AREA ESTIMATION AND OPTIMISATION OF FPGA ROUTING FABRICS

Alastair M. Smith, George A. Constantinides and Peter Y. K. Cheung

Dept. of Electrical and Electronic Engineering, Imperial College London
email: {alastair.smith, g.constantinides, p.cheung} @imperial.ac.uk

ABSTRACT

This paper presents a methodology for estimating and optimising FPGA routing fabrics using high-level modelling and convex optimisation techniques. Experimental methods for exploring design spaces suffer from expensive computation time, which is exacerbated by increased dimensionality due to the larger number of architectural parameters. In this paper we build on previously published work to describe a model of FPGA routing area. This model is used in conjunction with a form of optimisation known as geometric programming, in order to analytically derive optimised FPGA architectural parameters, demonstrating the power and accuracy of model-based approaches in configurable architecture design. We show that routing parameters such as connection and switch box flexibilities can be architected to save around 6% of area instead of using traditional "rules of thumb".

1. INTRODUCTION

Recently, a number of research publications have appeared that advocate the use of some form of analytical model to determine FPGA architecture parameters [1, 2, 3, 4, 5]. The main contribution of [2, 3, 4] has been to demonstrate that equation-based high-level models can relatively accurately capture some of the design trade-offs present in FPGA architectures, without resorting to computationally intensive empirical methods based on tools such as VPR [6]. Such an ability becomes increasingly important as the number of parameters that define an FPGA architecture increases (e.g. due to heterogeneous architectures), leading the empirical approach to suffer from the "curse of dimensionality". Due to their accuracy, empirical tools are never likely to disappear from use, but there is a demand for a high-level "first order" design exploration tool for FPGA architecture.

In this paper, we make the observation that the models proposed in the FPGA literature thus far are, by-and-large, not only suitable to reduce the run-time of parameter sweep approaches to architecture design, but offer a radically different approach - one based upon rigorous mathematical optimization theory. This potential essentially arises from two observations: (i) since the models are equation and inequality-based, they admit the potential for first and second-order derivative information to be used to steer search direction, (ii) several of the existing models can be transformed into a convex optimization problem, resulting in global optima. We demonstrate in this paper that, taken together, these criteria allow modern convex optimization techniques, based on interior-point methods, to perform a first-order exploration of the architecture space without suffering as a result of design-space dimensionality.

We exemplify this philosophy with a method for simultaneously determining the optimal choice of connection box flexibility for logic block inputs, connection box flexibility for logic block outputs, and routing channel width. The proposed approach combines the model of [3] for channel width with a simple model of the impact of logic block size, input and output connection box flexibility and channel width on FPGA routing area. The intuition is that reduced routing flexibility reduces the area of the multiplexers associated with connection and switch boxes, but at the same time exerts pressure on increased channel width. The proposed method automatically balances these concerns to propose an optimal choice of parameters for any given logic block size.

2. RELATED WORK

FPGA architectures have received a considerable amount of research effort. This is exemplified by the architecture exploration tool VPR [6]. VPR is a parameterisable CAD flow, and has been used to show, for example, the best combination of fine-grain logic parameters [7]. A subset of the architecture parameters that can be varied within VPR are given in Table 1. Whilst VPR provides accurate information on the delay and area of a wide variety of FPGA architectures given a set of benchmarks, the entire CAD process of technology mapping, placement and routing must be performed for each circuit and for each architecture specification. Coupled with the number and range of the parameters that can be varied, architecture exploration is a time consuming process and has motivated new techniques to explore architecture design spaces more efficiently.

Formal optimisation techniques have been used to effi-

This work has been funded by the EPSRC under grant numbers EP/C549481/1 and EP/E00024X/1

978-1-4244-3892-1/09/$25.00 ©2009 IEEE

Table 1. Model Parameters

Architectural Parameters:	
K	Number of inputs per lookup table
N	Number of lookup tables per logic block
I	Number of inputs per logic block
$F_{c,in}$	Number of tracks that connect to each logic input pin
$F_{c,out}$	Number of tracks each logic block can connect to
F_s	Number of track end-points that connect to each track driver
Circuit Parameters:	
p	Rent parameter of a given circuit
n_2	Number of 2-LUTs in a given circuit

ciently search design spaces. [1] showed how heterogeneous architecture layouts can be formulated in Integer Linear Program (ILP) form. [5] utilises an improved formulation to explore heterogeneous FPGA layouts. Given a fixed time budget, this improved formulation was shown to greatly improve on parameter sweep approaches. This provides significant motivation for the use of such formal techniques in architecture design. The key to such techniques is a good representative model of the underlying problem.

Research into the modelling of FPGA fabrics has been performed on both logic and routing architectures. In [4], a model was developed to estimate the number of architecture blocks required for a given high-level description of a circuit. The work estimates the number of architecture blocks required for a benchmark circuit given the number of 2-input lookup tables required n_2 and the Rent parameter p. The final result of [4] was to combine the model for the number of logic blocks with a routing area model to determine a first-order approximation of the area of an FPGA. Routing area models have also seen significant advancement over the past few years. In [3], a set of equations was developed that describes the number of programmable routing tracks necessary in homogeneous FPGA architectures depending on a number of parameters that describe the routing architecture. The work assumed all benchmark circuits have a constant wirelength, regardless of the FPGA architetcure parameters. Motivated by the observation that wirelength varies depending on benchmark characteristics, wirelength models from the ASIC community were employed in [2] to improve the channel width estimation.

The models outlined above suffer from two drawbacks. Firstly, the approximations of area are relatively crude: they use the number of programming bits required as an area approximation, which does not give an accurate estimate of the actual silicon area required. Secondly, they require a fixed set of architecture parameters to obtain a result, meaning that a sweep across each parameter is still necessary to determine an optimal architecture. It is precisely these issues that we address in this paper: by employing more accurate models of the components in an FPGA in conjunction with a Geometric Programming (GP) framework, we show how to capture tradeoffs in the routing fabric automatically to explore FPGA routing fabrics.

2.1. Geometric Programming

A *geometric program* (GP) is a constrained optimization problem of the following form:

$$\begin{aligned} \text{Minimize :} \quad & f_0(x) \\ \text{Subject to :} \quad & \\ & f_i(x) \leq 1, for\ i = 1, 2, ..., m \\ & g_i(x) = 1, for\ i = 1, 2, ..., l \end{aligned}$$

where x is a strictly positive n-vector of real values, and the functions f_i and g_i have special mathematical forms, known as *posynomials* and *monomials*, respectively.

A monomial is a function

$$g(x) = cx_1^{a_1}x_2^{a_2}\ldots x_n^{a_n}$$

where the coefficient c must be strictly positive. A posynomial is simply a sum of a finite number of monomials.

Notice that GP generalises the better known class of optimization problems known as *linear programs*. GPs are important in practice because some of the same efficient algorithms that are known for solving linear programs can also be applied to GPs, for example the class of interior point methods [8]. In particular, this means that GPs can be solved in polynomial time in theory, and extremely fast in practice [9]. For our purpose, formulation of architecture exploration as a GP would have serious implications for the speed at which the architecture space can be explored: the optimal solution to a GP can be found in time that grows only with $\sqrt{m}$ and cubically in $n + l$ [8]. Contrast this with a parameter sweep which, for a fixed number of k of points in each architecture parameter, has execution time that grows as k^n.

Of course, GP is not a panacea: the restriction to strictly positive leading coefficients in a monomial is a serious one and explicitly disallows many problems, including general *polynomial* optimization rather than *posynomial* optimization. However, we show in this paper that the models developed by the FPGA community thus far can indeed be cast in GP form.

3. ROUTING AREA MODEL

We assume an island-style FPGA in which an array of blocks is connected using tracks organized in horizontal and vertical channels with single-driver routing, as represented by VPR 5.0 [6]. The logic blocks in the architecture consist of K-input lookup tables (LUTs) packed into tightly connected configurable logic blocks (CLBs), each with N LUTs and with I external inputs. The routing blocks in the architecture can be described by three parameters: the number of tracks that can connect to each logic block input, $F_{c,in}$; the number of tracks that each logic block output can connect to, $F_{c,out}$; and the number of track end-points that connect to each track driver, F_s. The architecture parameters are summarised in Table 1.

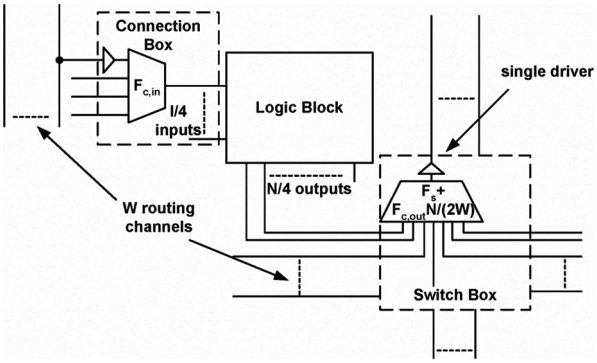

Fig. 1. Routing Fabric of an FPGA.

Figure 1 shows a detailed model of the routing fabric. We consider the amount of silicon area devoted to the routing fabric of the reconfigurable device to consist of all switch box and connection box multiplexers, in addition to their output buffers and configuration memories. Buffer sizing is assumed to be fixed for each set of fine-grain logic parameters. Routing area is thus dependent on the size of the grid of logic cells, the channel width and the size of the multiplexers used to connect signals to and from logic blocks and IO pins. We discuss each of these in turn.

3.1. Estimating Grid Size

The size of the grid is fixed for a given logic architecture and is dependent on the number of logic cells n_c. To enable the models to be retargetable across a wide variety of logic architectures we use [4] to estimate n_c. Since grid dimensions are discrete, the model used in approximating the number of required cells is likely to underestimate the number of architecture cells used. The actual grid size can thus be expressed as $N_c = \lceil \sqrt{n_c} \rceil^2$, where $\lceil \cdot \rceil$ represents the *ceiling* function. The combined area of all logic block connection boxes used in routing is the product of N_c and the connection block size. Since I/O pins are spread along the chip perimeter, the area devoted to the input connection boxes is the product of $4\sqrt{N_c}$ and the size of the I/O connection block. There are two types of switch boxes in the architecture: those at the edge of the device and those in the centre. Since the routing array grid is one unit of width larger than the logic array grid, the number of switch boxes at the edge of the device is $N_{s,e} = 4\left(\lceil \sqrt{n_c} \rceil + 1\right)$, and the number of switch boxes in the centre of the device is $N_{s,m} = \left(\lceil \sqrt{n_c} \rceil - 1\right)^2$.

3.2. Channel Width Model

To estimate the channel width of the device, we employ the model developed in [3]. This model is shown in (1) for architectures with wires that span one logic block, where the nominal minimum channel width W_{min} is described by

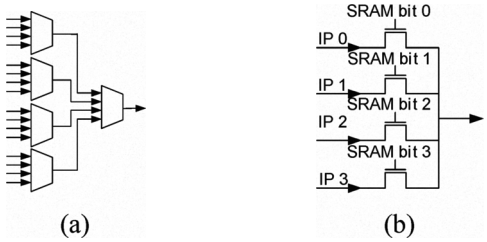

Fig. 2. Multiplexing schemes in VPR 5.0: (a) a two-level multiplexing scheme for a 16:1 multiplexer, (b) a pass transistor-based 4:1 multiplexer.

(2), and β, α_{in}, α_{out} are empirically derived constants. In (2), λ represents the average number of inputs used on each logic block, p_f represents an empirical constant referred to as the *peak factor* and $\bar{R}$ represents the average point-to-point wirelength.

$$W = W_{min} + \frac{1}{\beta}\left(\frac{W_{min}}{F_s}\right)\left(\frac{W_{min}}{F_{c,in}}\right)^{\alpha_{in}}\left(\frac{W_{min}}{F_{c,out}}\right)^{\alpha_{out}} \tag{1}$$

$$W_{min} = p_f \frac{\lambda \bar{R}}{2} \tag{2}$$

The average point-to-point wirelength is dependent on the logic architecture parameters and is thus constant for K, N and I. The methods described by [2] are used to calculate the value of point-to-point wirelength for different logic parameters.

3.3. Routing Multiplexer Area

A commonly employed method for implementing routing multiplexers in FPGAs is to use a two-level multiplexing scheme. An example of a 16:1 multiplexer using this scheme is shown in Figure 2(a). In such a scheme, the first and second level of multiplexers are balanced such that each stage has approximately the same size of multiplexer. Multiplexers can be efficiently implemented using transistors, as shown in Figure 2(b). This structure is taken from VPR 5.0, in which one-hot encoding of configuration bits is used. These observations lead to the expression for multiplexer area given in (3), where S_{SR} represents the size of an SRAM cell, S_t represents the size of the pass transistors and P is the number of inputs. In this equation $\lceil \cdot \rceil$ and $\lfloor \cdot \rfloor$ represent the *ceiling* and *floor* functions, and are used to account for non-square numbers of inputs: the size of the second multiplexer is always no larger than that of each input multiplexer. The combined expression can be approximated as in (4), which is used for the area models described by the GP formulation in Section 4.

$$S_{mux} = S_t\left(P + \lfloor \sqrt{P} \rfloor\right) + S_{SR}\left(\lceil \sqrt{P} \rceil + \lfloor \sqrt{P} \rfloor\right) \tag{3}$$

$$\approx S_t\left(P + \sqrt{P}\right) + 2S_{SR}\sqrt{P} \tag{4}$$

In the routing fabric there are two types of multiplexers: switch box multiplexers and connection box multiplexers. The size of the connection box multiplexers is dependent only on the architectural parameter $F_{c,in}$ from Table 1. Channel width is dependent on $F_{c,in}$ according to (1), hence we refer instead to $F'_{c,in} = \frac{F_{c,in}}{W}$, the proportion of routing tracks that can connect to each input pin. Each connection box multiplexer thus has $WF'_{c,in}$ inputs. This leads to the approximation of multiplexer area given in (5), where the size of the pass transistors in each block is given by $S_{t,sb}$.

The size of the switch box multiplexers is dependent on the architecture parameters F_s and $F_{c,out}$. $F'_{c,out} = \frac{F_{c,out}}{W}$ represents the proportion of routing tracks that each output can drive. VPR allows connections on each side of a CLB in two directions: either both horizontal (E/W) or both vertical (N/S) directions, assuming that pins are equally spread around the logic block, meaning each switch box multiplexers has $\frac{N}{2}F'_{c,out} + F_s$ inputs. This leads to (6) as an approximation to the multiplexer size for the switch boxes where the size of the pass transistors in each block is given by $S_{t,sb}$. $S_{sb,m}$ represents the area consumed by a switch box multiplexer in the centre of the array: switch box multiplexers on the outermost parts of the grid have a similar expression which accounts for fewer CLB outputs and the presence of I/O pins.

$$S_{cb} = S_{t,cb}\left(WF'_{c,in} + \sqrt{WF'_{c,in}}\right) + 2S_{SR}\sqrt{WF'_{c,in}} \quad (5)$$

$$S_{sb,m} = S_{t,sb}\left(\frac{N}{2}F'_{c,out} + F_s + \sqrt{\frac{N}{2}F'_{c,out} + F_s}\right) + 2S_{SR}\sqrt{\frac{N}{2}F'_{c,out} + F_s} \quad (6)$$

For each multiplexer used in the fabric, there is an associated driving buffer which also consumes area. In the routing fabric, one isolation buffer is required for each connection box input and one driving buffer is required for each wire segment. All other buffers are assumed to be contained within the logic cells. In this work we focus on optimising the routing fabric in terms of high-level architectural parameters such as $F'_{c,out}$ and $F'_{c,in}$, and will assume that the logic fabric and buffer sizing is fixed.

3.4. Combined Area Model

Combining these models leads to (7) for the routing area. Each connection box for an IO pin consumes area $S_{cb,io}$, which can be evaluated using the multiplexer area model. $2WS_{sb,m}$ represents the product of the size of a centre-array switch box and the channel width W, with the factor of two representing both horizontal and vertical routing directions. Since there are only three of four directions for channels on the edge of the device, $1.5WS_{sb,e}$ represents the product of the edge switch box multiplexer size, the channel width, with the factor of 1.5 representing both routing dimensions. For the sake of simplicity, we assume the corner switch boxes have the same size as any other edge switch box. The areas consumed by the connection block, edge connection block, middle switch box and edge switch box buffers are represented respectively by B_{cb}, $B_{cb,io}$, $B_{sb,m}$ and $B_{sb,e}$, and I_{IO} represents the number of I/O pins per I/O block. The buffer sizes are assumed to be fixed for a given set of logic architecture parameters.

Table 2. Summary of Notation in the Geometric Program

Transistor Sizing (const.)	$S_{t,cb}$ $B_{cb,io}$	$S_{t,sb}$ $B_{sb,e}$	S_{SR} $B_{sb,m}$	B_{cb}
Grid Size Parameters (const. for benchmark)	N_c	$N_{s,m}$	$N_{s,e}$	
Model Constants (from [3])	W_{min}	α_{in}	α_{in}	β
Architecture Constants	F_s	I	I_{io}	N
Architecture Variables	W	$F_{c,in}$	$F_{c,out}$	
Additional GP Variables	Q	P	A_r	

$$A_r = IN_c\left(S_{cb} + B_{cb}\right) + 4I_{io}\sqrt{N_c}\left(S_{cb,io} + B_{cb,io}\right) + 2N_{s,m}W\left(S_{sb,m} + B_{sb,m}\right) + 1.5N_{s,e}W\left(S_{sb,e} + B_{sb,e}\right) \quad (7)$$

4. GEOMETRIC PROGRAMMING FORMULATION

The model presented in Section 3 is not in a form that is amenable to GP. GP requires that the objective and inequality constraints are in *posynomial* form, while all equality constraints are in *monomial* form. In order to make the channel width expression fit into GP, (1) can be expressed as (9), since $W > 0$. Considering the various terms that represent device area, several approximations are necessary. The non-posynomial $ceiling$ and $floor$ functions are approximated as described in Section 3.3. These can be expressed in posynomial form as in (10), (11) and (13). The introduction of variable $Q \geq \frac{N}{2}F'_{c,out} + F_s$ is necessary to express the square root over a sum in (6) in the correct form: (12) represents this substitution, since $Q > 0$, and since the lowest Q leads to the lowest routing area, meaning that at optimality this inequality will be satisfied with strict equality. (14) and (15) represent the area devoted to switch boxes at the edge of the device, and include a similar substitution, using the variable P. The above approximations lead to the standard form GP representation given by (8)-(16). Constants and variables in the model are summarised in Table 2. Note that the optimisation problem only has 6 variables, independent of the size of the circuit or FPGA.

5. RESULTS

To evaluate the GP framework, we performed a comparison of our model to VPR 5.0. In order to implement the GP framework, we employed the convex programming tool CVX [10], a free plug-in toolset for MATLAB. To verify our model is a good approximation relative to VPR, we used a parameter-sweep approach: whilst GP can be used to derive the optimal parameters of interest, variables can be fixed in

$$\text{min}: \quad A_r \tag{8}$$

subject to :

$$W_{min}W^{-1} + \beta^{-1}F_s^{-1}W_{min}^{(1+\alpha_{in}+\alpha_{out})}W^{-1}F_{c,in}^{-\alpha_{in}}F_{c,out}^{-\alpha_{out}} \leq 1 \tag{9}$$

$$S_{t,cb}S_{cb}^{-1}WF'_{c,in} + S_{t,cb}S_{cb}^{-1}W^{\frac{1}{2}}F'^{\frac{1}{2}}_{c,in} + 2S_{SR}S_{cb}^{-1}W^{\frac{1}{2}}F'^{\frac{1}{2}}_{c,in} \leq 1 \tag{10}$$

$$S_{t,cb}S_{cb,io}^{-1}WF'_{c,in} + S_{t,cb}S_{cb,io}^{-1}W^{\frac{1}{2}}F'^{\frac{1}{2}}_{c,in} + 2S_{SR}S_{cb,io}^{-1}W^{\frac{1}{2}}F'^{\frac{1}{2}}_{c,in} \leq 1 \tag{11}$$

$$\frac{N}{2}F'_{c,out}Q^{-1} + F_sQ^{-1} \leq 1 \tag{12}$$

$$S_{t,sb}S_{sb,m}^{-1}Q + S_{t,sb}S_{sb,m}^{-1}Q^{\frac{1}{2}} + 2S_{SR}S_{sb,m}^{-1}Q^{\frac{1}{2}} \leq 1 \tag{13}$$

$$\frac{N}{4}F'_{c,out}P^{-1} + I_{io}F'_{c,out}P^{-1} + F_sP^{-1} \leq 1 \tag{14}$$

$$S_{t,sb}S_{sb,e}^{-1}P + S_{t,sb}S_{sb,e}^{-1}P^{\frac{1}{2}} + 2S_{SR}S_{sb,e}^{-1}P^{\frac{1}{2}} \leq 1 \tag{15}$$

$$IN_cS_{cb}A_r^{-1} + IN_cB_{cb}A_r^{-1} + 4I_{io}\sqrt{N_c}S_{cb,io}A_r^{-1} + 4I_{io}\sqrt{N_c}B_{cb,io}A_r^{-1} + 2N_{s,m}WS_{sb}A_r^{-1} + 2N_{s,m}B_{sb,m}WA_r^{-1} + 1.5N_{s,e}WS_{sb}A_r^{-1} + 1.5N_{s,e}B_{sb,e}WA_r^{-1} \leq 1 \tag{16}$$

Fig. 3. Geometric program for routing area optimisation.

the GP framework to evaluate the model as a closed form equation and determine the accuracy of the model across the parameters of interest. In these experiments, we assume a fixed logic architecture: $K = 4, N = 10, I = 22$. The experimental setup routes for minimum channel width using the standard binary search algorithm within VPR. Figures 4(a) and (b) show the results of our parameter sweep across the entire range of $F'_{c,in}$ and $F'_{c,out}$ for the model and experiment respectively. In the figures, each contour represents a difference of 10K transistors. To generate the data, 20 MCNC circuits have been employed, with an architecture generated for each. The results show that our model gives an accurate representation of the total FPGA area; the absolute values of area and trends are present in both the GP model and VPR. The model correctly identifies that the minimum area architecture has much lower $F_{c,out}$ than $F_{c,in}$. However, the model breaks down for extremely low values of $F_{c,out}$.

Figure 4(c) shows minimum channel width as a function of $F'_{c,in}$ and $F'_{c,out}$ for the model, with contours representing a difference of two channels, and Figure 4(d) shows the experimental values. It is evident from these two figures that the channel width model [3] under-estimates $F'_{c,out}$ in the region $0 - 0.04$ and is the main cause of the break-down of the area model in this region. Nevertheless, the figures showing channel width also provide some additional insight into the underlying tradeoffs of the optimisation space: fully flexible routing will give a smaller channel width, however this is at the expense of increased multiplexer sizes.

A particularly useful observation for FPGA architects is that the GP model is considerably faster than the VPR parameter sweep. Each point within the GP takes approximately 10 seconds to generate, whereas on the same machine the runtime of VPR for each experiment is in the range of two minutes (for a small benchmark with fully flexible routing) to four hours (for a large benchmark with very restrictive routing parameters). This gives the model a significant advantage for early-stage architecture exploration, a 30× to 120× speedup. However the advantage of GP in this context is more than just in the run-time for each experimental point. GP *does not* require the parameter sweep as a prelude: if the goal is to determine the optimal set of architecture parameters, then this also takes around 10 seconds to find. In contrast, a sweep of just 10 values of W, $F_{c,in}$ and $F_{c,out}$ would take 3 CPU hours to 17 CPU days to compute. Moreover, the runtime in the worst-case is not dependent on benchmark size: since a benchmark circuit is only represented by two parameters the GP is insensitive to circuit size and thus the runtime improvements will be greater for larger benchmarks. Previous study on bidirectional routing architectures examined the relationship between the parameters $F'_{c,in}$ and $F'_{c,out}$ [11]. It was suggested that the two should be set to the same value, which as a function of the logic block granularity N, should be $F_{c,in} = F_{c,out} = \frac{W}{N}$. To examine this intuition against uni-directional routing, we have applied our model across a variety of logic block sizes and compared to this scheme for selecting F_c.

Figure 4(e) shows how the optimal values for $F_{c,out}$ and $F_{c,in}$ vary with logic block size. Included in the figure are the values of F_c for the bi-directional rule-of-thumb from [11]. The results show that the best architecture in terms of device area is one for which both flexibility parameters are roughly constant regardless of logic block size. Figure 4(f) shows the optimised routing area and compares it to that obtained by choosing F_c by this rule of thumb. It is interesting to see that whilst the flexibility values are significantly different between the two architectures, the area consumed by both sets of architectures is quite close. Nevertheless, the potential savings of the optimised scheme are around 5-6.5%.

6. CONCLUSION

This paper has presented a method for early stage exploration of FPGA routing architectures. It has been shown that this problem can be approximated in a form that is amenable to geometric programming, and as a consequence can be used to determine architectural parameters for routing architectures in an efficient manner. The approach shows good potential for reducing area in FPGA architectures. However, there are several parts of the framework that could be

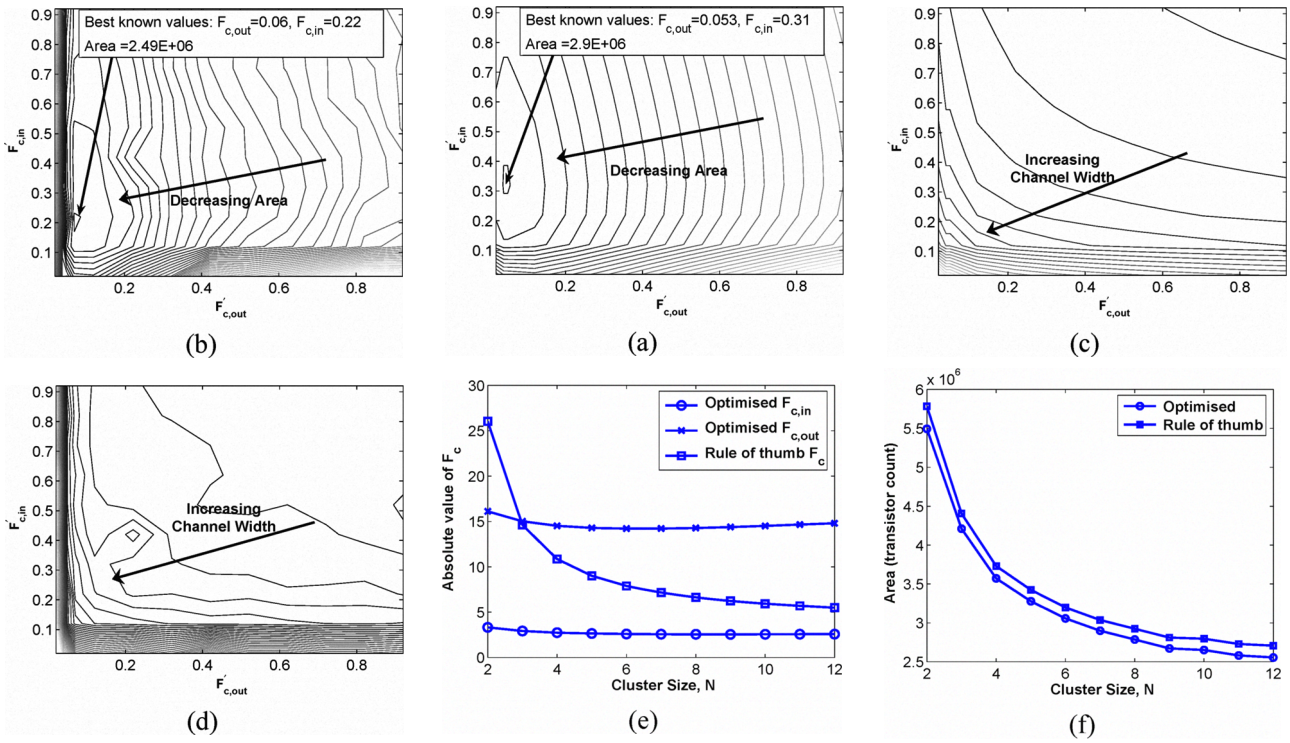

Fig. 4. (a) Contour plots of theoretical routing area (N=10), (b) Contour plot of experimental routing area (N=10), (c) Contour plot of theoretical channel width (N=10), (d) Contour plot of experimental channel width (N=10) (e) Optimal and "rule of thumb" F_c parameters according to logic block size, (f) Estimated routing area for optimal and "rule of thumb" F_c parameters.

extended to express the problem in more detail. The issue of buffer sizing is still to be addressed and current modelling techniques do not account for delay. Geometric programming has previously been shown to be capable of expressing both of these issues and thus provides a powerful framework for continuing this work.

7. REFERENCES

[1] A. M. Smith, G. A. Constantinides, and P. Y. K. Cheung, "Integrated floorplanning, module-selection, and architecture generation for reconfigurable devices," *IEEE Trans. on Very Large Scale Integration (VLSI) Systems*, vol. 16, no. 6, pp. 733–744, Jun 2008.

[2] A. M. Smith, J. Das, and S. J. E. Wilton, "Wirelength modeling for homogeneous and heterogeneous FPGA architectural development," in *Int'l Symp. on Field-Programmable Gate Arrays*, Feb. 2009, pp. 181–190.

[3] W. Fang and J. Rose, "Modeling FPGA routing demand in early-stage architecture development," in *Int'l Symp. on Field-Programmable Gate Arrays*, Feb. 2008, pp. 139–148.

[4] A. Lam, S. J. Wilton, P. Leong, and W. Luk, "An analytical model describing the relationships between logic architecture and FPGA density," in *Int'l Conf. on Field-Programmable Logic and Applications*, Sep. 2008.

[5] A. Kahoul, G. A. Constantinides, A. M. Smith, and P. Y. K. Cheung, "Heterogeneous architecture exploration: Analysis vs parameter sweep," in *Int'l Workshop. on Applied Reconfigurable Computing*, Mar. 2009.

[6] J. Luu, I. Kuon, P. Jamieson, T. Campbell, A. Ye, M. Fang, and J. Rose, "Vpr 5.0: FPGA CAD and architecture exploration tools with single-driver routing, heterogeneity and process scaling," in *Int'l Symp. on Field-Programmable Gate Arrays*, Feb. 2009, pp. 133–142.

[7] E. Ahmed and J. Rose, "The effect of LUT and cluster size on deep-submicron FPGA performance and density," *IEEE Trans. on Very Large Scale Integration (VLSI) Systems*, vol. 12, no. 3, pp. 288–298, Mar. 2004.

[8] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge University Press, 2004.

[9] S.-J. Kim, S. P. Boyd, S. Yun, D. D. Patil, and M. A. Horowitz, "A heuristic for optimizing stochastic activity networks with applications to statistical digital circuit sizing," *Optim Eng*, vol. 8, no. 4, pp. 397–430, Dec 2007.

[10] M. Grant and S. Boyd, "CVX: Matlab software for disciplined convex programming (web page and software)," Feb. 2009, http://stanford.edu/~boyd/cvx.

[11] V. Betz, J. Rose, and A. Marquardt, *Architecture and CAD for Deep-submicron FPGAs*. Kluwer Academic Publishers, 1999.