On the Impact of Modelling, Robustness, and Diversity to the Performance of a Multi-Objective Evolutionary Algorithm for Digital VLSI System Design

Robert Thomson* and **Tughrul Arslan*** † [1]
* School of Engineering and Electronics, Edinburgh University, UK.
Robert.Thomson@ee.ed.ac.uk, Tughrul.Arslan@ee.ed.ac.uk
† Jet Propulsion Laboratory, Avionics Section, Pasadena, CA 91109.
Tughrul.Arslan@jpl.nasa.gov

Abstract- This paper describes the operation of an Evolutionary Algorithm (EA) for the creation of linear digital VLSI circuit designs. The EA can produce hardware designs from a behavioural description of a problem. The designs are based upon a library of high-level components.

The EA performs a multi-objective search, using models of the longest-path delay and the silicon area of a design. These models are based upon the properties of real-world components, implementable in a 0.18 micron technology. The accuracy of these models is investigated.

Two important aspects of multi-objective evolution are the population diversity, and the variability of the results. Both of these areas are examined. The population diversity is assessed in terms of conflict between the objectives, and the robustness of the EA is experimentally investigated.

1 Introduction

Evolutionary Algorithms [1, 2] are a class of stochastic algorithms that can be used for the discovery of near-optimal solutions to complex multi-modal problems. EAs make use of a population of solutions, enabling the simultaneous discovery of multiple solutions to a problem.

Multi-Objective Evolutionary Algorithms (MOEAs) [3] attempt to optimise more than one objective simultaneously. The results from such a system are a set of solutions, where each solution makes a different compromise between satisfying the different objectives.

1.1 Evolutionary Hardware Design

Evolutionary Algorithms have been applied to a large number of problems related to electronic design. In particular, they have been used for the design of digital hardware, both at gate level [4], and also using high-level components [5]. EAs can also be used to improve circuit properties such as area, longest-path delay or power [6]. EAs can be used with both adaptive and non-adaptive hardware.

There are several advantages to evolutionary approaches to circuit design. Evolutionary techniques can create useful hardware designs with very little human intervention. Many electronics problems have multiple objectives; for example silicon area, power dissipation, longest path delay or fault tolerance. When there are multiple objectives, multi-objective EAs are a powerful technique for design space exploration.

1.2 Primitive Operator Circuit Designs

Many useful digital circuits are based upon constant factor multiplications. When multiplying by a constant, it is often inefficient to use a full multiplier. In many cases, a dedicated constant-factor multiplier can be used instead. Constant-factor multipliers are typically more efficient than a full multiplier, with respect to area, power and longest-path delay.

Constant-factor multipliers can be constructed from Primitive Operators. These are components such as adders, subtracters, negators, and bit-shifts. When compared with a full multiplier, Primitive Operators use much less power and silicon area, and operate much faster. The number of Primitive Operator components required for the implementation of a particular constant multiplication depends upon the multiplication factor. While constant multiplier designs can be derived from the binary or Canonical Signed Digit (CSD) [7] representation of the constant, better results can be achieved using the algorithms proposed by Dempster and Macleod [8].

In many cases, a single value is multiplied by several different constants. This enables common sub-expression sharing between the different multiplications. Rather than using a set of distinct multipliers, a combined multiplication block can be used.

The synthesis of efficient multiplication blocks is far more complex than the problem of synthesising constant multipliers. In fact, the problem of creating a multiplication block using the minimum number of components is NP-complete [9, 10]. Therefore, optimal solutions can not normally be found, and the only reasonable approach is to search for near-optimal solutions instead. This can be done with search algorithms such as the RAG-n algorithm [11]. Constant-factor multiplication blocks are used in transposed form Finite Impulse Response (FIR) filters, and for many other signal processing tasks.

Further improvements can be achieved by combining several multiplication blocks into a single piece of hardware. The combined hardware has multiple inputs and multiple outputs, and performs a linear transform. An example of an algorithm for the generation of efficient solutions to this problem is given in [12].

[1] Dr. Tughrul Arslan is currently working as a research scientist at the Jet Propulsion Laboratory, Avionics Section, Pasadena, CA 91109.

2 Multi-Objective Evolution

There are many problems for which a single-objective EA is inadequate. Such problems have multiple conflicting objectives. Multi-Objective Evolutionary Algorithms (MOEAs) extend Evolutionary Algorithms so that, as well as attempting to meet particular objectives, various compromises between the objectives are considered. The use of a population of solutions enables the simultaneous discovery of multiple compromise solutions.

If there is no conflict between a set of objectives, the set of objectives can be combined into a single objective, for example by summing the objective values. If the objectives do conflict, such a strategy is less likely to be useful; the goal is to find a set of compromise solutions, but a single-objective approach will typically rank one of the compromise solutions highest.

The best solutions that an MOEA discovers are known as the non-dominated set, and the individual best solutions are known as non-dominated solutions. A solution is dominated if there is another solution that is no worse with respect to all objectives, and better with respect to at least one objective. Hence, a non-dominated solution is one that is not dominated by any other solution, and can therefore be considered a 'best' solution.

If an MOEA has a large number of objectives, it is unlikely that one solution will dominate another solution. This can reduce the amount of competition between the solutions, eventually leading to poorer quality results. If there are few objectives, competition between solutions is more likely. For this reason, an MOEA should avoid having non-conflicting objectives, as far as possible.

Multi-objective EAs need to find a diverse set of solutions, covering many different compromises between the objectives. This can be achieved through the use of ranking algorithms such as Goldberg's Non-Dominated Sorting technique [1]. The advantage of the Non-Dominated Sorting algorithm is that it ranks all of the non-dominated solutions equally, reducing the tendency for all solutions to converge on a particular part of the solution space. It is usually combined with a method for explicitly encouraging diversity, such as *niching*.

In this paper, we describe an EA system for the creation of linear digital VLSI circuits. It has three objectives. The objectives are the functional correctness of a circuit design, the longest-path delay, and the silicon area required.

3 The Evolution of Digital Hardware

The system described in this paper evolves digital VLSI circuit designs that perform linear transforms. The desired transform is described by a user-specified matrix, and the EA produces circuit netlists for appropriate circuit designs. Each circuit netlist specifies how high-level components such as adders and subtracters can be connected together to form a complete circuit design. The netlists are written in the Verilog hardware description language.

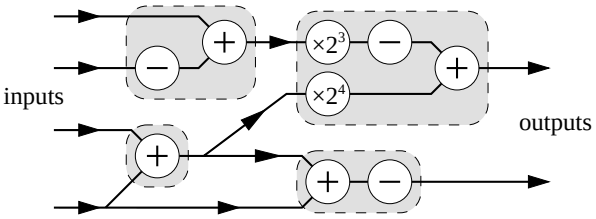


Figure 1: A chromosome represented by a graph. The grey areas represent the graph nodes.

3.1 The Chromosome

Each chromosome represents a circuit design. The chromosome is encoded as a graph, where the nodes represent components and the edges represent connections. This means that there is a very close correspondence between the genotype and the phenotype, and that it a very straight-forward task to convert from the genotype to the phenotype.

Each node in the graph can calculate a value of the form $\pm 2^a x \pm 2^b y$, where x and y are the node inputs, and a and b are integer constants. The nodes can be implemented using an adder, a subtracter, or an adder followed by a negator. Multiplications by constant powers of 2 are achieved by bit-shifting values.

Figure 1 shows an example of how a graph is used to represent a chromosome. The graph in figure 1 has four inputs, four nodes, and two outputs. The nodes are represented by the grey areas. The multiplications in figure 1 can be implemented using bit-shifts. Implementations of the top two nodes could make use of subtracters.

3.2 Hardware Modelling

The silicon area and longest-path delay are estimated using data derived from a library of real components. The library includes adders, subtracters and negators, for all component widths between 1 and 64 bits.

As the hardware models are only used for the comparison of designs, absolute accuracy is not necessary. All that is required is that the models should be able to establish ordinal relationships between designs. Thus the overall system is insensitive to many types of inaccuracy in the hardware models. This is very useful, as it reduces the need for complex, computationally expensive hardware models.

The area estimate is only based upon the cell area. This is the major part of the total area, and it can be accurately estimated. The area used by interconnects is still a significant part of the total, which is ignored. Note that the interconnect area is in some respects a function of the cell area, as larger designs require longer wires. The area estimate is calculated by summing the areas of all of the components used in the design. The component area values are precalculated.

The longest-path delay estimate is found by calculating the delay at every point in a design. These delays are found by summing the longest-path delays for all of the components on a particular path. The interconnect delays are ignored.

There are two significant sources of inaccuracy in the

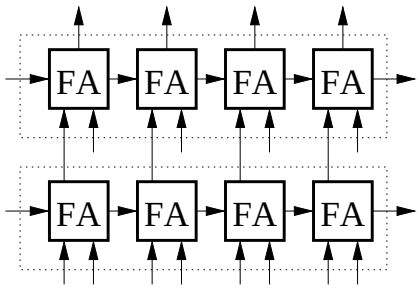


Figure 2: Two 4-bit ripple adders in series. The critical path is 5 full-adders long. The critical path for a single 4-bit ripple adder is 4 full-adders long.

Operator	Probability
insertion of a new component	0.2
modification of a connection	0.2
modification of a shift and/or negation	0.2
component removal	0.2
component reorganisation (associativity)	0.2

Table 1: The genetic operators.

delay estimation system. One problem is that in reality, different bits in a value can be delayed by different amounts, but the delay model assumes all of the bits have the same, worst-case delay. For example, the delay through two ripple adders is less than twice the longest-path delay of a single ripple adder. This is illustrated in figure 2. A second problem is that the interconnect delays are ignored. The interconnect delays can be significant, as wires that have a large area can take a relatively long time to propagate data.

The delay model could be improved by calculating the delays for each bit rather than for each value. Further improvements could be achieved by simulating the increase in area, and hence the increase in delay, as the fanout[2] of a connection is increased.

The desired response of a design is specified by the user. It is a transformation matrix. The functionality of a design can be assessed through calculation of the impulse response of the design, and by comparing the impulse response with the desired transformation matrix. The current system calculates the sum of squares difference between the actual and desired responses, and uses that value as a measure of how functional a particular design is. This objective has been termed the 'functional error'.

3.3 Genetic Operators

The genetic operators are entirely mutational. There is no crossover operator. There are five genetic operators which are listed in table 1. When a child chromosome is created, one of these operators is applied to a copy of the parent chromosome.

[2]The *fanout* of a wire is the number of components that are driven by that wire.

3.4 The Evolutionary Algorithm

The selection scheme should preferentially select individuals on the basis of the three objective values. The selection scheme should also encourage individuals to meet the functionality constraint.

Size-2 tournament selection has been used. The tournament selection chooses individuals based upon the rank assigned by the non-dominated sorting [1] algorithm. If the two individuals in a tournament have the same rank according to the non-dominated sorting scheme, then the individual with the lowest functional error wins the tournament. This ensures that the most functional designs are rewarded. This selection scheme is able to reward progress with respect to all three objectives, while particularly encouraging the development of functional designs.

The EA is a $(\mu + \lambda)$ system. An initial population of 100 individuals is expanded through the creation of 100 mutant child chromosomes. The resulting population of 200 is then reduced to 100 individuals. Selection occurs in two different circumstances — the selection operator is used when choosing parents, and also for choosing survivors when reducing the population size.

To prevent the elimination of the best solutions, elitism is also used when reducing the population size. The elitism operator preserves up to ten of the most functional non-dominated solutions, so that they are not eliminated when the population size is reduced.

There is usually some level of functional error that can be considered acceptable. We have termed such functionally acceptable designs 'correct'. If a design is correct, the EA should only concentrate on reducing the area and longest-path delay. All correct designs are considered to have the same functional error score, so there is no reward for designs that function better than required. The score at which designs are considered correct is a user-specified parameter.

4 Experiments

The system was tested on the 4-point Discrete Cosine Transform. This is a linear transform, which has four inputs and four outputs. It can be specified with the following matrix:

$$\begin{bmatrix} 0.5 & 0.5 & 0.5 & 0.5 \\ 0.6533 & 0.2706 & -0.2706 & -0.6533 \\ 0.5 & -0.5 & -0.5 & 0.5 \\ 0.2706 & -0.6533 & 0.6533 & -0.2706 \end{bmatrix}$$

This problem was used because it is a relatively small problem, which can be solved using few components, so large numbers of tests are possible.

The functional error of the 'correct' solutions was set at 0.004 or lower. In other words, the sum of the squares of the coefficient errors should be 1000 times smaller than the sum of the squared coefficients. Only designs that meet this constraint are shown in the following results.

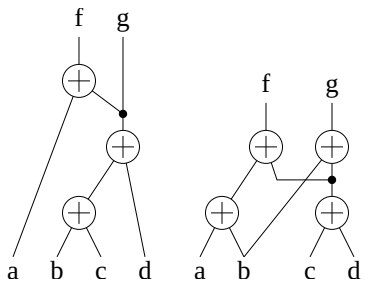


Figure 3: An example of conflicting objectives.

4.1 Reliability and Robustness

As EAs are a stochastic technique, there is no guarantee that they will find an acceptable solution, although for many problems it is very likely that such a solution will be found. More significantly, the solution quality can vary considerably between EA runs. This problem is caused by the multimodality inherent to many electronic design problems. In many cases, a small change to a solution can result in severe changes to the solution properties. Therefore, many electronics problems have an extremely irregular search space, and the robustness of a search technique is extremely important.

In order to estimate the likelihood that a particular solution can be discovered, a series of runs can be performed. The objective space can then be divided into different regions, depending upon the proportion of runs that produce results dominating each point in the objective space.

The results from 1000 runs of 750 generations with the basic system, are shown in figure 4(a). These results show that there is a large amount of variation between the runs. For example, while the best solutions have areas below $12000\mu m^2$, 50% of runs did not produce designs with areas as low as $17000\mu m^2$.

The lines in figure 4(a) divide up the solution space according to the number of runs that dominate a particular point. For example, each point on the 1% line is dominated by the results of 10 out of the 1000 runs. Points to the upper-right of the 1% line can be dominated by more runs, while points below and to the left of the 1% line are dominated by fewer runs.

Disabling elitism results in a considerable improvement in the quality of many of the solutions. The results without using elitism are shown in figure 4(b). Although there are relatively minor improvements in the performance of the best solutions, the worst solutions are considerably improved. Note that for both tests, all 1000 runs produced correct solutions. It is possible that elitism discourages exploration, and many of the elitist runs become stuck in local minima.

4.2 Solution Diversity

There are very strong conflicts between functionality and area, and between functionality and longest-path delay. In other words, there are many cases where removing components from a design will increase the functional error.

The area and longest-path delay objectives do not inevitably conflict. There are some cases in which these objectives do conflict. One such situation is illustrated in figure 3. Figure 3 shows a minimum-area circuit and a minimum-delay circuit, both of which compute $f = a + b + c + d$ and $g = b + c + d$. There are also many cases where the area and delay do not conflict; physically larger circuits tend to have longer critical paths.

The best results in figure 4 show very little diversity. The best solutions are concentrated at a single point on the objective space. This suggests that the area and delay objectives are not conflicting in this case. The results suggest that there are limits to the silicon area and longest-path delay for a correct design. The limits are largely independent, leading to a small trade-off surface.

Figures 4(c) and 4(d) show the results of combining the area and longest-path delay into a single objective. The combined objective is computed as:

$$\text{area} + 4000.\text{delay}$$

These runs produced far fewer solutions in comparison with the original EA. Although all runs produced correct results, the number of non-dominated results per run was much lower. The quality of the solutions was very slightly lower with the modified system, possibly as a result of having slightly more homogeneous populations. Finally, the elitist system in figure 4(c) did not perform as well as the non-elitist version in figure 4(d).

The original system was also tested on three other problems. For these tests elitism was disabled. The problems are as follows:

- the multiplication block for a 24th order FIR filter,
- transformation from an RGB colour model to an XYZ colour model [13],
- the 8-point 1-dimensional DCT.

The FIR filter mentioned above is a low-pass filter with 30dB attenuation in the stop band. The first 13 of the 25 coefficients are:

$$26, 54, 45, 36, -14, -62, -95, -73, 14, 159, 321, 448, 496$$

The coefficient set is symmetrical about the 13th coefficient. When factors of $\pm 2^x$ are divided out, only the following coefficients are actually required:

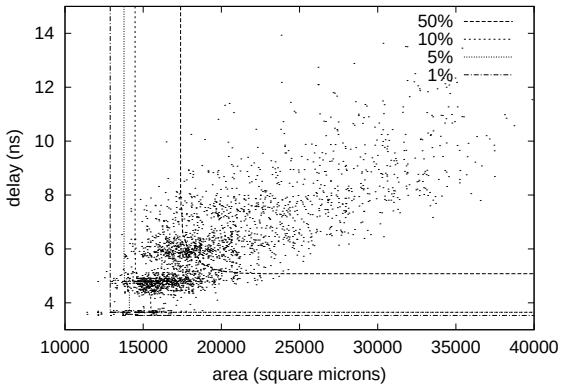
$$7, 9, 13, 27, 31, 45, 73, 95, 159, 321$$

The results for these problems are shown in figure 5. It can be seen that the amount of diversity in the solution set depends upon the problem.

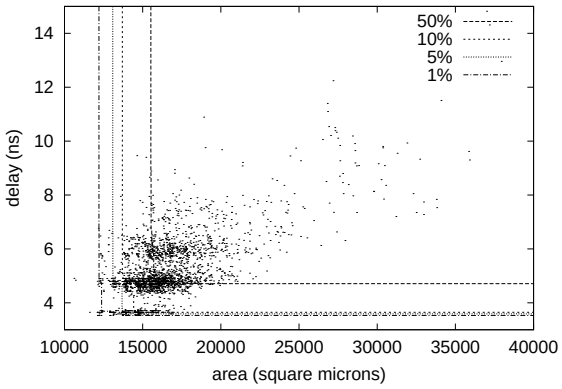
4.3 Hardware Modelling

The quality of the hardware models is important. If a model is inaccurate, the EA system is less likely to produce good results.

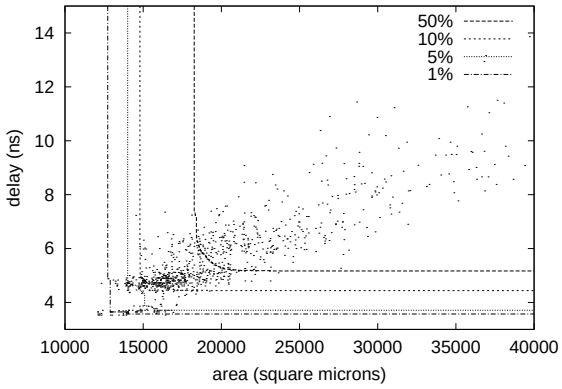
The correct 4-point DCT designs from 100 runs of the EA system were extracted as Verilog netlists. The properties of these designs were then modelled by the Synopsys



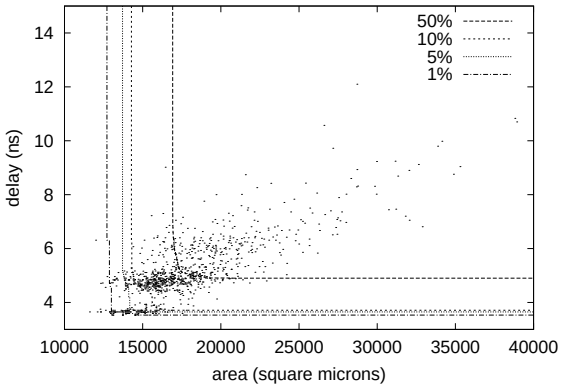
(a) Non-dominated sorting, elitist.



(b) Non-dominated sorting, non-elitist.

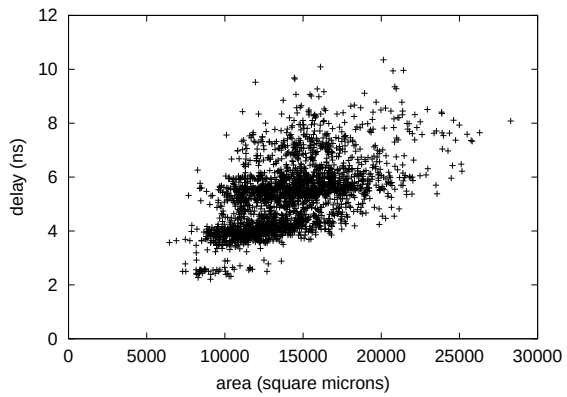


(c) Linear, elitist.

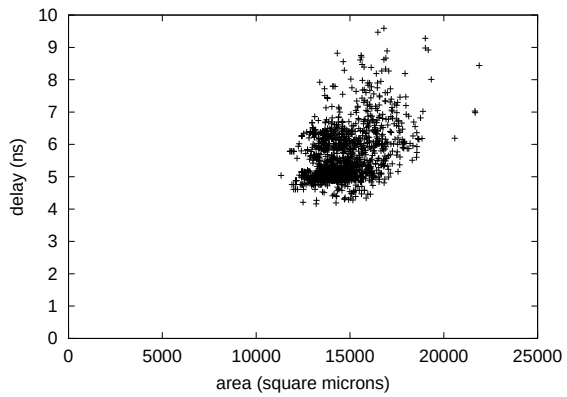


(d) Linear, non-elitist.

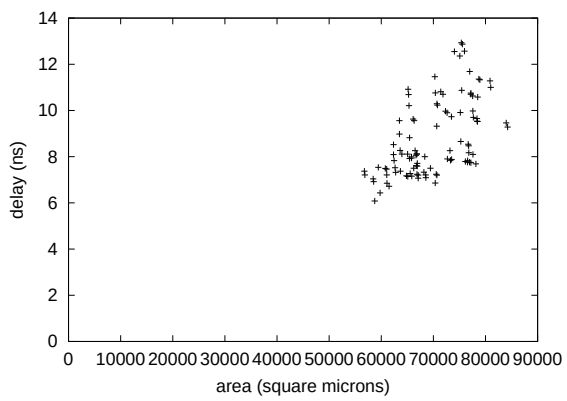
Figure 4: Correct DCT designs for different selection schemes. As all of these designs meet the functional constraint, the functional error values are not illustrated here.



(a) Results of 1000 runs with the 24th order FIR problem.

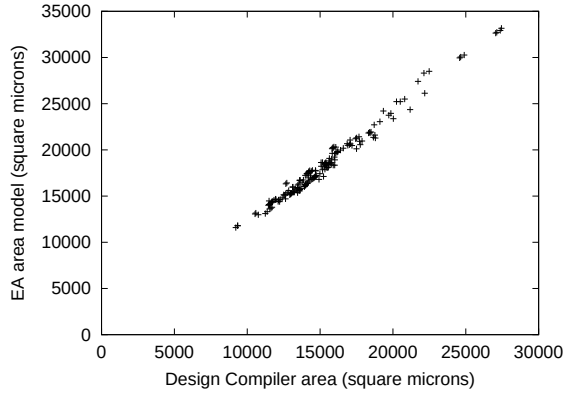


(b) Results of 1000 runs with the RGB to XYZ transformation.

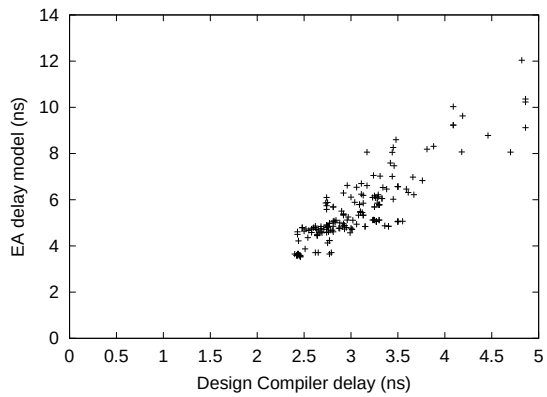


(c) Results of 20 runs with the 8-point DCT.

Figure 5: Correct results for various problems.



(a) The area model compared with Design Compiler.



(b) The delay model compared with Design Compiler.

Figure 6: A comparison of the hardware models with Synopsys Design Compiler.

Design Compiler synthesis system. This produced higher-quality pre-placement estimates to the area and longest-path delay. The results of comparing the EA models with Design Compiler are shown in figure 6.

The EA area model produces results that are very similar to the design compiler results. There is no interconnect area information in the low-level technology library, so neither of these models incorporates an estimate of the interconnect area.

The results of the delay comparison are shown in figure 6(b). The two different models often give significantly different results. Possible reasons for this were discussed earlier. Note that the EA overestimates the delay by nearly a factor of 2. As mentioned in section 3.2, the delay model could be improved by incorporating wire-load modelling, and by calculating delays on a per-wire basis.

Finally, the area-delay properties calculated by Design Compiler are shown in figure 7. The use of a wire-load model might be expected to increase the diversity of the solutions, as designs with few components might have more

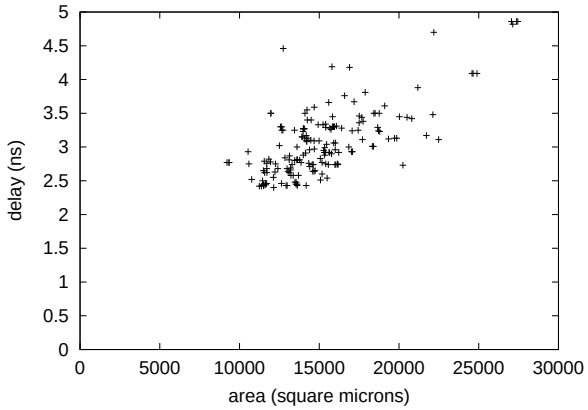


Figure 7: Results using the Design Compiler hardware models.

high-fanout connections. The non-dominated solutions in figure 7 show a trade-off of about 15% of the absolute area and delay values, suggesting that there is a small increase in the population diversity.

5 Conclusions

The paper has studied issues of robustness, reliability, population diversity, and modelling, within the context of a multi-objective EA for the design of digital VLSI systems. The EA considers three objectives; functionality, area, and delay.

The area and longest-path delay do not always conflict, and can potentially be combined into one objective. The amount of conflict between the objectives is problem dependent — this was demonstrated using several real-world problems. The search space can be characterised as having a lower limit for area, and a lower limit for delay. In many cases this causes the non-dominated set to be concentrated in a small area.

There is a large amount of variation between individual EA runs. This is a symptom of the complex, multi-modal search space. The likelihood that particular solutions are discovered is something that can be experimentally assessed. This was demonstrated with the results in figure 4.

Elitism was found to be counterproductive. This is probably due to reduced population diversity.

The accuracy of the area and longest-path delay models was assessed. It was found that the area model performs acceptably. The delay model is inaccurate. There are two ways in which the delay model could be improved. If the delays were modelled on a per-wire basis, rather than on a per-value basis, the results would be more accurate. In particular, the tendency to overestimate the delay would be reduced. A second way in which the delay model could be improved would be to incorporate a model of wire-loads. In spite of the inaccuracy, the delay model is still useful in encouraging the development of low-delay designs.

The results produced by Design Compiler suggest that, if the wire-load modelling is improved, the area and delay objectives are more likely to conflict. If the lowest area solutions for a particular problem have high-fanout connections on the critical path, then there will probably be a conflict between the objectives. The alternative possibility is that the lowest area solution does not have high-fanout connections on the critical path, in which case objective conflicts are less likely.

6 Acknowledgements

The authors would like to acknowledge the support of the EPSRC, under grant number 0030467X.

Bibliography

[1] David E. Goldberg, *Genetic Algorithms in search, optimization, and machine learning*, Addison-Wesley, 1989.

[2] T. Bäck, D. B. Fogel, and T. Michalewicz, *Evolutionary Computation 1*, Institute of Physics Publishing, 2000.

[3] Kalyanmoy Deb, *Multi-Objective Optimization Using Evolutionary Algorithms*, John Wiley and Sons, 2001.

[4] T. Kalganova and J. Miller, "Evolving more efficient digital circuits by allowing circuit layout evolution and multi-objective fitness," in *Proceedings of the First NASA/DoD Workshop on Evolvable Hardware*, A. Stoica, D. Keymeulen, and J. Lohn, Eds., July 1999, pp. 54–63.

[5] T. Higuchi, M. Murakawa, M. Iwata, I. Kajitani, Weixin Liu, and M. Salami, "Evolvable hardware at function level," in *IEEE International Conference on Evolutionary Computation*, Apr. 1997, pp. 187–192.

[6] M. S. Bright and T. Arslan, "Synthesis of low-power DSP systems using a genetic algorithm," *IEEE Transactions on Evolutionary Computation*, vol. 5, no. 1, pp. 27–40, Feb. 2001.

[7] Kai Hwang, *Computer Arithmetic: Principles, Architecture, and Design*, John Wiley and Sons, Jan. 1979.

[8] A. G. Dempster and M. D. Macleod, "Constant integer multiplication using minimum adders," *IEE Proceedings — Circuits, Devices and Systems*, vol. 141, no. 5, pp. 407–413, Oct. 1994.

[9] D. R. Bull and D. H. Horrocks, "Primitive operator digital filters," *IEE Proceedings — Circuits, Devices and Systems*, vol. 138, no. 3, pp. 401–412, June 1991.

[10] Michael R. Garey and David S. Johnson, *Computers and Intractability / A Guide to the Theory of NP-Completeness*, W. H. Freeman and Company, 1979.

[11] A. G. Dempster and M. D. Macleod, "Use of minimum-adder multiplier blocks in FIR digital filters," *IEEE Transactions on Circuits and Systems II:*

Analog and Digital Signal Processing, vol. 42, no. 9, pp. 569–577, Sept. 1995.

[12] M. Potkonjak, M. B. Srivastava, and A. P. Chandrakasan, "Multiple constant multiplications: efficient and versatile framework and algorithms for exploring common subexpression elimination," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 15, no. 2, pp. 151–165, Feb. 1996.

[13] R. W. G. Hunt, *The Reproduction of Colour*, chapter 8, p. 143, Fountain Press, fifth edition, 1995.