

Analysis and Improvement of Genetic Algorithms
using Concepts from Information Theory

John Milton

THESIS

Submitted to the Faculty of Engineering and IT
University of Technology Sydney (UTS)
in partial fulfillment of the requirements for the
degree of

Doctor of Philosophy
in
Computing Sciences

2009

CERTIFICATE OF AUTHORSHIP/ORIGINALITY

I certify that the work in this thesis has not previously been submitted for a degree nor has it been submitted as part of requirements for a degree except as fully acknowledged within the text.

I also certify that the Thesis has been written by me. Any help that I have received in my research work and the preparation of the thesis itself has been acknowledged. In addition, I certify that all information sources and literature used are indicated in the thesis.

Signature of Candidate

ACKNOWLEDGEMENT

The patience of my wife and son have been greatly appreciated over the many years of work required to complete this Thesis part-time.

I would also like to thank Dr Paul Kennedy for his advice, patience and tireless editorial assistance with both this Thesis and the papers which preceded some of the chapters. Paul's professionalism and guidance have been of enormous benefit to me, both in seeing this thing through and in the quality of the research undertaken.

CONTENTS

1. <i>Introduction</i>	1
1.1 Thesis Outline	8
2. <i>A Review of the Genetic Algorithm Literature</i>	9
2.1 The Genetic Algorithm Literature	12
2.1.1 Schema Theorem and Building Block Hypothesis	13
2.1.2 Little Models	14
2.1.3 Population Size	15
2.1.4 Allele Cardinality	19
2.1.5 Genome Length	22
2.1.6 Mutation	23
2.1.7 Crossover	26
2.1.8 Selection and Ranking	30
2.1.9 Termination Criteria	34
2.1.10 Representations and Mapping	36
2.2 Information Theory Applied to Genetic Algorithms	37
2.3 Genetic Algorithm Benchmarks	40
2.4 Problems in Industry	42
2.4.1 Genetic Algorithms Applied to Industry Problems	43
2.5 Summary of Contributions	44
3. <i>A Model of Genetic Algorithm Behaviour Inspired by Information Theory</i> 47	
3.1 Population Construction	49
3.1.1 Allele Cardinality	49
3.1.2 Solution Density	50
3.1.3 Individual Length	55
3.1.4 Resolution	56
3.1.5 Population Size	60
3.1.6 Summary of Key Ideas	63
3.2 Mutation	63
3.2.1 Allele Replacement Mutation Analysed	64
3.2.2 An Alternative Mutation Mechanism – Allele Flipping.	69
3.2.3 Discussion	71
3.2.4 Why Use Mutation At All?	71
3.2.5 Targeting Mutation to Add Information.	72
3.2.6 Summary of Key Ideas	74

3.3	A Model of Solution Density in a Genetic Algorithm Subject to Selection	75
3.3.1	The Model	76
3.3.2	Static Threshold	83
3.3.3	A Model with Parents	85
3.3.4	Dynamic Threshold	87
3.3.5	Summary of Key Ideas	90
3.4	Some Observations Regarding Crossover	92
3.4.1	Crossover and the Distribution of Ideal Alleles	93
3.4.2	Difficulty Respecting Non-Repeating Sequences	94
3.4.3	Graph Properties	95
3.4.4	Sampling Rate of a Population Subject to Crossover	98
3.4.5	Approaches to Crossover	99
3.4.6	Crossover Section Length	99
3.4.7	Calculating Sufficient Crossover	100
3.4.8	Comparison of Crossover to the Random Selection of Alleles	108
3.4.9	Summary of Key Ideas	109
3.5	Finding the Thresholds with Imperfect Knowledge	110
3.5.1	Dynamic Range Explained	110
3.5.2	Epistasis Described	112
3.5.3	Eliminating Epistasis	113
3.5.4	Managing Dynamic Range	114
3.5.5	Entropy Profile	117
3.5.6	Integer Partition	123
3.5.7	Most Probable Individuals	125
3.5.8	Summary of Key Ideas	126
3.6	Termination	127
3.6.1	Presence of the Optimal Solution with Arbitrary Confidence	128
3.6.2	Feasible Space for Exhaustive Search	130
3.6.3	Solution Density at ‘Feasible’ Search Point	131
3.6.4	Most Probable Search Space	135
3.6.5	Discussion	139
3.6.6	Summary of Key Ideas	140
3.7	Fundamental Contributions	141
4.	<i>Simulations to Examine the Fidelity of the Model</i>	145
4.1	Simulation One	145
4.1.1	Simulation One Construction and Parameter Settings	146
4.1.2	Simulation One Results	148
4.1.3	Discussion of Simulation One	148
4.2	Simulation Two	151
4.2.1	Simulation Two Construction and Parameter Settings	153
4.2.2	Simulation Two Results	155
4.2.3	Discussion of Simulation Two	156
4.3	Conclusion	159

5. <i>Prototype Genetic Algorithms Applied to Benchmark Problems</i>	160
5.1 Genetic Algorithm Prototype	161
5.2 Bit Traps	162
5.3 <i>NK</i> Landscapes	167
5.4 Benchmark Functions	171
5.5 Mapping and Representation	173
5.6 Conclusion	174
6. <i>Industrial Problem – Scheduling</i>	176
6.1 A Final Test	176
6.2 The Scheduling Problem	177
6.2.1 Job Shop Schedules	179
6.3 The Application of Genetic Operators to Schedules	183
6.3.1 Problem Knowledge	185
6.3.2 A Common Representation	186
6.3.3 An Alternative Representation	187
6.4 Experimental Trials	193
6.4.1 Experiment Design	194
6.4.2 Selected Benchmarks from Taillard	199
6.4.3 What Experimental Observations Characterise a ‘Good’ versus a ‘Poor’ Result?	202
6.4.4 Job Shop Schedule Results	203
6.4.5 Result of Significance Testing	212
6.5 Conclusion	213
7. <i>Conclusion</i>	216
7.1 Population Construction	217
7.2 Mutation	217
7.3 Selection	218
7.4 Crossover	218
7.5 Ranking	219
7.6 Termination	219
7.7 Testing the Model	219
7.8 Future Work	222

LIST OF FIGURES

3.1	Coding Efficiency for Binary ($A = 2$) allele cardinality as the Number of Possible Symbols ($ \Phi $) increases.	51
3.2	An unranked population of ten individuals (rows) having ten loci (columns) with each ideal allele indicated in grey from the solution (A,C,C,D,D,C,D,B,A,B).	52
3.3	Population Growth with Allele Cardinality (A) and a variety of Confidence levels B that at least one ideal allele exists in each loci ($L = 60$).	62
3.4	Population Growth with Genome Length (L), Allele Cardinality ($A = 64$) and a variety of Confidence levels B that at least one ideal allele exists in each loci.	62
3.5	A population of ten individuals (rows) having ten loci (columns) ranked by ideal allele from the solution (A,C,C,D,D,C,D,B,A,B) indicated in grey. The threshold separates individuals with more / less ideal alleles than the population average (as generated by the memoryless information source).	73
3.6	The binomial distribution $\mathbf{p}_0^L(0, \lambda)$ describes the number of ‘ideal’ alleles per individual before selection. 78	
3.7	The probability distribution $\mathbf{p}_3^L(0, \lambda)$ of the population after selection has removed individuals with less than 3 ideal alleles (Selection threshold $k = 2$). 78	
3.8	The truncated distribution of Fig. 3.7 with alleles redistributed by <i>sufficient crossover</i> to return it to a binomial distribution $\mathbf{p}_0^L(1, \lambda)$	78
3.9	This figure illustrates the movement of the probability density function from $\mathbf{p}(0, \lambda)$ (columns) to the region of higher solution density at $\mathbf{p}(g, \lambda)$ (lines) due to repeated selection and crossover.	80
3.10	The expected solution density predicted by Equation (3.35) for a variety of selection thresholds k . ($N = 31, L = 13, A = 6$).	85
3.11	The expected solution density predicted by Equation (3.45) for the optimal dynamic selection threshold k_g and for $k_g + 1, k_g - 1, k_g - 2, k_g - 3$. ($N = 31, L = 13, A = 6$).	89

3.12	While one of the trials shown has sufficient selection pressure to improve its solution density, the other four trials have selection thresholds below the dynamic threshold and hence failed to improve from generation to generation.	90
3.13	This snapshot of allele frequency vs loci at $G = 30$ shows how, with low selection pressure, no allele dominates in any loci. . . .	91
3.14	This snapshot of allele frequency vs loci at $G = 30$ shows how, with selection pressure guided by a dynamic threshold, certain alleles come to dominate loci. If ranking is not deceived, these will be ideal alleles.	91
3.15	Graph showing Hamming Distance between Parents and Children Subject to Crossover.	96
3.16	Distance between the distribution ψ_c and the distribution $\mathbf{p}(g+1, \lambda)$ for alleles exchanged $Y = 1$ to 7 per crossover operation. ($N = 31, L = 13, A = 6, k = 2$).	105
3.17	Distance between the distribution ψ_c and the distribution $\mathbf{p}(g+1, \lambda)$ for population sizes $N = 31, 62, 93$. ($L = 13, A = 6, Y = L/2, k = 2$).	105
3.18	The distance between the distribution ψ_c and the distribution $\mathbf{p}(g+1, \lambda)$ for a variety of genome lengths $L = 13, 26, 52$. For each value of L , seven crossover section lengths are shown distributed between $Y = 1$ and $Y = L/2$. In each case the line for $Y = L/2$ has the fastest change in distance. ($N = 31, A = 6, k = 2$).	106
3.19	Distance between the distribution ψ_c and the distribution $\mathbf{p}(g+1, \lambda)$ for allele cardinality $A = 4, 6, 16$. For each value of L , seven crossover section lengths are shown from $Y = 1$ to $Y = 7$. In each case the line for $Y = L/2$ has the fastest change in distance. ($N = 31, L = 13, k = 2$).	106
3.20	Distance between the distribution ψ_c and the distribution $\mathbf{p}(g+1, \lambda)$ for selection thresholds $k = 2, 4, 6$. For each value of L , seven crossover section lengths are shown from $Y = 1$ to $Y = 7$. In each case the line for $Y = L/2$ has the fastest change in distance. ($N = 31, L = 13, A = 6$).	107
3.21	The multiplier Γ which provides the number of crossover operations $C = \Gamma N$ required to return a population of individuals with length L , solution density ρ_g , selection threshold k_g and uniform crossover probability equal to 0.5 , to a binomial distribution. . . .	108
3.22	The ranking of a population having low dynamic range between some alleles scored using the raw objective function result. Perfect ranking would result in a 45° line. Those individuals falling in the quadrant marked 'BR' are deleted when they should be retained, leading to an information loss of 19 bits. The horizontal and vertical dotted lines represent the selection threshold at the generation shown.	116

-
- 3.23 The ranking of the Figure 3.22 population scored using the A vs L matrix $\mathbf{I}_{a,l}$. Note how a 45° line is more closely approximated indicating close to ideal ranking. The horizontal and vertical dotted lines represent the selection threshold at the generation shown. 116
- 3.24 The ranking of a population having high dynamic range between some alleles scored using the raw objective function result. Note the increased misranking due to the higher dynamic range (100 times greater than in Figure 3.22). The structured ‘lines’ occur as some individuals contain none, one, the other, or both high contribution alleles. The horizontal and vertical dotted lines represent the selection threshold at the generation shown. 118
- 3.25 The ranking of the Figure 3.24 population scored using the A vs L matrix $\mathbf{I}_{a,l}$. The horizontal and vertical dotted lines represent the selection threshold at the generation shown. 118
- 3.26 The entropy profile of a population of individuals each of 10 loci long formed from a 64 symbol alphabet. The upper line is the list randomly arranged. One ranking uses the raw objective function score (dashed), the other (bottom line) ranks by the use of the cumulative scoring method. 120
- 3.27 The entropy profile of a population of individuals each of 60 bits long formed from a 64 symbol alphabet. The entropy profile below the threshold (n) appear as lines with a positive gradient, while the lines with negative gradient are the entropy profile above the threshold (n). The upper solid lines are from the list randomly arranged. The dashed lines use the raw objective function score to rank while the lower solid lines rank by the use of the cumulative scoring method described in Section 3.5.4. The dotted lines are the maximum and minimum possible entropy profiles for a list of this type. 122
- 3.28 The average results of 100 trials where crossover has been performed between randomly selected pairs of individuals 310 times. ($N = 31, L = 13, A = 6$). 135
- 3.29 The upper two lines show the maximum entropy per loci for a population with allele cardinalities $A = 16$ and $A = 64$ as solution density of the population rises. The bottom line is the minimum entropy for each of these cases. The actual entropy of a population will lie between these maximum and minimum lines. 138
- 4.1 The expected solution density predicted by Equation (3.35) for a variety of selection thresholds k . ($N = 31, L = 13, A = 6$). 148
- 4.2 The average results of 100 trials where crossover has been performed between randomly selected pairs of individuals 310 times each generation, $C = 310$. ($N = 31, L = 13, A = 6$). 149

4.3	The average results of 100 trials where crossover has been performed between randomly selected pairs of individuals with only $C = 10$ crossovers each generation. ($N = 31, L = 13, A = 6$).	150
4.4	N_g is the size of the full population at generation g , while N_{g,k_0} is the size of the population up to the static threshold k_0 at generation g and N_{g,k_g} is the size of the population up to the dynamic threshold k_g at generation g	152
4.5	The bottom ranked N_{g,k_0} individuals are replaced with randomly generated individuals. Replacing the next $N_{g,k_g} - N_{g,k_0}$ individuals with randomly selected individuals from above N_{g,k_g}	155
4.6	The expected solution density predicted by Equation (3.45) for the optimal dynamic selection threshold k_g and for $k_g + 1, k_g - 1, k_g - 2, k_g - 3$ and the average of these three. ($N = 31, L = 13, A = 6$).	156
4.7	The average results of 100 trials where crossover has been performed between randomly selected pairs of individuals 310 times. ($N = 31, L = 13, A = 6$).	157
4.8	The average results of 100 trials where crossover has been performed between randomly selected pairs of individuals with only $C = 10$ crossovers per generation. ($N = 31, L = 13, A = 6$).	158
5.1	The HMXT algorithm with allele cardinality $A = 16$ attempting to solve a problem of 10 concatenated 6-bit traps (problem length $L_p = 60$ bits, $N = 114$). The structural similarity is a normalised measure of the Hamming distance to the optimal solution. The average of five trials is shown.	165
5.2	The HMXT algorithm with allele cardinality $A = 64$ solving a problem of 10 concatenated 6-bit traps (problem length $L_p = 60$ bits, $N = 439$). The structural similarity is a normalised measure of the Hamming distance to the optimal solution. The average of five trials is shown.	166
5.3	The HMXT algorithm with allele cardinality $A = 64$ solving a problem of 10 concatenated 6-bit traps (problem length $L_p = 60$ bits, $N = 439$) the same as Figure 5.2 but ranking using only the raw score. The structural similarity is a normalised measure of the Hamming distance to the optimal solution. The average of five trials is shown.	166
5.4	A problem of 60 concatenated 6-bit traps. The genetic algorithm has a length of $L_p = 360$ bits and a population size of $N = 553$. 5530 evaluations are completed in $G = 10$ generations. The genetic algorithm suffered information loss which prevented it from finding the optimum. Note the low structural similarity achieved.	168

5.5	A problem of 60 concatenated 6-bit traps, the same problem and allele cardinality as Figure 5.4, but with a doubled population size. The genetic algorithm has a length of $L_p = 360$ bits and a population size of $N = 1106$. 11060 evaluations are completed in $G = 10$ generations. Note the improved results due to doubled population size which has countered the information loss apparent in Figure 5.4.	168
5.6	An NK landscape problem with $K = 6$ and $L_p = 24$. When only 4 bits are grouped by the genetic algorithm into a 16 symbol allele cardinality, a $K = 6$ landscape is difficult to solve. The genetic algorithm has a length of $L_p = 24$ bits and a population size of $N = 100$ individuals. 1000 evaluations are completed in $G = 10$ generations. The structural similarity is a normalised measure of the Hamming distance to the optimal solution.	170
5.7	An NK landscape problem with $K = 6$ and $L_p = 24$. When 6 bits are grouped by the genetic algorithm into a 64 symbol allele cardinality, a $K = 6$ landscape is much easier to solve. The genetic algorithm has a length of $L_p = 24$ bits and a population size of $N = 381$. 3810 evaluations are completed in $G = 10$ generations. The structural similarity is a normalised measure of the Hamming distance to the optimal solution.	170
6.1	A schedule with critical blocks shown underlined.	181
6.2	A schedule with sequence point bottlenecks circled.	182
6.3	A schedule with timing point bottlenecks circled.	182
6.4	Nearchou C3 crossover operator which preserves allele frequency in a child individual (Parent 1 on top, Parent 2 on bottom, child center).	184
6.5	An individual deciding the sequence of operations in a critical block. Each pair of letters shown (CD), (DB) and (AD) are treated as single alleles by the HMXT algorithm.	189
6.6	A schedule constructed by applying the individual of Figure 6.5 to the ambiguous schedule of Table 6.3.	189
6.7	Another individual resolving the second sequence point bottleneck (loci 2) differently to the individual shown in Figure 6.5. . .	190
6.8	The schedule resulting from applying the individual of Figure 6.7 to the ambiguous schedule of Table 6.3.	190
6.9	An individual encoding a shift of -2 changes the upper schedule of Figure 6.10 into the lower schedule by shifting the 7th bottleneck operator two positions left.	192
6.10	Applying the individual shown in Figure 6.9 to the upper schedule, results in the lower schedule where job 'A' has shifted two places left on machine 3. The lower schedule has an improved makespan and a changed critical path. This is how stage two of the HMXT algorithm continually refines schedules.	192

6.11	An illustration of the sparse matrix Z showing how operations are placed into a three dimensional array, with competing operations placed in the next available position in the dimension marked ‘a’. This is used to construct an individual such as in Figure 6.7. . . .	195
6.12	All 5 trials for a HMXT algorithm optimising a Taillard 02 benchmark as an illustration of the typical variation between trials for a given problem.	201
6.13	Using a hypothesised control as a comparison, the anticipated mean profiles of ‘excellent’, ‘good’ and ‘poor’ algorithms are illustrated.	204
6.14	Using a hypothesised control as a comparison, the anticipated difference profiles of ‘excellent’, ‘good’ and ‘poor’ algorithms are illustrated.	204
6.15	Each line is the average of five trials of Taillard 01 to 10, acted on by the <i>control</i> with Linear Ranking (A), then the <i>control</i> with Tournament Selection (B) and finally the HMXT algorithm (C).	206
6.16	The percentage difference between the results shown in Figures 6.15 and 6.15 – A (HMXT minus Linear Ranking <i>control</i>). Note how the HMXT genetic algorithm outperforms the <i>control</i> (Taillard 01 to 10).	208
6.17	The percentage difference between the results shown in Figures 6.15 and 6.15 – B (HMXT minus Tournament Selection <i>control</i>). Note how the HMXT genetic algorithm outperforms the <i>control</i> , especially early in the process (Taillard 01 to 10).	208
6.18	All 5 trials for Taillard 38 benchmark as an illustration of the typical variation between trials for a given problem. The genetic algorithm uses parameters from Chapter 3 of this Thesis.	209
6.19	Each line is the average of five trials of Taillard 38, 62 (bottom) and 71 (upper), acted on by the <i>control</i> with Linear Ranking (A), then the <i>control</i> with Tournament Selection (B) and finally the HMXT algorithm (C).	210
6.20	The percentage difference between the results shown in Figures 6.19 – C and 6.19 – A (HMXT minus Linear Ranking <i>control</i>). Note how the HMXT genetic algorithm outperforms the <i>control</i> . (Taillard 38, 62 and 71.)	211
6.21	The percentage difference between the results shown in Figures 6.19 – C and 6.19 – B (HMXT minus Tournament Selection <i>control</i>). Note how the HMXT genetic algorithm outperforms the <i>control</i> . (Taillard 38, 62 and 71.)	211

LIST OF TABLES

3.1	The five parameters in Algorithm 2 which influence how many crossover operations are <i>sufficient</i> are; population size (N), genome length (L), allele cardinality (A), selection threshold (k) and crossover section length (Y). Because of the interesting behaviour of varying crossover section lengths, up to seven of these lengths from one loci up to $L/2$ loci are illustrated in Figures (3.16) to (3.20).	103
3.2	Fitness of the four example individuals. The score contributed by each loci containing a 1 adds linearly to an individual's score.	111
3.3	Search space size and 'equivalent generations' to complete an exhaustive search for increasing solution density and $A=6$, $L = 13$ and $N = 31$. $S = A^L$ is the total solution space while S_{min} is calculated using Equation (3.64) and S_{max} using Equation (3.71).	136
3.4	Most probable search space size and 'equivalent generations' to complete an exhaustive search as solution density improves for $A=6$, $L = 13$ and $N = 31$. S_{mp} is calculated using Equation (3.75). Generations is S_{mp}/N	139
5.1	Parameters, ranking and threshold setting for five bit trap benchmarks.	164
5.2	The average results over five independent trials of the HMXT algorithm applied to each function while utilising the $\mathbf{I}_{a,l}$ matrix to determine solution density and selection thresholds with ranking by the raw score. Each point in the function space is represented by 6 bits. The values shown in brackets are those from Li et al. (2006).	173
6.1	An example sequence of machines per job (\mathbf{M}). For clarity of presentation the matrix \mathbf{M} is transposed here and machines are indicated by numbers.	180
6.2	An example matrix (\mathbf{D}) which specifies the time (in units) required by each operation of each job in the example problem. For clarity of presentation the matrix \mathbf{D} is transposed here and machines are indicated by numbers.	180

6.3	An ambiguous schedule showing the sequence of jobs per machine as transformed from Table 6.1. Note that decisions need to be made as some jobs compete for the same machine at the same sequence point.	180
6.4	Summary of the parameters used by the representation described in Section 6.3.3 to operate on various Taillard benchmarks. These parameters vary from stage to stage and trial to trial. Hence they are only representative of the magnitude they take.	202
6.5	Summary of the Best Makespan(above) with average and standard deviation (below) for each of the <i>controls</i> and the HMXT algorithm. The last column shows the known optimum makespan for the indicated problem.	212
6.6	Summary of significance test (t-test, <i>p</i> value) results for each of the <i>control</i> algorithms. Bold entries indicate that the difference between the HXMT mean and <i>control</i> mean is statistically significant(ie; $p < 0.05$).	213

ABSTRACT

Evolutionary algorithms are based on the principles of biological evolution (Bremermann et al., 1966; Fraser, 1957; Box, 1957). Genetic algorithms are a class of evolutionary algorithm applicable to optimisation of a wide range of problems because they do not assume that the problem to be optimised is differentiable or convex. Potential solutions to a problem are encoded by allele sequences (genes) on an artificial genome in a manner analogous to biological DNA. Populations of these artificial genomes are then tested and bred together, combining artificial genetic material by the operation of crossover and mutation of genes, so that encoded solutions which more completely optimise the problem flourish and weaker solutions die out.

Genetic algorithms are applied to a very broad range of problems in a variety of industries including financial modeling, manufacturing, data mining, engineering, design and science. Some examples are:

- Traveling Salesman Problems such as vehicle routing,
- Scheduling Problems such as Multiprocessor scheduling, and
- Packing problems such as Shipping Container Operations.

However, relative to the total volume of papers on genetic algorithms, few have focused on the theoretical foundations and identification of techniques to build effective genetic algorithms. Recent research has tended to focus on industry applications, rather than design techniques or parameter setting for genetic

algorithms. There are of course exceptions to these observations. Nevertheless, the exceptions generally focus on a particular parameter or operator in relative isolation and do not attempt to find a foundation, approach or model which underpins them all.

The objective of this Thesis is to establish theoretically sound methods for estimating appropriate parameter settings and structurally appropriate operators for genetic algorithms. The Thesis observes a link between some fundamental ideas in information theory and the relative frequency of alleles in a population. This observation leads to a systematic approach to determining optimum values for genetic algorithm parameters and the use of generational operators such as mutation, selection, crossover and termination criteria. The practical significance of the Thesis is that the outcomes form theoretically justified guidelines for researchers and practitioners.

The Thesis establishes a model for the analysis of genetic algorithm behaviour by applying fundamental concepts from information theory. The use of information theory grounds the model and contributions to a well established mathematical framework making them reliable and reproducible. The model and techniques contribute to the field of genetic algorithms by providing a clear and practical basis for algorithm design and tuning.

Two ideas are central to the approach taken. Firstly, that evolutionary processes encode information into a population by altering the relative frequency of alleles. Secondly, that the key difference between a genetic algorithm and other algorithms is the generational operators, selection and crossover. Hence the model maximises a population's information as represented by the relative frequency of solution alleles in the population, encourages the accumulation of these alleles and maximises the number of generations able to be processed.

Information theory is applied to characterise the information sources used for mutation as well as to define selection thresholds in ranked populations. The importance of crossover in distributing alleles throughout a population and in promoting the accumulation of information in populations is analysed, while the Shannon–McMillan theorem is applied to identify practical termination criteria.

The concept of *ideal alleles* as being those symbols in the appropriate loci, which form an optimal solution and the associated *solution density* of the population is central to this analysis. The term *solution density* is introduced to refer to the relative frequency of ideal alleles in the population at a particular generation. Solution density so defined represents a measure of a population’s fitness.

By analysing the key genetic operators in terms of their effect on solution density, this Thesis identifies ten contributions.

- A model for the analysis of genetic algorithm behaviour inspired by information theory.
- A *static selection* threshold in ranked populations.
- A *dynamic selection* threshold in ranked populations.
- A maximum limit on the number of loci participating in epistasis is identified whereby more epistatic loci degrade directed search.
- A practical limit to the amount of useful crossover is identified as *sufficient*.
- An optimal crossover section length is found.
- A cumulative scoring method for identifying solution density.
- An *entropy profile* of ranked lists is described.
- A practical termination criteria of *most probable individuals* based on the Shannon–McMillan theorem is provided.

- An alternative genome representation which incorporates job–shop schedule problem knowledge in the genome rather than the algorithm’s generational operators is developed.

Each of these contributions is validated by simulations, benchmark problems and application to a real–world problem.