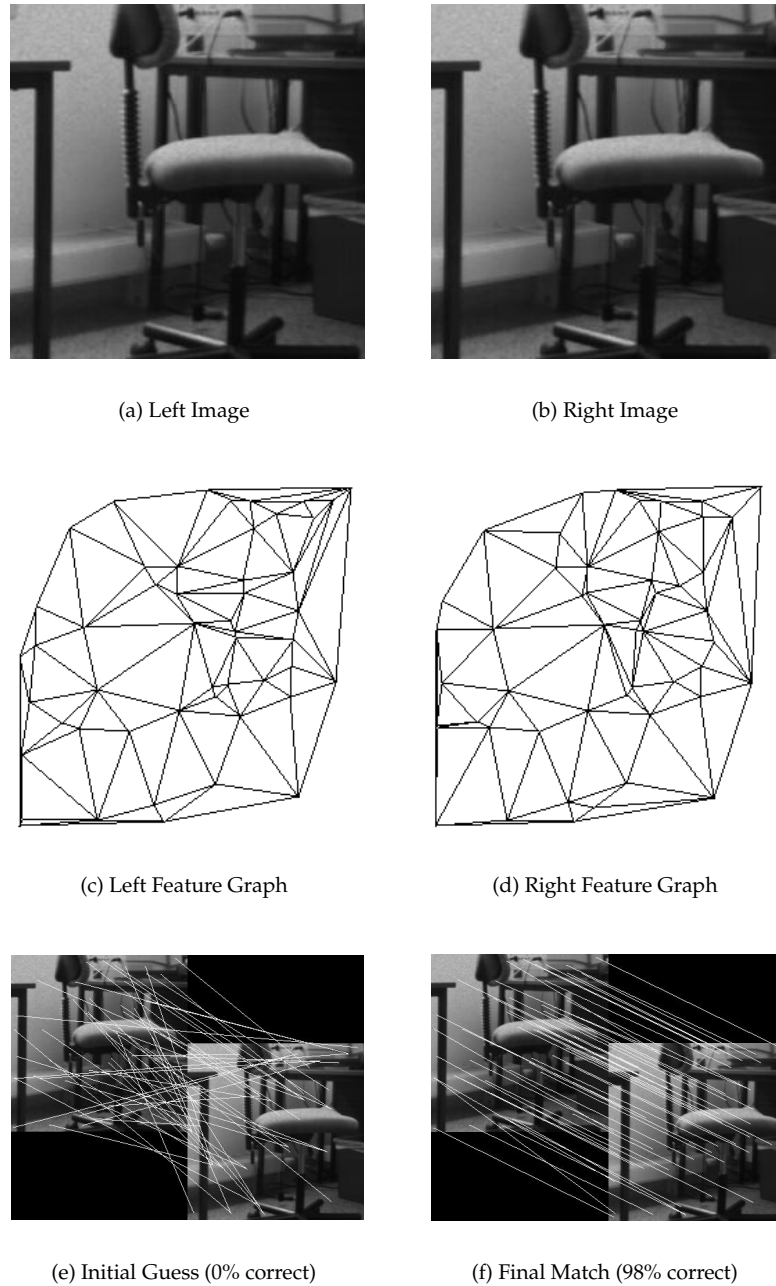


Figure 15: Uncalibrated Stereogram. The camera positions are not known.

When considering a limited number of factor levels, it is necessary to choose those levels with care in order not to miss some interesting area of the parameter space. A reduced factorial design using a graeco-latin square embedded within a full factorial

framework was proposed. This arrangement permits a larger range of factor levels to be considered, although their interactions cannot be modelled. This framework is a useful basis for preliminary studies since it will furnish a rough idea of the main effects.

The analysis of the factorial experiments had three objectives. The first was to find optimal genetic algorithm control variable settings for line labelling problems. Optimal conditions for line labelling are: the addition of a local search step, the use of the arithmetic mean probabilistic labelling criterion, and the use of disruptive crossovers at a high rate. Without local search, the mutation rate should be set no higher than about 0.05. With local search, however, higher mutation rates up to around 0.5 are tolerated. Only a few algorithm iterations were needed.

The second goal of the analysis was to establish the stability of the models under extrapolation. The empirical models for solution quality were found to be reasonably stable under moderate extrapolation with respect to problem size, predicting performance with reasonable accuracy for problems at least three times larger than those initially studied.

The third goal was to model the relationships between control variables and algorithm performance. In general, the most important variable determining algorithm performance was the size of the problem to be solved, with structural effects rather than the number of lines responsible for much of the variation in the data sets. Second to problem size was population size, which can be minimised by observing that the genetic operators need only assemble a good initial guess for local search, which seems to be responsible for most of the optimisation.

It was also shown that the modelling process can be extended to harder labelling problems such as graph matching, where the control variables obtained by modelling outperform those previously suggested in the literature.

There is considerable scope for extending the experimental study reported here and for enhancing the framework. First, only hybrid and standard genetic algorithms have been considered. It would be interesting to analyse other algorithms such as the steady-state GA (SyswerdaSyswerda1989), GENITOR (WhitleyWhitley1989), or CHC (EshelmanEshelman1991). In view of the finding that structural effects in the line drawing predominate, a clearer picture of the role played by the drawing size could be obtained by using several different line drawings of the same size. If these drawings were constructed at random, any bias due to the structural component would be removed. However, constructing random but well-formed line drawings of trihedral scenes would not be trivial. It was not feasible to consider higher than triple interactions in these experiments, although they may well exist. However, the same data set can be re-analysed at a later date if higher order interactions are to be studied.

## References

- D. H. Ackley (1987). *A Connectionist Machine for Genetic Hillclimbing*. Kluwer Academic.
- P. Andrey and P. Tarroux (1994). Unsupervised image segmentation using a distributed genetic algorithm. *Pattern Recognition* 27, 659–673.
- F. J. Anscombe (1953). Contribution to discussion of a paper by H. Hotelling. *Journal of the Royal Statistical Society B15*, 229–230.
- F. J. Aranda-Ordaz (1981). On two families of transformations to additivity for binary response data. *Biometrika* 68, 357–363.

- M. Bedau (1995). Three illustrations of artificial life's working hypothesis. *Lecture Notes in Computer Science* 899, 53–68.
- S. M. Bhandarkar, Y. Zhang, and W. D. Potter (1994). An edge extraction technique using genetic algorithm-based optimization. *Pattern Recognition* 27, 1159–1180.
- M. F. Bramlette (1991). Initialization, mutation and selection methods in genetic algorithms. In R. K. Belew and L. B. Booker (Eds.), *Proceedings of the 4<sup>th</sup> International Conference on Genetic Algorithms*. Morgan Kaufmann.
- H. J. Bremermann (1958). The evolution of intelligence. The nervous system as a model of its environment. Technical Report 477(17), Department of Mathematics, University of Washington.
- M. B. Clowes (1971). On seeing things. *Artificial Intelligence* 2, 79–116.
- W. G. Cochran and G. M. Cox (1957). *Experimental Designs* (2<sup>nd</sup> ed.). Wiley.
- D. Collett (1991). *Modelling Binary Data*. Chapman and Hall.
- S. A. Cook (1971). The complexity of theorem proving procedures. In *Proceedings of the 3<sup>rd</sup> ACM Symposium on the Theory of Computing*, pp. 151–158.
- A. D. J. Cross, R. Myers, and E. R. Hancock (2000). Convergence of a hill-climbing genetic algorithm. *Pattern Recognition* 33, 1863–1880.
- A. D. J. Cross, R. C. Wilson, and E. R. Hancock (1997). Inexact graph matching using genetic search. *Pattern Recognition* 30, 953–970.
- L. Davis (1989). Adapting operator probabilities in genetic algorithms. In J. D. Schaffer (Ed.), *Proceedings of the 3<sup>rd</sup> International Conference on Genetic Algorithms*, pp. 61–69.
- L. S. Davis (1991). *A Handbook of Genetic Algorithms*. Van Nostrand Reinhold.
- K. A. DeJong (1975). *An Analysis of the Behaviour of a Class of Genetic Adaptive Systems*. Ph. D. thesis, Department of Computer and Communication Sciences, University of Michigan.
- L. J. Eshelman (1991). The CHC adaptive search algorithm: How to have safe search when engaging in nontraditional genetic recombination. In G. J. E. Rawlins (Ed.), *Foundations of Genetic Algorithms*, Volume 1, pp. 265–283. Morgan Kaufmann.
- O. D. Faugeras and M. Berthod (1981). Improving consistency and reducing ambiguity in stochastic labeling: An optimisation approach. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 3, 412–424.
- D. B. Fogel (1994). An introduction to simulated evolutionary optimisation. *IEEE Transactions on Neural Networks* 5, 3–14.
- B. Francis, M. Green, and C. Payne (Eds.) (1993). *The GLIM System Release 4 Manual*. Oxford University Press.
- A. S. Fraser (1957). Simulation of genetic systems by automatic digital computers. *Australian Journal of Biological Science* 10, 484–491.

- K. S. Fu (1983). A step towards unification of syntactic and statistical pattern recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 5, 200–205.
- M. R. Garey and D. S. Johnson (1979). *Computers and Intractability*. Freeman.
- D. Geiger and F. Girosi (1991). Parallel and deterministic algorithms from MRFs: Surface reconstruction. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 13, 401–412.
- S. Geman and D. Geman (1984). Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 6, 721–741.
- D. Goldberg (1989). *Genetic Algorithms in Search, Optimisation and Machine Learning*. Addison-Wesley.
- D. E. Goldberg and J. Richardson (1987). Genetic algorithms with sharing for multimodal function optimization. In *Proceedings of the 2<sup>nd</sup> International Conference on Genetic Algorithms*, pp. 41–49.
- J. J. Grefenstette (1986). Optimisation of control parameters for genetic algorithms. *IEEE Transactions on Systems, Man and Cybernetics* 16, 122–128.
- J. J. Grefenstette (1993). Deception considered harmful. In D. Whitley (Ed.), *Foundations of Genetic Algorithms*, Volume 2. Morgan Kaufmann.
- E. R. Hancock (1994). An optimisation approach to line labelling. In S. Impedovo (Ed.), *Progress in Image Analysis and Processing*, Volume 3, pp. 159–165. World Scientific.
- E. R. Hancock and J. Kittler (1990a). Discrete relaxation. *Pattern Recognition* 23, 711–733.
- E. R. Hancock and J. Kittler (1990b). Edge labelling using dictionary-based relaxation. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 12, 165–181.
- R. M. Haralick, L. S. Davis, and A. Rosenfeld (1978). Reduction operations for constraint satisfaction. *Information Science* 14, 199–219.
- R. M. Haralick and L. G. Shapiro (1979). The consistent labelling problem: Part 1. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 1, 173–184.
- R. M. Haralick and L. G. Shapiro (1980). The consistent labelling problem: Part 2. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 2, 193–203.
- W. L. Hays (1994). *Statistics* (5<sup>th</sup> ed.). Harcourt Brace.
- J. H. Holland (1975). *Adaptation in Natural and Artificial Systems*. MIT Press.
- D. A. Huffman (1971). Impossible objects as nonsense sentences. In B. Meltzer and D. Michie (Eds.), *Machine Intelligence*, Volume 6, pp. 295–323. Edinburgh University Press.
- R. A. Hummel and S. W. Zucker (1983). On the foundations of relaxation labeling processes. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 5, 267–287.
- P. Jeavons (1998). On the algebraic structure of combinatorial problems. *Theoretical Computer Science* 200, 185–204.

- D. Jefferson, R. Collins, C. Cooper, M. Dyer, M. Flowers, R. Korf, C. Taylor, and A. Wang (1991). Evolution as a theme in artificial life: The genesys/tracker system. In C. G. Langton, C. Taylor, J. D. Farmer, and S. Rasmussen (Eds.), *Artificial Life II*. Addison-Wesley.
- S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi (1983). Optimisation by simulated annealing. *Science* 220, 671–680.
- J. R. Koza (1992). *Genetic Programming*. MIT Press.
- H. Lipson and M. Shpitalni (1996). Optimization-based reconstruction of a 3D object from a single freehand line drawing. *CAD* 28, 651–663.
- S. A. Lloyd (1983). An optimisation approach to relaxation labelling algorithms. *Image and Vision Computing* 1, 85–91.
- A. K. Mackworth (1977). Consistency in networks of relations. *Artificial Intelligence* 8, 99–118.
- A. K. Mackworth and E. C. Freuder (1985). The complexity of some polynomial network consistency algorithms. *Artificial Intelligence* 25, 65–74.
- P. McCullagh and J. A. Nelder (1989). *Generalised Linear Models* (2<sup>nd</sup> ed.). Chapman and Hall.
- M. Mitchell (1996). *An Introduction to Genetic Algorithms*. MIT Press.
- J. L. Mohammed, R. A. Hummel, and S. W. Zucker (1983). A gradient projection algorithm for relaxation methods. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 5, 330–332.
- H. Mühlenbein (1994). Genetic algorithms. In E. Aarts and J. K. Lenstra (Eds.), *Local Search in Combinatorial Optimisation*. Wiley.
- H. Mühlenbein and D. Schlierkamp-Voosen (1995). Analysis of selection, mutation and recombination in genetic algorithms. *Lecture Notes in Computer Science* 899, 142–168.
- B. Nudel (1983). Consistent labelling problems and their algorithms. *Artificial Intelligence* 21, 135–178.
- W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery (1992). *Numerical Recipes in C* (2<sup>nd</sup> ed.). Cambridge University Press.
- A. Prügel-Bennett and J. L. Shapiro (1994). An analysis of genetic algorithms using statistical physics. *Physical Review Letters* 72, 1305–1309.
- X. Qi and F. Palmieri (1994a). Theoretical analysis of evolutionary algorithms with an infinite population size in continuous space, part 1: Basic properties of selection and mutation. *IEEE Transactions on Neural Networks* 5, 102–119.
- X. Qi and F. Palmieri (1994b). Theoretical analysis of evolutionary algorithms with an infinite population size in continuous space, part 2: Analysis of the diversification rôle of the crossover. *IEEE Transactions on Neural Networks* 5, 120–129.

- I. Rechenberg (1973). *Evolutionsstrategie - Optimierung Technischer Systeme nach Prinzipien der biologischen Information*. Fromman Verlag.
- J. Reed, R. Toombs, and N. A. Barricelli (1967). Simulation of biological evolution and machine learning. *Journal of Theoretical Biology* 17, 319–342.
- A. Rosenfeld, R. A. Hummel, and S. W. Zucker (1976). Scene labelling by relaxation operations. *IEEE Transactions on Systems, Man and Cybernetics* 6, 420–433.
- G. Rudolph (1997). *Convergence Properties of Evolutionary Algorithms*. Kovač.
- H. Saito and M. Mori (1995). Application of genetic algorithms to stereo matching of images. *Pattern Recognition Letters* 16, 815–822.
- J. D. Schaffer, R. A. Caruna, L. J. Eshelman, and R. Das (1989). A study of control parameters affecting online performance of genetic algorithms for function optimisation. In *Proceedings of the 3<sup>rd</sup> International Conference on Genetic Algorithms*, pp. 51–60.
- H. Schwefel (1981). *Numerical Optimization of Computer Models*. Wiley.
- J. R. Shewchuk (1996). Triangle: Engineering a 2D quality mesh generator and Delaunay triangulator. In *Proceedings of the 1<sup>st</sup> Workshop on Applied Computational Geometry*, pp. 124–133.
- M. Srinivas and L. M. Patnaik (1994). Adaptive probabilities of crossover and mutation in genetic algorithms. *IEEE Transactions on Systems, Man and Cybernetics* 4, 656–667.
- G. Syswerda (1989). Uniform crossover in genetic algorithms. In *Proceedings of the 3<sup>rd</sup> International Conference on Genetic Algorithms*, pp. 2–9.
- P. W. M. Tsang (1997). A genetic algorithm for affine invariant recognition of object shapes from broken boundaries. *Pattern Recognition Letters* 18, 631–639.
- M. D. Vose (1995). Modelling simple genetic algorithms. *Evolutionary Computation* 3, 453–472.
- D. Waltz (1975). Understanding line drawings of scenes with shadows. In P. H. Winston (Ed.), *The Psychology of Computer Vision*, pp. 19–91. McGraw-Hill.
- D. Whitley (1989). The GENITOR algorithm and selection pressure: Why rank-based allocation of reproductive trials is best. In *Proceedings of the 3<sup>rd</sup> International Conference on Genetic Algorithms*, pp. 116–121.
- D. Whitley, R. Beveridge, C. Graves, and K. Mathias (1995). Test driving three 1995 genetic algorithms: New test functions and geometric matching. *Journal of Heuristics* 1, 77–104.
- G. N. Wilkinson and C. E. Rogers (1973). Symbolic description of factorial models for analysis of variance. *Applied Statistics* 22, 392–399.
- R. C. Wilson and E. R. Hancock (1997). Structural matching by discrete relaxation. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 19, 634–648.
- A. L. Yuille and J. J. Kosowsky (1994). Statistical physics algorithms that converge. *Neural Computation* 6, 341–356.