Empirical Modelling of Genetic Algorithms

Richard Myers rich@cs.york.ac.uk
Department of Computer Science, University of York, York, Y01 5DD, UK

Edwin R. Hancock erh@cs.york.ac.uk
Department of Computer Science, University of York, York, Y01 5DD, 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 derived can be used to determine optimal algorithm parameters and to shed light on interactions between the parameters and their relative importance. 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 optimization methods based on the concept of Darwinian evolution in populations (Holland, 1975; Goldberg, 1989; Fogel, 1994; Mitchell, 1996). Evolutionary algorithms have been proposed by several authors (Fraser, 1957; Bremermann, 1958; Reed et al., 1967; Holland, 1975; Rechenberg, 1973; Schwefel, 1981). Holland's (1975) formulation is regarded as the standard for genetic algorithms, although there are others (e.g., evolutionary strategies (Schwefel, 1981)). Such algorithms have applications in many disciplines including artificial life (Jefferson et al., 1991) and evolutionary programming (Koza, 1992). In the field of computer vision, genetic algorithms have been used for a host of applications, such as image segmentation (Andrey and Tarroux, 1994), object recognition (Tsang, 1997), stereo matching (Saito and Mori, 1995), edge extraction (Bhandarkar et al., 1994), and graph matching (Cross et al., 1997).

Many factors influence the behavior 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
- the existence of 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,
- the crossover operator used,
- the selection operator used,
- the population size,
- the terminating criterion,
- the crossover rate, and
- the mutation rate.

Additional procedures, such as fitness sharing (Goldberg and Richardson, 1987) or otherwise constraining the selection operator (Goldberg, 1989), may increase the number of parameters. The result is that applying a genetic algorithm to a problem, while simple in principle, can be difficult in practice (Davis, 1991). This issue can be addressed by theoretical modelling, empirical modelling, or ad-hoc experimental approaches to parameter setting.

There has also been interest in applying mathematical models to evolutionary optimization (Mühlenbein, 1994; Mühlenbein and Schlierkamp-Voosen, 1995; Bedau, 1995). However, general theoretical models have limited use in practical settings. 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 earlier, any theoretical model that predicts parameter values should consider the nature of the problem. By taking advantage of specific problem features, better quantitative predictions about the behavior of the algorithm have been made (Cross et al., 2000; Prügel-Bennett and Shapiro, 1994).

The alternative to theoretical modelling is empirical modelling. Few substantial experimental studies of genetic algorithms have been undertaken. De Jong (1975) considered a suite of five numerical optimization problems, at least three of which are relatively easy (Whitley et al., 1995). Problems from De Jong's test suite are still used to test and compare genetic algorithms, an approach criticized in Whitley et al. (1995). Schaffer et al. (1989) presented a significant, large-scale, experimental study of genetic algorithms. However, their experiments were mostly based on numerical test problems and have been shown in Mühlenbein (1994) to be unreliable under extrapolation. It is important to stress that the extrapolation of a regression model within the same problem domain should be cautiously undertaken. Moreover, the extrapolation of the model outside the problem domain is often meaningless.

There have been attempts to determine genetic algorithm parameters experimentally without modelling (Grefenstette, 1986; Bramlette, 1991; Davis, 1989, 1991). These attempts used genetic algorithms to optimize the parameters in a bootstrapping framework. One area that has received little 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 attempting to generalize them. We feel there is no substitute for empirical modelling when seeking to determine appropriate values for algorithm parameters. However, there is no reason to suppose a genetic algorithm will behave in the same way for different problem classes (e.g., continuous numerical optimization vs. (discrete) combinatoric optimization). There is also no reason to suppose that parameters that work for numerical optimization problems will also work for symbolic ones.

The main contribution of this paper is the development of specific, empirical models of genetic algorithm performance for consistent labelling problems. We look at the well-researched line labelling problem in detail (Huffman, 1971; Clowes, 1971; Waltz, 1975; Lipson and Shpitalni, 1996). The empirical models are based on factor analysis. Our aim is to use factor analysis to determine the best combination of genetic algorithm parameters for consistent labelling problems. We fit generalized linear models (McCullagh and Nelder, 1989) to the data. The empirical models that result from our factor analysis (Cochran and Cox, 1957) summarize the outcome of 500,000 program runs. The 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.

An additional motivation for this paper is that consistent labelling encompasses problems ranging from the trivial to the intractable. Ackley's (1987) ONEMAX problem 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. Because labelling problems have been extensively studied (Mackworth, 1977; Haralick and Shapiro, 1979, 1980; Haralick et al., 1978; Nudel, 1983; Jeavons, 1998), they might provide a realistic framework in which to further investigate genetic algorithm behavior.

The outline of this paper is as follows. Section 2 discusses consistent labelling and introduces and formulates the line labelling problem. Section 3 describes the experimental design and methods. Section 4 analyzes the results of full factorial experiments by fitting generalized linear models. Section 5 presents 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 that exist between (subsets of) these objects. It is formally equivalent to the general satisfiability problem and is NP-complete in its general form (Nudel, 1983; Garey and Johnson, 1979; Cook, 1971; Haralick 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 (Mackworth, 1977; Haralick and Shapiro, 1979, 1980).

Early solutions to this problem involved searching a configuration space, which can be pruned using discrete relaxation operators (Mackworth, 1977; Mackworth and Freuder, 1985; Waltz, 1975). However, these methods did not use evidence derived from the scene, relying merely on pre-defined constraint relations. To overcome this problem, Rosenfeld et al. (1976) formulated probabilistic relaxation. Much of the work involving consistent labelling has adopted Hummel and Zucker's optimization paradigm (Hummel and Zucker, 1983; Faugeras and Berthod, 1981; Lloyd, 1983; Mohammed et al., 1983; Wilson and Hancock, 1997). The problem is to find a set of label assignments that optimizes 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 optimization technique is gradient ascent, in which

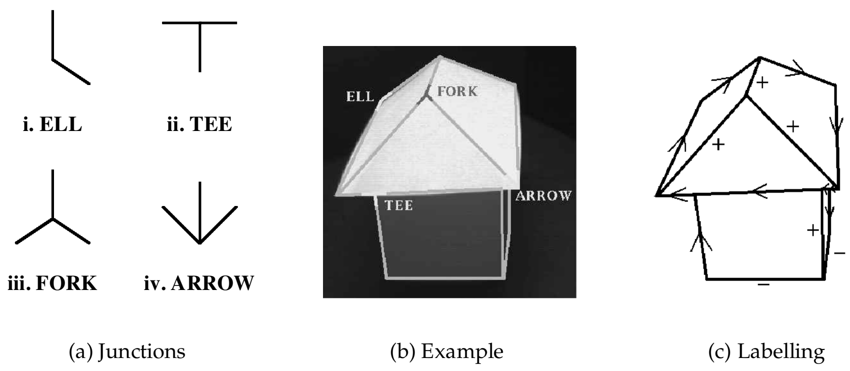


Figure 1: The four junction types defined by Huffman (1971).

the update operator is required to monotonically increase the quality of the solution: this was the approach used by Hummel and Zucker (1983) and others (Hancock and Kittler, 1990a; Faugeras and Berthod, 1981; Lloyd, 1983; Mohammed et al., 1983; Wilson and Hancock, 1997). 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 optimization techniques such as simulated annealing (Kirkpatrick et al., 1983; Geman and Geman, 1984), mean field annealing (Geiger and Girosi, 1991; Yuille and Kosowsky, 1994), or genetic search (Holland, 1975).

2.1 Line Labelling

Line drawing interpretation was independently formulated in the mid 1970s (Huffman, 1971; Clowes, 1971). This work led Waltz to his seminal discrete relaxation algorithm (Huffman, 1971; Clowes, 1971; Waltz, 1975). Waltz showed 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 document analysis, processing architects' sketches, and automatic interpretation of engineering drawings.

Trihedral polyhedra are solid shapes whose vertices may involve up to three faces. Drawings representing scenes composed of trihedral polyhedra contain four types of line junctions. The junctions are classified by the shape of their topologies: ELL, TEE, FORK, or ARROW. 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 ($\leftarrow$ and $\rightarrow$) (the occluding surface is always to the right as one follows the arrow). An example is given in Figure 1. Waltz (1975) 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.

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

Consistent Labelling	Line Labelling
$\mathbf{V}$	The set of lines in the drawing
$\mathbf{\Lambda}$	The labels, $+$, $-$, $\leftarrow$ and $\rightarrow$
$\mathbf{C}$	The set of junctions in the drawing
γ_j	A labelling of the j^{th} junction
Θ_j	The (Waltz) dictionary of the j^{th} junction
$\mathbf{\Gamma}$	A labelling of the entire 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 $\mathbf{\Lambda}$, a set of neighborhoods $\mathbf{C}$ can be constructed. Each element C_j of $\mathbf{C}$ is defined as $C_j =< i \mid i \in \mathbf{V} \wedge \mathbf{Conn}(j, i) >$, where **Conn** is the connectivity relation that is reflexive but not necessarily symmetric or transitive. A labelling γ_j of the jth neighborhood is a list of labels applied to its constituent objects, $\gamma_j \subset C_j \times \mathbf{\Lambda}$. Each neighborhood has a dictionary Θ_j of legal labellings. The dictionaries capture the structure of the problem.[1] Waltz filtering (Waltz, 1975) in this context would remove globally inconsistent labellings from the dictionaries Θ_j. The consistency of a labelling of all the objects, i.e., the global configuration of line-label assignments $\mathbf{\Gamma} \subset V \times \Lambda$, can be determined by considering separately the consistency of the labellings of the neighborhoods $\mathbf{\Gamma}_j$. Table 1 illustrates this formulation in the case of line labelling.

2.2.1 Probabilistic Labelling Criterion

Hancock and Kittler (1990a) built on the work of others (Faugeras and Berthod, 1981; Hummel and Zucker, 1983) by developing a probabilistic framework for measuring consistency. This framework couples an explicit dictionary representation of constraints, as adopted by Waltz (1975), 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 allows both statistical and structural considerations to influence the labelling process. Scene interpretation is achieved by finding the label configuration that optimizes the combined probability criterion. This was originally done by gradient ascent (Hancock and Kittler, 1990a). This approach has been applied to scene labelling (Hancock and Kittler, 1990a), edge detection (Hancock and Kittler, 1990b), graph matching (Wilson and Hancock, 1997), and line labelling (Hancock, 1994).

Hancock and Kittler developed a label-error process that can be used to compute the probability of an observed label configuration γ_j given a dictionary of permissible labellings of the same configuration of objects. Suppose that S_i is the ith configuration of permissible labels in the dictionary Θ_j of the junction indexed j. The model commences from the assumption that the label errors occur at different line sites in a

[1]If all the neighborhoods are the same size, this is equivalent to Haralick and Shapiro's (1979, 1980) original formulation of the consistent labelling problem as a network constraint satisfaction problem. The neighborhoods are the "unit-constraint relation," and the dictionaries are the "unit-label constraint relation."

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 γ_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 γ_j and the junction dictionary item S_i. Using Bayes theorem, Hancock and Kittler show that the probability of observing the label configuration γ_j for the junction with label-configuration dictionary Θ_j is given by

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

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 Kittler, 1990a; Wilson and Hancock, 1997). This is analogous to the annealing of the temperature parameter in simulated annealing. The best strategy found has been to reduce P_e by a constant factor with time.

To obtain a global measure of label consistency Γ, 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 neighborhoods). The resulting measure of label consistency is

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

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:

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

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

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

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

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

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.,

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

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 Γ 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: 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

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 (Eshelman, 1991) 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 consists 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, so parameters that 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. 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 behavior and the control variable settings.

	1	2	3
I	A_α	B_γ	C_β
II	B_β	C_α	A_γ
III	C_γ	A_β	B_α

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 α, β, and γ.

3.1 Design

In our experiments, all relevant explanatory variables are examined simultaneously. Such experimental designs are called *factorial*. Factorial designs have two major advantages over other designs (see Cochran and Cox (1957, 150–151)). 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 (Cochran and Cox, 1957). Other authors argue that a factorial design can serve as a summary of the data (Hays, 1994). We take the view that it is worth using the largest feasible design and then simplifying.

A cheaper alternative to a fully crossed, balanced factorial design is a fractionally replicated factorial or graeco-latin square design (see 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 α, β, and γ.) There are no pairwise correlations between the explanatory variables in this design. This allows the investigation of individual effects of explanatory variables using fewer combinations. The price of reducing the dimensionality in this way is an inability to consider interactions. 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 emphasizing 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.

3.2 Method

We study a simple generational algorithm, c.f. Goldberg (1989). Each individual will be represented as a string of labels. 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 ith global label configuration appearing in the population by $\Gamma^{<i>}$. The population size is x_P. The fitness function is

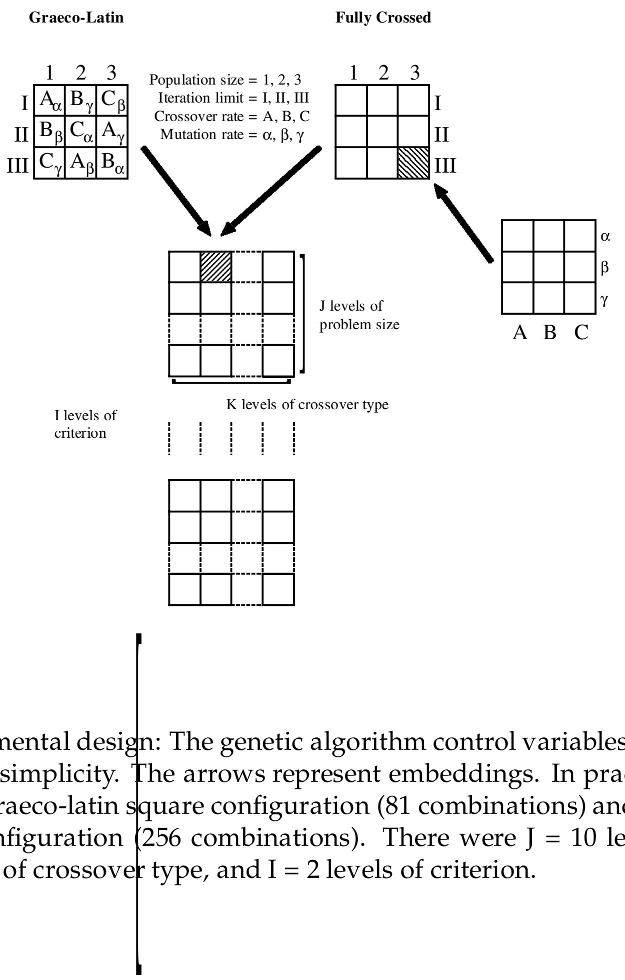


Figure 3: Experimental design: The genetic algorithm control variables are shown with 3 levels each for simplicity. The arrows represent embeddings. In practice, there were 9 levels for the graeco-latin square configuration (81 combinations) and 4 levels for the fully crossed configuration (256 combinations).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

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

In these experiments, we 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 labellings for which $E(\Gamma) = 0$. Thus, algorithm performance can be measured by a binary variable that 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). Analyzing 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

Generalized linear models (McCullagh and Nelder, 1989) were fitted to the data using NAg's GLIM Version 4 update 8 (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 that transforms r so that it matches the range of the linear predictor η, and ϵ 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 η, which in turn depends on the explanatory variables as follows:

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

where J is the number of explanatory variables, and β_j is the coefficient of the jth explanatory variable x_j. It is conventional to take $x_0 = 1$ so that β_0 represents a constant offset. Such models are known as *generalized linear models* (McCullagh and Nelder, 1989) because they are linear in the parameters β 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 η, 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:

- The logistic or logit link, $\mathrm{logit}(r) = \ln\left(\frac{r}{1-r}\right)$, which has a natural interpretation as the log-odds of the event associated with r.
- The probit link, $\mathrm{probit}(r) = \Phi^{-1}(r)$, where $\Phi(a)$ is the standardized normal integral from $-\infty$ to a.
- The complementary log-log link, $\mathrm{cll}(r) = \ln[-\ln(1-r)]$.
- The parameterized link function introduced by Aranda-Ordaz (1981), 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, $g(r;1) \equiv \mathrm{logit}(r)$. Second, when $\lim_{\alpha \to 0} g(r;\alpha) = \mathrm{cll}(r)$, the complementary log-log link function results.

The model algebra developed by Wilkinson and Rogers (1973) will be used to describe the form of the linear predictor. The notation is shown in Table 2. Thus the notation

```
1+A*B+C<2>
```
(10)

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

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 $\beta_{A1} X_A^1 + ... + \beta_{Ak} X_A^k$
M`**n`	represents the cartesian product $\mathbf{M}^n$

corresponds to the model

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

where the coefficients γ 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 β (and γ) 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 (Collett, 1991).

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 χ^2 on the residual degrees of freedom (d.f.) of the current model. (See Collett (1991) and Francis et al. (1993) for a detailed discussion.) The general method adopted in this study is 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 χ^2 on the difference in their d.f. (Collett, 1991; Francis 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 removed or added to a model, their significance being judged on the basis of χ^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 Collett (1991). In this test, the link function with the lowest

Table 3: Control variables for preliminary study.

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	$0.001, 0.005, 0.01, 0.02, 0.05, 0.075, 0.1, 0.2, 0.3$

deviance gives the best fit. For the Aranda-Ordaz link function, the constructed variable technique described in Collett (1991) 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 (Anscombe, 1953; Collett, 1991; Francis 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 Collett (1991) and Francis et al. (1993).

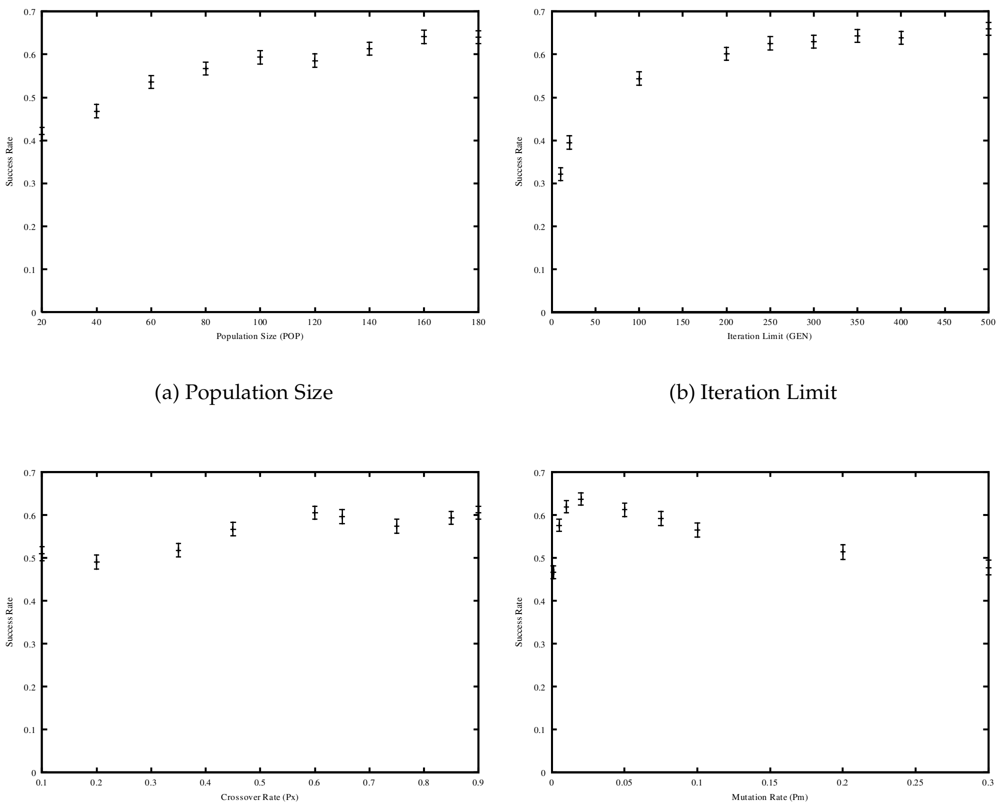
4 Experimental Results and Analysis

This section presents and analyzes the results of the modelling experiments described in the previous section. Section 4.1 describes a preliminary study using graeco-latin squares. Section 4.2 gives brief details of the full factorial experiments. These experiments are analyzed in Sections 4.3 (optimal performance), 4.4 (robustness), and 4.5 (algorithm behavior). 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. The total number of observations was 2x4x10x9x9 = 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 ("LINES") will be used in the text, and the variable names ("x_L") will be used in the formulae. For non-ordinal variables, switch variables such as "w_X" will be defined in formulae.

The results of the experiments must be analyzed 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



(a) Population Size (b) Iteration Limit

(c) Crossover Rate (d) Mutation Rate

Figure 4: Main effects of genetic algorithm control variables.

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 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.

Table 4: Factors in the line labelling experiments. There are $2 \times 4 \times 4 \times 4 \times 4 \times 4 \times 4 = 8192$ combinations.

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	$0.01, 0.02, 0.03, 0.04$

4.2 Factorial Experiments

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 summarized in Table 4.

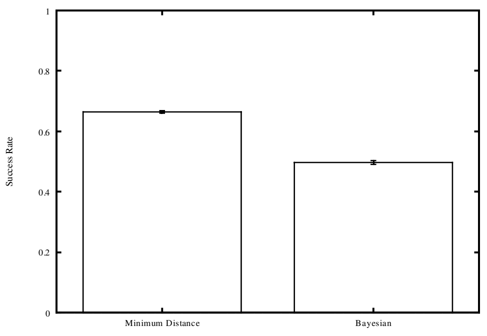
4.3 Modelling Optimal Performance

4.3.1 Standard versus Hybrid Algorithm

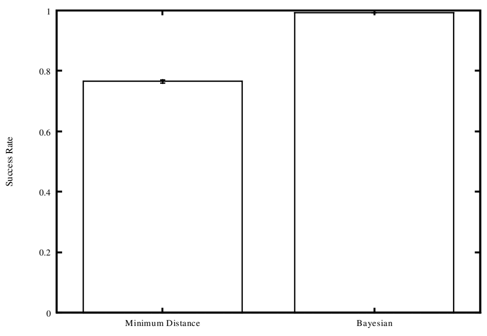
In these experiments, the addition of a local search step was found to greatly improve algorithm performance: it achieved a higher success rate with one fifth of the population size and one twenty-fifth of the iterations when compared to the plain algorithm. This section considers some contrasts between the plain algorithm and the hybrid version. Perhaps the most interesting differences 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 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 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

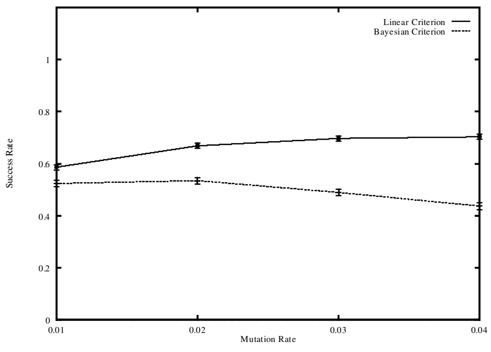


(a) Plain Algorithm

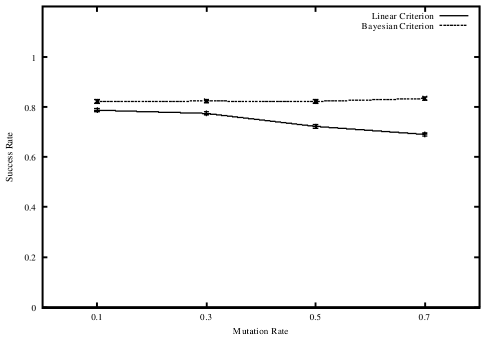


(b) Hybrid Algorithm

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.



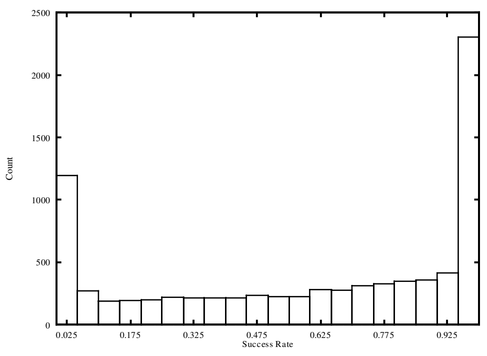
(a) Plain Algorithm (mutation rate in {0.01,0.04})



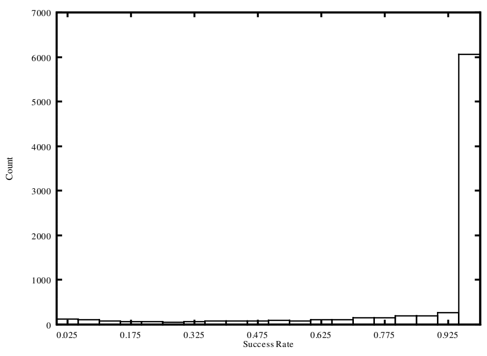
(b) Hybrid Algorithm (mutation rate in {0.1,0.7})

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.

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.



(a) Plain Algorithm



(b) Hybrid Algorithm

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.

Table 5: Saturated factor levels for hybrid algorithm. The values are summarized 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 that had 20 out of 20 successes for each factor level.

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)

4.3.2 Local Search Hybrid

As the histogram in Figure 7(b) 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 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

$$\begin{aligned}&\texttt{1+(COST+CROSS+LINES+POP+GEN+}P_x\texttt{+}P_m\texttt{)**2}\\&\texttt{+(COST+CROSS)*(LINES<2>+POP<2>+GEN<2>+}P_x\texttt{<2>+}P_m\texttt{<2>),}\end{aligned} \tag{12}$$

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.

	LINES			
COST	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 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.

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

which has a deviance[2] of 6776.9 on 8124 d.f.. The model contains 68 terms, which suggests that it is somewhat over-fitted.

The analysis of deviance for the interactions of COST in model 12 is given in Table 7. Table 7 shows that in addition to an obvious main effect, four out of 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.

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 that are linear in the explanatory variables: there is insufficient variation in the data set to support more complex models. The first model is

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

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.

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

[2]See Section 3.3.

Table 8: Parameter estimates for CROSS. The t-value is the ratio of the estimate to its standard error $\hat{\beta}/\mathrm{s.e.}(\hat{\beta})$. For binomial data, the t-value has a standard normal distribution under the null hypothesis, $\beta = 0$ (Collett, 1991). The significance test is one-tailed.

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	$4.700\text{x}10^{-14}$	0.1275	$3.687\text{x}10^{-13}$	1.00

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.

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

function, given in Table 9, shows that the probit link gave the best fit. The shape of the curves in Figure 4(a) and (b) suggests 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

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

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.

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

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

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,

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.

LINES	Simplex Optimum					Predicted 95%	Actual p
	POP	GEN	P_x	P_m	$\hat{p}$	Conf. Int. for p	(N=100)
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

numerical optimization techniques such as the downhill simplex method can be used (see Press et al. (1992, 408–412)). 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 P_m. The required population sizes are at odds with both the predictions of Equation 21 and one's experience with genetic algorithms. On 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 is reason 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 that is logarithmic in LINES rather than in POP and GEN. Such a model was fitted, and its functional form is given in Equation 16. It also includes interaction terms that 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

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.

LINES	Simplex Optimum					Predicted 95%	Actual p
	POP	GEN	P_x	P_m	$\hat{p}$	Conf. Int. for p	(N=100)
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

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.

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

The surface can be visualized 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.

4.5 Algorithm Behavior

In this section, we examine the role of 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:

```
1+CROSS*(LINES<3>+ln(POP)<3>+ln(GEN)<3>+ Px<3>+Pm<3>
+(LINES+ln(POP)+ln(GEN)+ Px+Pm)**2)                        (17)
+(LINES+ln(POP)+ln(GEN)+ Px+Pm)**3
```

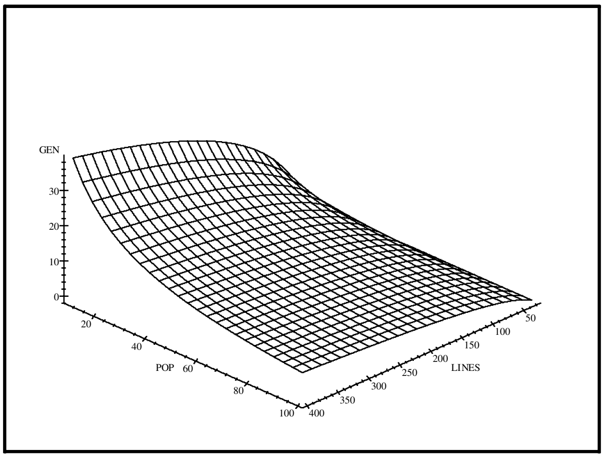


Figure 8: Plot of model 16. Number of iterations (GEN) required to achieve a success rate of 0.99 is plotted against LINES and POP. P_x and P_m are fixed at 0.8 and 0.03.

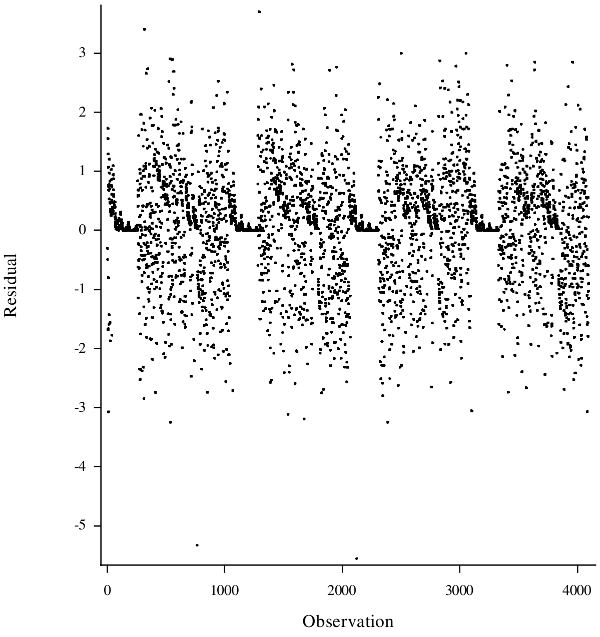


Figure 9: Anscombe Residuals for model 18. The periodic dense regions in the plot show that the model over-fits the 9-line problem. Apart from that, there is no pattern to the residuals.

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

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

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 χ^2 on 34 d.f. and found to be insignificant. This is the simplest model that 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.

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

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

This is a very complex six-dimensional function. The only way to visualize 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 Figure 11(a) and (b) for POP and GEN.

Although this model provides a good statistical explanation of the data, there remains 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 that appears in Equation 19 ($\ldots -1.213x_L + 0.05144x_L^2 - 0.0007183x_L^3 + \ldots$). 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 that 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 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 Figure 11(c).

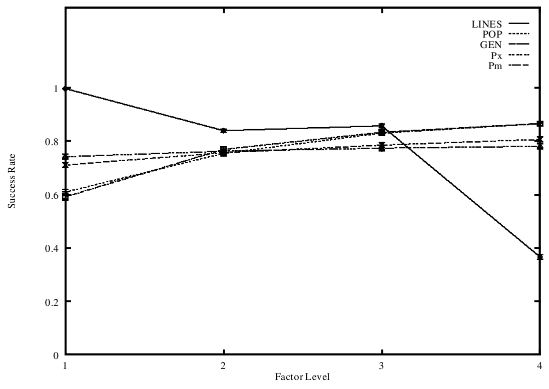


Figure 10: Success rate plots. A larger slope indicates that a particular variable has a greater effect on the success rate. The ranges of the variables must also be taken into account – for example, only $\frac{1}{25}$th of the range of mutation rate is explored.

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 is 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.

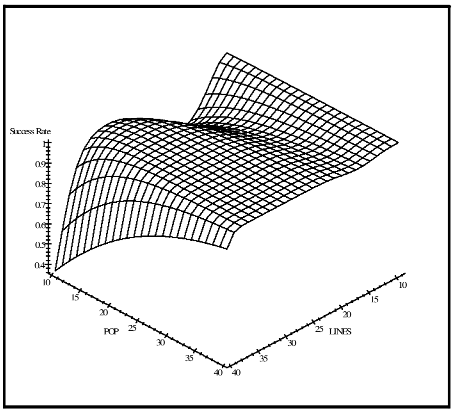
The quality of the initial population can be assessed in terms of the number of loci at which all labels appear. For example, an initial population in which every individual incorrectly assigned the label + to the same line is unlikely to locate an optimal solution quickly. A much better initial population 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

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

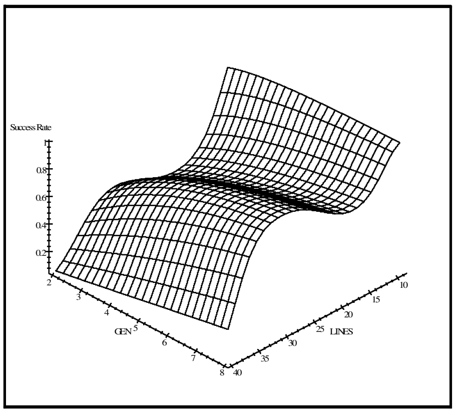
This is shown in Figure 12, together with the expected number of missing labels (LINES $\times$ p_a), as a function of LINES and POP. For smaller problems, these are effectively zero for population sizes larger than 30. This agrees with the results of the preliminary study and suggests that POP should 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:

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

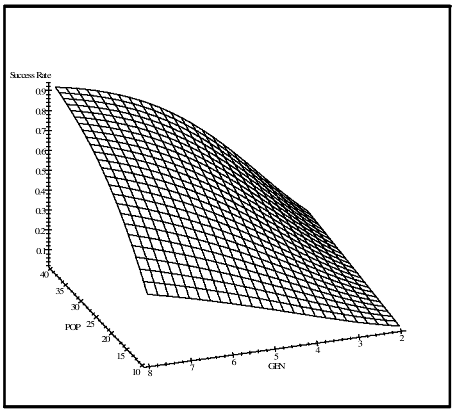
where $|\mathbf{\Lambda}|$ is the size of the label set (4 in this case). The optimal population size clearly increases with LINES with no upper bound. This formula is not amenable to direct ma-



(a) Success vs. LINES and POP



(b) Success vs. LINES and GEN

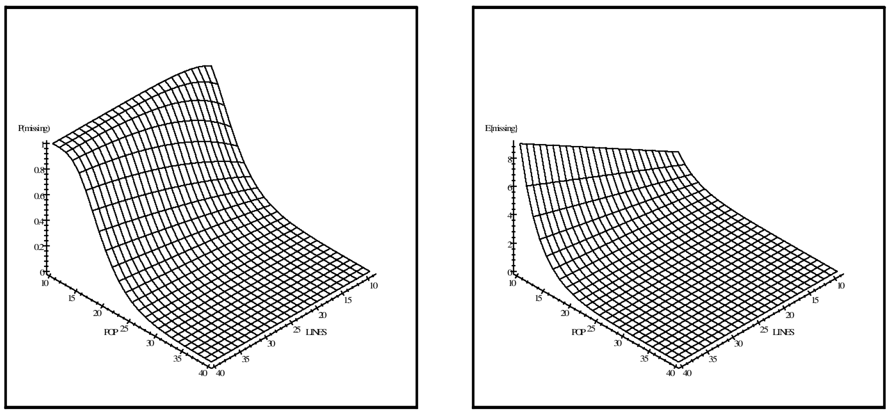


(c) Success vs. POP and GEN

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.

nipulation, 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



(a) Probability

(b) Expectation

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.

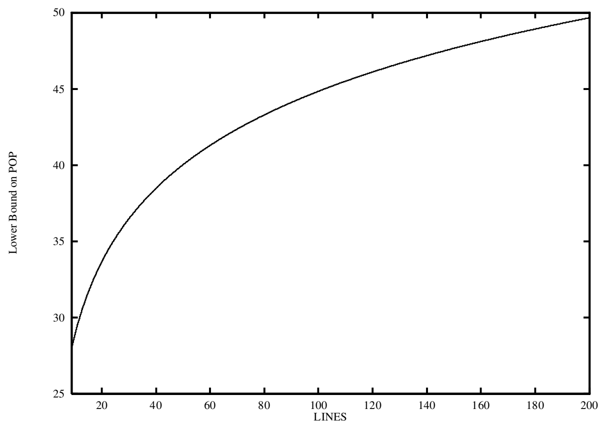
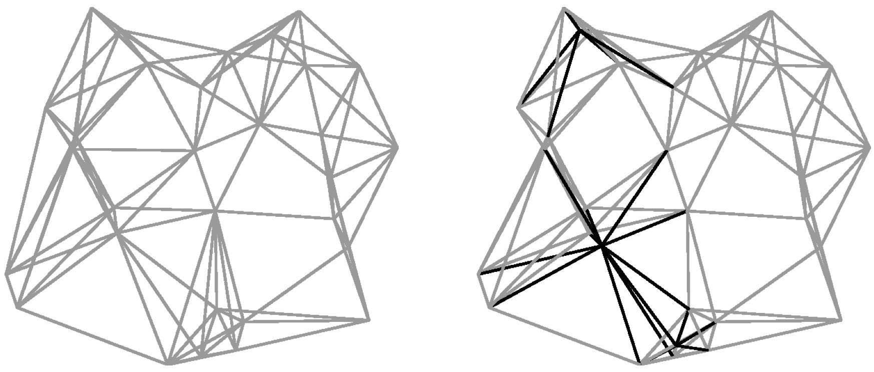


Figure 13: Lower bound on population size. Equation 21 is plotted as a function of LINES with P^* set to 0.1.

indiscriminately perturbs the population. Such a danger does not exist for crossover, since it has its disruptive effect only at those loci where there is disagreement as to the labelling.

In summary, the behavior 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 a single algorithm iteration. Provided the population is large enough to furnish a good selection of labels for all loci, crossover and mutation can 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.



(a) Original Graph

(b) 10% Added Clutter

Figure 14: Synthetic graph with corruption. Clutter is indicated by dark edges.

Table 12: The formulation as it applies to 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 neighborhoods in the data graph
$\mathbf{\Gamma}_j$	A labelling of the j^{th} neighborhood
Θ_j	The dictionary of the j^{th} neighborhood
$\mathbf{\Gamma}$	A labelling of the entire data graph

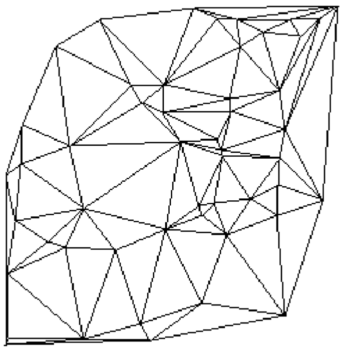
4.6 Graph Matching

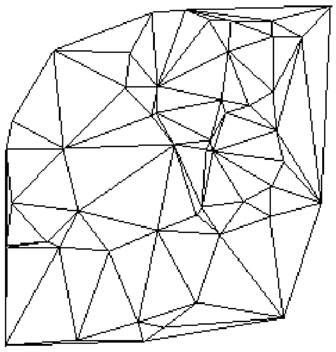
This section describes the extension of the modelling results of Sections 4.3, 4.4, and 4.5 to matching attributed relational graphs. Like line labelling, graph matching is a classical consistent labelling problem. The variant of concern here is attributed relational graph matching (Fu, 1983). A detailed probabilistic formulation of this problem is given in Wilson and Hancock (1997). 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.

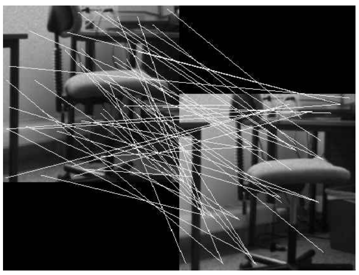
The algorithm was tested on 4 pairs of synthetic graphs each containing 30 nodes. An example of the graphs used in our experiments is shown in Figure 14. The graphs were synthesized by generating random configuration points on a plane and then computing their six nearest neighbor graphs. By randomly perturbing the positions of the original points, we 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.

(a) Left Image

(b) Right Image


(c) Left Feature Graph


(d) Right Feature Graph


(e) Initial Guess (0% correct)

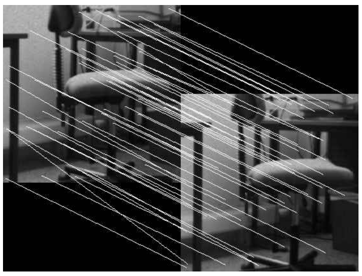

(f) Final Match (98% correct)

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

Table 13: Control variable sets for graph matching. Each set required approximately 700,000 fitness evaluations.

	Set A (Modelling)	Set L1 (De Jong, 1975)	Set L2 (Grefenstette, 1986)
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 14: Algorithm performance. Standard errors are given in parentheses.

Parameter Set	Average Fraction Correct
A (Modelling)	0.93 (0.0041)
L1 (De Jong, 1975)	0.73 (0.0049)
L2 (Grefenstette, 1986)	0.91 (0.0047)

4.6.1 Control Variables

Cross et al. (1997) have shown that the local search step is essential when matching graphs. Combining Wilson and Hancock's (1997) observation that the local search step can recover from significant initialization error 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 (De Jong, 1975; Grefenstette, 1986). 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.

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.

The control variables derived by modelling gave the best performance, followed by those suggested in Grefenstette (1986).

4.6.2 Real Images

Figure 15 is an office scene taken with a low quality web-cam. Regions were extracted from the grey-scale image pair ((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 (Shewchuk, 1996). Examples of the Delaunay graphs are shown in Figure 15(c) and (d). 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 of 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 behavior cannot be considered in isolation because they may interact.

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 that 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.

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 minimized 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 optimization.

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 analyze other algorithms such as the steady-state GA (Syswerda, 1989), GENITOR (Whitley, 1989), or CHC (Eshelman, 1991). 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-analyzed at a later date if higher order interactions are to be studied.

References

Ackley, D. H. (1987). *A Connectionist Machine for Genetic Hillclimbing*. Kluwer Academic, Dordtrecht, The Netherlands.

Andrey, P. and Tarroux, P. (1994). Unsupervised image segmentation using a distributed genetic algorithm. *Pattern Recognition*, 27:659–673.

Anscombe, F. J. (1953). Contribution to discussion of a paper by H. Hotelling. *Journal of the Royal Statistical Society*, B15:229–230.

Aranda-Ordaz, F. J. (1981). On two families of transformations to additivity for binary response data. *Biometrika*, 68:357–363.

Bedau, M. (1995). Three illustrations of artificial life's working hypothesis. *Lecture Notes in Computer Science*, 899:53–68.

Bhandarkar, S. M., Zhang, Y., and Potter, W. D. (1994). An edge extraction technique using genetic algorithm-based optimization. *Pattern Recognition*, 27:1159–1180.

Bramlette, M. F. (1991). Initialization, mutation and selection methods in genetic algorithms. In Belew R. K. and Booker, L. B., editors, *Proceedings of the Fourth International Conference on Genetic Algorithms*, pages 100–107, Morgan Kaufmann, San Mateo, California.

Bremermann, H. J. (1958). The evolution of intelligence. The nervous system as a model of its environment. Technical Report 477(17), Department of Mathematics, University of Washington, Seattle, Washington.

Clowes, M. B. (1971). On seeing things. *Artificial Intelligence*, 2:79–116.

Cochran, W. G. and Cox, G. M. (1957). *Experimental Designs*, Second Edition. John Wiley and Sons, New York, New York.

Collett, D. (1991). *Modelling Binary Data*. Chapman and Hall, London, UK.

Cook, S. A. (1971). The complexity of theorem proving procedures. In *Proceedings of the Third ACM Symposium on the Theory of Computing*, pages 151–158, Association of Computing Machinery, New York, New York.

Cross, A. D. J., Wilson, R. C., and Hancock, E. R. (1997). Inexact graph matching using genetic search. *Pattern Recognition*, 30:953–970.

Cross, A. D. J., Myers, R., and Hancock, E. R. (2000). Convergence of a hill-climbing genetic algorithm. *Pattern Recognition*, 33:1863–1880.

Davis, L. (1989). Adapting operator probabilities in genetic algorithms. In Schaffer, J. D., editor, *Proceedings of the Third International Conference on Genetic Algorithms*, pages 61–69, Morgan Kaufmann, San Mateo, California.

Davis, L. S. (1991). *A Handbook of Genetic Algorithms*. Van Nostrand Reinhold, New York, New York.

DeJong, K. A. (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, Ann Arbor, Michigan.

Eshelman, L. J. (1991). The CHC adaptive search algorithm: How to have safe search when engaging in nontraditional genetic recombination. In Rawlins, G. J. E., editor, *Foundations of Genetic Algorithms*, Volume 1, pages 265–283, Morgan Kaufmann, San Mateo, California.

Faugeras, O. D. and Berthod, M. (1981). Improving consistency and reducing ambiguity in stochastic labeling: An optimisation approach. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 3:412–424.

Fogel, D. (1994). An introduction to simulated evolutionary optimisation. *IEEE Transactions on Neural Networks*, 5:3–14.

Francis, B., Green, M., and Payne, C., editors (1993). *The GLIM System Release 4 Manual*. Oxford University Press, Oxford, UK.

Fraser, A. S. (1957). Simulation of genetic systems by automatic digital computers. *Australian Journal of Biological Science*, 10:484–491.

Fu, K. S. (1983). A step towards unification of syntactic and statistical pattern recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 5:200–205.

Garey, M. R. and Johnson, D. S. (1979). *Computers and Intractability*. Freeman, San Francisco, California.

Geiger, D. and Girosi, F. (1991). Parallel and deterministic algorithms from MRFs: Surface reconstruction. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13:401–412.

Geman, S. and Geman, D. (1984). Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 6:721–741.

Goldberg, D. (1989). *Genetic Algorithms in Search, Optimisation and Machine Learning*. Addison-Wesley, Reading, Massachusetts.

Goldberg, D. E. and Richardson, J. (1987). Genetic algorithms with sharing for multimodal function optimization. In Grefenstette, J., editor, *Proceedings of the Second International Conference on Genetic Algorithms*, pages 41–49, Lawrence Erlbaum, Hillsdale, New Jersey.

Grefenstette, J. J. (1986). Optimisation of control parameters for genetic algorithms. *IEEE Transactions on Systems, Man and Cybernetics*, 16:122–128.

Hancock, E. R. (1994). An optimisation approach to line labelling. In Impedovo, S., editor, *Progress in Image Analysis and Processing*, Volume 3, pages 159–165, World Scientific, Singapore.

Hancock, E. R. and Kittler, J. (1990a). Discrete relaxation. *Pattern Recognition*, 23:711–733.

Hancock, E. R. and Kittler, J. (1990b). Edge labelling using dictionary-based relaxation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12:165–181.

Haralick, R. M. and Shapiro, L. G. (1979). The consistent labelling problem: Part 1. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 1:173–184.

Haralick, R. M. and Shapiro, L. G. (1980). The consistent labelling problem: Part 2. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2:193–203.

Haralick, R. M., Davis, L. S., and Rosenfeld, A. (1978). Reduction operations for constraint satisfaction. *Information Science*, 14:199–219.

Hays, W. L. (1994). *Statistics*, Fifth Edition. Harcourt Brace Harcourt Brace, Fort Worth, Texas.

Holland, J. H. (1975). *Adaptation in Natural and Artificial Systems*. MIT Press, Cambridge, Massachusetts.

Huffman, D. A. (1971). Impossible objects as nonsense sentences. In Meltzer, B. and Michie, D., editors, *Machine Intelligence*, Volume 6, pages 295–323, Edinburgh University Press, Edinburgh, Scotland.

Hummel, R. A. and Zucker, S. W. (1983). On the foundations of relaxation labeling processes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 5:267–287.

Jeavons, P. (1998). On the algebraic structure of combinatorial problems. *Theoretical Computer Science*, 200:185–204.

Jefferson, D. et al. (1991). Evolution as a theme in artificial life: The genesys/tracker system. In Langton, C. G. et al., editors, *Artificial Life II*. Addison-Wesley, Reading, Massachusetts.

Kirkpatrick, S., Gelatt, C. D., and Vecchi, M. P. (1983). Optimisation by simulated annealing. *Science*, 220:671–680.

Koza, J. R. (1992). *Genetic Programming*. MIT Press, Cambridge, Massachusetts.

Lipson, H. and Shpitalni, M. (1996). Optimization-based reconstruction of a 3D object from a single freehand line drawing. *CAD*, 28:651–663.

Lloyd, S. A. (1983). An optimisation approach to relaxation labelling algorithms. *Image and Vision Computing*, 1:85–91.

Mackworth, A. K. (1977). Consistency in networks of relations. *Artificial Intelligence*, 8:99–118.

Mackworth, A. K. and Freuder, E. C. (1985). The complexity of some polynomial network consistency algorithms. *Artificial Intelligence*, 25:65–74.

McCullagh, P. and Nelder, J. A. (1989). *Generalised Linear Models*, Second Edition. Chapman and Hall, London, UK.

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

Mohammed, J. L., Hummel, R. A., and Zucker, S. W. (1983). A gradient projection algorithm for relaxation methods. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 5:330–332.

Mühlenbein, H. (1994). Genetic algorithms. In Aarts, E. and Lenstra, J. K., editors, *Local Search in Combinatorial Optimisation*. Wiley, Bognor Regis, UK.

Mühlenbein, H. and Schlierkamp-Voosen, D. (1995). Analysis of selection, mutation and recombination in genetic algorithms. *Lecture Notes in Computer Science*, 899:142–168.

Nudel, B. (1983). Consistent labelling problems and their algorithms. *Artificial Intelligence*, 21:135–178.

Press, S. A. et al. (1992).*Numerical Recipes in C*, Second Edition. Cambridge University Press, Cambridge, UK.

Prügel-Bennett, A. and Shapiro, J. L. (1994). An analysis of genetic algorithms using statistical physics. *Physical Review Letters*, 72:1305–1309.

Rechenberg, I. (1973). *Evolutionsstrategie - Optimierung Technischer Systeme nach Prinzipien der biologischen Information*. Fromman Verlag, Stuttgart, Germany.

Reed, J., Toombs, R., and Barricelli, N. A. (1967). Simulation of biological evolution and machine learning. *Journal of Theoretical Biology*, 17:319–342.

Rosenfeld, A., Hummel, R. A., and Zucker, S. W. (1976). Scene labelling by relaxation operations. *IEEE Transactions on Systems, Man and Cybernetics*, 6:420–433.

Rudolph, G. (1997). *Convergence Properties of Evolutionary Algorithms*. Kovač, Hamburg, Germany.

Saito, H. and Mori, M. (1995). Application of genetic algorithms to stereo matching of images. *Pattern Recognition Letters*, 16:815–822.

Schaffer, J. D. et al. (1989).A study of control parameters affecting online performance of genetic algorithms for function optimisation. In Schaffer, J. D., editor, *Proceedings of the Third International Conference on Genetic Algorithms*, pages 51–60, Morgan Kaufmann, San Mateo, California.

Schwefel, H. (1981). *Numerical Optimization of Computer Models*. Wiley, Bognor Regis, UK.

Shewchuk, J. R. (1996). Triangle: Engineering a 2D quality mesh generator and Delaunay triangulator. In *Proceedings of the First Workshop on Applied Computational Geometry*, pages 124–133, Association for Computing Machinery, New York, New York.

Syswerda, G. (1989). Uniform crossover in genetic algorithms. In Schaffer, J. D., editor, *Proceedings of the Third International Conference on Genetic Algorithms*, pages 2–9, Morgan Kaufmann, San Mateo, California.

Tsang, P. W. M. (1997). A genetic algorithm for affine invariant recognition of object shapes from broken boundaries. *Pattern Recognition Letters*, 18:631–639.

Vose, M. D. (1995). Modelling simple genetic algorithms. *Evolutionary Computation*, 3:453–472.

Waltz, D. (1975). Understanding line drawings of scenes with shadows. In Winston, P. H., editor, *The Psychology of Computer Vision*, pages 19–91, McGraw-Hill, London, UK.

Whitley, D. (1989). The GENITOR algorithm and selection pressure: Why rank-based allocation of reproductive trials is best. In Schaffer, J. D., editor, *Proceedings of the Third International Conference on Genetic Algorithms*, pages 116–121, Morgan Kaufmann, San Mateo, California.

Whitley, D. et al. (1995). Test driving three 1995 genetic algorithms: New test functions and geometric matching. *Journal of Heuristics*, 1:77–104.

Wilkinson, G. N. and Rogers, C. E. (1973). Symbolic description of factorial models for analysis of variance. *Applied Statistics*, 22:392–399.

Wilson, R. C. and Hancock, E. R. (1997). Structural matching by discrete relaxation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19:634–648.

Yuille, A. L. and Kosowsky, J. J. (1994). Statistical physics algorithms that converge. *Neural Computation*, 6:341–356.