Reconfigurable FPGA 시스템을 위한 위상기반 회로분할

최 연 경 (崔 然 景)
경민대학 전자계산과
전화 : (0351) 828-1333, Fax : (0351) 828-1380, email : ykchoi@kyungmin-c.ac.kr
임 종 석(林 鐘 錫)
서강대학교 컴퓨터학과
전화 : (02) 705-8490, Fax : (02) 704-8273, email : csrim@ccs.sogang.ac.kr

Topology-Based Circuit Partitioning for Reconfigurable FPGA Systems

Yeun Kyung, Choi
Department of Computer Science, Kyungmin College
Chong Suk, Rim
Department of Computer Science and Engineering, Sogang University

Abstract

This paper proposes a new topology-based partition method for reconfigurable FPGA systems whose components and the number of interconnections are predetermined. Here, the partition problem must also consider nets that pass through components such as FPGAs and routing devices to route 100%. We formulate it as a quadratic boolean programming problem suggest a paritition method for it. Experimental results show 100% routing, and up to 15% improvement in the maximum number of I/O pins.

I. 서 론

FPGA(Field Programmable Gate Array) 기술의 발전으로 재프로그래밍(reprogramming)이 가능한 FPGA는 대용량 시스템의 프로토타이핑(prototyping)을 위한 보드나 재구성 가능한(reconfigurable) 시스템 등에 유용하게 사용되고 있다. 이와 같은 FPGA를 기반으로 하는 시스템들은 FPGA, 외부와의 연결을 위한 입출력 전용 FPGA 그리고 이들간의 연결 전용 칩 등 여러 모듈들로 구성되어 있으며 이들간을 연결할 수 있는 연결선들이 고정되어 있다. 이러한 시스템을 사용하기 위해서는 입력 회로를 주어진 보드의 칩들에 놓기 위한 회로 분할 방법이 필요하다.

FPGA 기반 시스템으로 회로를 분할할 경우에는 분할된 회로를 각 FPGA 칩에서 구현한 후에 이들간을 100% 연결할 수 있도록 칩의 크기와 핀수를 고려하여 분할하여야 한다. 이를 위하여 FPGA칩을 통과하거나 연결전용칩을 사용하는 우회(bypass)하는 네트의 수도 함께 고려하여야 한다.

기존의 분할 방법으로는 이동에 의한 분할 방법들이 있으며[3,4], 수학적인 기법을 사용한 분할방법들[6,7]이 있으나 우회하는 네트에 대한 고려가 없다.

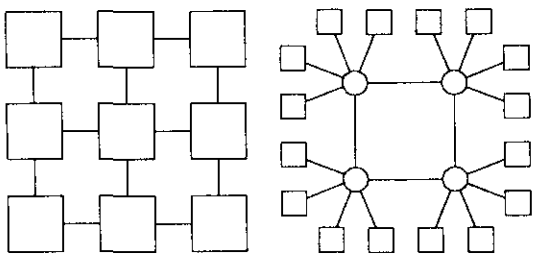
본 논문에서는 기존의 분할 문제에서 이차 이진 프로그래밍(quadratic boolean programming)를 이용하여 분할과 회로에 대한 정의를 일반화한 분할 문제[6,7]을 수정하여 우회하는 네트들을 추가로 고려하는 분할 문제를 정의하고 이를 해결하는 분할 방법을 제안한다. 분할 방법은 문제를 선형화한 후 이를 해결하는 방법으로 구성된다.

먼저 2장에 분할문제를 정의한다. 3장에서 분할 방법을 기술하고 이의 실험결과를 4장에 보인 후 결론을 낸다.

II. 분할 문제

FPGA 기반 시스템들은 다양한 연결 구조를 갖고 있는데 이러한 시스템을 위상그래프 $G=(I,E)$로 표시할 수 있다[3]. 즉, 각 모듈을 정점의 집합 I로 하고 두 모듈간에 연결선이 있을 경우 이들간에 에지로 연결하여 이들의 집합을 E로 한다. 구성한 그래프의 두 가지 예를 그림 1에 보인다. 그림에서 사각형은 FPGA 칩을, 원은 연결전용칩을 나타낸다. 회로를 이와 같은 위상그래프로 분할하는 문제를 위상기반 회로분할문제라고 한다[3].

보드에 존재하는 모듈의 수를 K라고 하자. 각 모듈 $i \in I$는 회로를 넣을 수 있는 용량 c_i, 사용가능한 핀수 p_i와 외부로 나가는 입칩출력의 수 t_i가 정해져 있다.



(a) Mesh 구조. (b) AXB-AP4 구조[9].

그림 1. 위상그래프.

모듈과 모듈간에는 보드상에서 연결을 위한 거리(distance)를 나타낼 수 있으며 이를 $K\times K$ 행렬 $B=(b_{i_1i_2})$로 표현한다. 이때 $b_{i_1i_2}$는 두 모듈 i_1과 모듈 i_2간의 연결을 위한 거리이다.

입력 회로는 $j\in J$ 로 나타내며 회로의 수를 N 으로 하자. 모든 회로는 크기 s_j를 갖는다. 회로들간의 연결된 네트의 수는 $N\times N$ 행렬 $A=[a_{j_1j_2}]$로 표현하며 이때 $a_{j_1j_2}$는 회로 j_1와 j_2간의 연결선의 수를 나타낸다. 만일 회로의 네트가 외부 입출력으로 사용되면 이를 외부 입출력으로 사용하는 핀들의 수에 이를 포함하여야 하므로 외부 입출력이 필요한 회로에 대하여 필요한 입출력의 수를 g_j로 한다.

회로 j 의 모듈 i 로의 할당(assign)은 x_{ij}로 나타낸다. 만일 회로 j 가 모듈 i 에 있을 때에는 $x_{ij}=1$이고 그렇지 않을 경우에는 0이다.

회로 분할 시 회로에 연결된 네트들이 다른 모듈을 우회하는 경우 이들 모듈의 사용된 핀 수에 우회하는 네트를 위한 핀 수를 포함시켜야 한다. 이를 위하여 어떤 모듈이 우회 연결을 위하여 사용되는지 알아야 한다. 그러나 분할 과정에서 배선되는 형태를 알기 힘들므로 어떤 모듈이 우회 연결을 위하여 사용되는 지를 확률로 나타낸다.

$F^{i_1}=[f^{i_1}_{i_2i_3}]$를 모듈 i_1가 우회 연결을 위하여 사용될 확률 행렬이라고 하자. $f^{i_1}_{i_2i_3}$는 모듈 i_2에서 모듈 i_3 사이에 연결해야 할 네트가 존재할 경우에 모듈 i_1가 우회 연결을 위하여 사용될 확률이다. 이는 모듈 i_2과 모듈 i_3 사이에 존재하는 모든 최단경로 중 모듈 i_1가 경로에 포함되는 최단 경로의 수로 한다. 만일 모듈 A와 B 사이를 연결하는데 최단경로의 수가 3이고 이때 모듈 C를 포함하는 최단경로의 수가 2인 경우 f^C_{AB}는 2/3이다.

앞의 정의들을 사용하여 회로를 위상그래프로 분할할 때에 만족해야할 제한들을 나열하면 다음과 같다.

제한 1 :

$$\sum_{j=1}^{N} s_j\, x_{ij} \le c_i, \quad \forall i\in I .$$

제한 2 :

$$\sum_{i_2=1}^{K}\sum_{j_1=1}^{N}\sum_{j_2=1}^{N} a_{j_1j_2}b_{i_1i_2}x_{i_1j_1}x_{i_2j_2} +$$
$$2\sum_{i_2=1}^{K}\sum_{j_1=1}^{N}\sum_{i_3=1}^{K}\sum_{j_2=1}^{N} f^{i_1}_{i_2i_3}a_{j_1j_2}x_{i_2j_1}x_{i_3j_2} \le b_{i_1}, \forall i_1\in I .$$

제한 3 :

$$\sum_{j=1}^{N} x_{ij}\, g_j \le t_i, \quad \forall i\in I .$$

제한 4 :

$$\sum_{i=1}^{K} x_{ij}=1, \quad \forall j\in J .$$

여기서 회로 J 를 그래프 G로 분할하는 문제는 위의 제한들을 모두 만족하면서 다음의 식을 최소화하는 문제이다.

$$\min \sum_{i_1=1}^{K}\sum_{j_1=1}^{N}\sum_{i_2=1}^{K}\sum_{j_2=1}^{N} a_{j_1j_2}b_{i_1i_2}x_{i_1j_1}x_{i_2j_2} \qquad (1)$$

식 (1)은 다음과 같이 일차원 벡터 y 를 사용하는 식으로 변형할 수 있다.

$$\min\ y^{T}Qy \qquad (2)$$

여기서 y 는 x_{ij} 행렬을 $r=i+(j-1)\times K$ 로 하는 y_r들의 벡터로 변형시킨 것이며 $Q=[q_{r_1r_2}]$는 $q_{r_1r_2}=a_{j_1j_2}b_{i_1i_2}$ 로써, 이때 $r_1=i_1+(j_1-1)\times K$, $r_2=i_2+(j_2-1)\times K$ 이다. 그러므로 재프로그래밍이 가능한 FPGA 기반 보드에서의 위상 기반 회로 분할 문제는 제한 1~4를 만족하면서 식(2)가 최소화되는 해 y 를 구하는 것과 동일하다.

Ⅲ. 회로 분할

본 장에서는 앞에서 정의한 위상 기반 회로 분할 문제를 해결하는 방법을 기술한다. 식(2)의 문제는 Balas와 Mazzola[1]에 의하여 선형(linear) 문제로 변형이 가능하며 변형된 문제는 Buckard와 Bonniger[2]의 휴리스틱에 의하여 해를 구할 수 있다. 이 휴리스틱에서는 두 개의 선형 문제를 반복적으로 해결하는 방법으로 해를 구한다.

그런데 두 개의 선형 문제는 할당 문제(assignment problem)와 동일하여 Shih와 Kuh[6]의 분할 방법에서는 Martello와 Toth[5]의 일반화된 할당 문제를 적용하여

회로 분할의 해를 구하였다. 또 이 방법은 용량과 I/O를 고려한 분할 방법[7]에서도 적용되었다. 그러나 이 방법에서도 앞서 정의한 문제의 제한 2와 3을 고려하지 않았다. 그러므로 본 논문에서는 Buckard와 Booniger의 기본적인 휴리스틱을 따르나 제한 2와 3을 고려한 분할 방법을 제안한다.

3.1 Buckard와 Bonniger의 휴리스틱

그림 2에 Buckard와 Bonniger의 휴리스틱을 보인다.

1. $m := 1$, $h_r^{(1)} := 0 (1 \le r \le KN)$을 초기화한다. 상한계 $\theta_r (1 \le r \le KN)$를 계산하고 초기해 $u^{(1)}$를 최적해 $u^* = u^{(1)}$, $z^* := u^{(1)T} Q u^{(1)}$로 초기화한다.
2. 모든 $1 \le r \le KN$에 대하여 다음을 계산한다.
$$\delta_l^{(m)} := \sum_{r=1}^{KN} q_{rl} u_r^{(m)}, \quad (1 \le l \le KN).$$
$$\phi^{(m)} := \sum_{r=1}^{KN} \theta_r u_r^{(m)}.$$
3. 다음식을 구한다.
$$z := \min_{u \in S} \sum_{r=1}^{KN} \delta_r^{(m)} u_r.$$
4. 단계 3의 결과를 이용하여 다음을 계산한다.
$$h_r^{(m)} := h_r^{(m)} + \frac{1}{\max(1, |z - \phi^{(m)}|)} \delta_r^{(m)}, \quad (1 \le r \le KN).$$
5. 다음식을 구한다.
$$\min \sum_{r=1}^{KN} h_r^{(m)} u_r^{(m+1)}$$
이때 구한 해를 $u^{(m+1)}$라고 한다.
6. 만일 $z^{(m+1)} := u^{(m+1)T} Q u^{(m+1)} < z^*$이면 $u^* := u^{(m+1)}$, $z^* := u^{(m+1)T} Q u^{(m+1)}$로 해를 갱신한다.
7. 만일 $m <$ MITER이면 $m := m+1$로 갱신하고, 아니면 단계 2로 간다.

그림 2. Buckard와 Bonniger의 휴리스틱.

여기서 단계 3과 5는 결국 위상기반 회로분할방법의 제한들을 고려하는 할당문제이므로 Martello와 Toth의 방법을 수정하여 우리의 문제에 맞도록 사용하였으며 이를 다음절에 기술한다.

3.2 위상기반 회로분할방법을 위한 할당방법

본 논문에서는 분할 시 회로를 놓는 분할만을 고려하지 않고 연결전용칩 등을 포함한 모듈을 분할의 대상으로 하여 분할 문제의 모든 제한들을 만족하도록 해야 한다. 이를 위하여 모든 회로에 대하여 제한을 넘지 않는 모듈들 중에서 벡터 δ(또는 h)의 값이 최소가 되는 모듈과 두 번째 최소가 되는 값을 구한다. 그 최소값과 두 번째 최소값의 차를 구한 후 차가 가장 큰 회로를 먼저 선택하여 최소값을 갖게 하는 모듈로 회로를 할당한다.

그런데 모든 회로의 제한이 만족되는지를 매번 고려하는 것은 제한 2의 우회 네트에 대한 고려 때문에 많은 시간을 필요로 한다. 그런데 회로를 할당하는 초기에는 모듈들에 할당된 회로가 적고 핀수 제한을 모두 만족하므로 매번 제한을 확인할 필요가 없다. 그러므로 회로 할당을 시작하기 전에 모든 회로에 대하여 제한들을 만족하는 모듈들 중에서 벡터 δ(또는 h)의 값이 최소가 되는 모듈과 두 번째 최소가 되는 값을 구한다. 모든 회로에 대하여 그 최소값과 두 번째 최소값의 차를 구한 후 그 차에 대하여 회로를 감소순(nonincreasing order)으로 정렬한다. 정렬된 순서에 의하여 회로를 선택하여 모듈에 회로를 할당한다.

회로를 할당한 후에는 각 모듈에 대하여 추가할 수 있는 회로의 크기와 핀수를 수정하여야 한다. 회로가 할당된 모듈에 대해서는 c_i에서 s_j를, p_i에서 새로 추가되는 핀의 수를 빼주면 된다. 회로가 외부 입출력이 필요한 경우에는 t_i에서 외부로 나가는 핀의 수를 빼주면 된다.

만일 우회로 사용되는 모듈이 있을 경우에는 이 모듈들에 대하여 p_i를 수정하여야 한다. 우회하는 네트들은 각 네트를 위한 연결선의 수의 두 배만큼의 핀수가 필요하므로 이 네트의 연결선의 수가 $a_{j_1 j_2}$인 경우 p_i에서 $2 \times f^i_{i_2 i_3} \times a_{j_1 j_2}$를 빼준다. 그림 3에 제한한 할당 방법을 보인다.,

모든 회로가 할당되었으면 할당을 중지한다. 만일 정렬된 순서로 할당을 하는 과정에서 제한에 의하여 회로가 모듈로 들어갈 수 없는 경우에는 할당되지 않은 회로에 대하여 다시 벡터 δ(또는 h)의 값에 의하여 최소값들을 구하고 재정렬한다. 그렇지 않을 경우에도 할당되는 회로에 의하여 할당되지 않은 다른 회로들이 영향을 받게 되어 재정렬할 필요가 발생할 수 있으므로 할당 도중에 모듈의 크기에 따라서 재정렬을 반복 수행한다.

지금까지 기술한 분할 방법의 복잡도는 행렬 연산에 의하여 $O(K^2N^2)$이다. 그러나 행렬들의 값들이 대부분 0이므로 이들에 대해서는 연산을 하지 않으므로 실제의 복잡도는 표기한 복잡도보다 적다.

Ⅳ. 실험결과

본 논문에서 제안한 분할 방법은 C언어로 구현하였으며 Sun Ultra-2에서 실험하였다. 입력으로는 Partition93의 Xilinx XC-3000[8] 시리즈의 FPGA로 기술사상된 회로를 사용하였으며 이들 회로를 표 1에 보인다.

회로	CLBs	IOBs	Nets
s5378	381	84	626
s9234	454	41	714
s13207	915	152	1375
s15850	842	99	1262
s35932	2317	192	3360

표 1. 입력회로.

본 논문에서 구현한 방법은 Shih와 Kuh[7]의 분할 방법(SK 분할)을 구현하여 Mesh 구조와 AXB-AP4 구조에서 비교하였다. 실험은 회로의 크기를 고려하여 작은 회로 s5378과 s9234는 4개, 나머지 큰 회로들은 16개의 가상 FPGA을 사용하였다. 이때 각 모듈의 크기와 핀수는 회로의 크기에 비례하도록 조정하였다.

Mesh 구조에서 외부와의 연결은 가장자리에 놓인 모듈에서만 가능하도록 제한하였으며 AXB-AP4 구조에서는 각 연결전용칩에 하나의 입출력전용칩이 연결되어 있어서 외부와의 연결이 가능하다고 가정하였다.

IOB인 회로들은 외부와의 연결을 위하여 외부와의 입출력이 가능한 칩에만 넣도록 하였으며 SK 분할 방법에서도 동일하게 처리하였다. 두 분할 방법에서 모두 MITER는 100회로 한정하였다. 또, 실제 배선 시 두 방법의 차이를 보기 위하여 분할된 회로에 대하여 각 위상 그래프상에서 미로배선방법을 사용하여 배선을 하였다. 배선 시 모듈을 사용하는 핀수 사용율이 100%가 넘게 되면 다른 모듈을 사용하도록 블럭킹(blocking)하였다.

표 2와 3에 실험결과를 보인다. 표에서 '핀수(I)'는 SK 방법에 의하여 분할된 결과에서 배선 전에 최대 핀수를 사용하는 모듈의 핀수 사용율을 나타낸다. 핀수(I)에는 모듈을 우회하는 네트들은 포함되지 않았다. '핀수(B)'는 본 논문에서 구현한 방법에서의 배선 전의 최대 핀수를 갖는 모듈의 핀수의 사용율을 나타내며, 이 경우에는 우회하는 네트들이 포함되어 있다.

'핀수(R)'은 배선 후에 최대 핀수를 갖는 모듈의 핀수를 사용율을 나타낸다. AXB-AP4 구조에서는 연결전용칩이 존재하므로 이들 중에서 최대 핀수를 사용한 칩의 핀수 사용율을 '핀수(S)'로 한다. '시간'은 각 방법에서의 수행시간을 초단위로 나타낸다.

표 2에서처럼 SK 분할 방법의 경우에는 핀 사용율이 100%가 넘어서 배선이 완료될 수 없는 경우도 있으나 본 논문에서 구현한 분할 방법에서는 배선이 완료됨을 알 수 있다. 또 핀수(I)에는 우회하는 네트가 포함되지 않아서 핀수(B)에 비하여 적지만 배선 후에 핀수(R)을 보면 비슷하거나 적다.

표 3에서는 연결전용칩이 존재하는데 본 논문에서는 이를 고려하여 분할하므로 핀수(S)가 최대 15% 적음을 알 수 있다. 이는 본 논문에서 구현한 방법이 가능한 적은 수의 연결 전용 칩을 통과하는 해를 구하기 위하여노력하기 때문이다.

회로	SK 분할			Our 분할		
	핀수(I)	핀수(S)	시간	핀수(B)	핀수(S)	시간
s5378	89.6%	59.7%	159	99.5%	56.6%	233
s9234	94.5%	37.0%	167	80.2%	33.2%	187
s13207	89.4%	70.9%	1850	92.2%	59.0%	2862
s15850	84.8%	67.5%	1594	95.8%	51.8%	2509
s35932	96.9%	79.7%	3215	97.7%	67.5%	6662

표 2. AXB-AP4 구조에서의 분할결과.

회로	SK 분할			Our 분할		
	핀수(I)	핀수(S)	시간	핀수(B)	핀수(S)	시간
s5378	89.6%	59.7%	159	99.5%	56.6%	233
s9234	94.5%	37.0%	167	80.2%	33.2%	187
s13207	89.4%	70.9%	1850	92.2%	59.0%	2862
s15850	84.8%	67.5%	1594	95.8%	51.8%	2509
s35932	96.9%	79.7%	3215	97.7%	67.5%	6662

표 3. AXB-AP4 구조에서의 분할결과.

결론

본 논문에서는 FPGA를 기반으로 하는 재프로그래밍이 가능한 시스템을 위한 분할 문제인 위상기반 회로분할 문제를 이차 이진 분할 문제로 정의하였으며 이를 위한 분할 방법을 제안하였다. 실험결과에 보인 것과 같이 모든 회로에 대하여 분할 후 배선이 완료되었으며 다른 방법과 비교하여 사용하는 핀수를 최대 15% 줄일 수 있음을 보였다.

참고문헌

[1] E. Balas and J.B. Mazzola, "Quadratic 0-1 Programming by a New Linearization", Presented at the TIMS/ORSA Meeting, 1980.

[2] R.E. Buckard and T. Bonniger, " A Heuristic for Quadratic Boolean Programs with Applications to Quadratic Assignment Problems", European Journal of Operational Research 13, pp.374-386, 1983.

[3] Y.Choi and C.S. Rim, "A Topology-Based Multi-Way Partition for ASIC Prototyping, Proc. of MSCS, pp. 357-360, 1996.

[4] C. Kim and H. Shin, "A Performance-Driven Logic Emulation System: FPGA Network Design and Performance-Driven Partitioning", IEEE Trans. on CAD of IC and Systems, Vol. 15, No. 5, May, 1996.

[5] S. Martello and P. Toth, *Knapsack Problems*, Chap. 7, pp. 189-220, 1990.

[6] M. Shih and E.S. Kuh, "Quadratic Boolean Programming for Performance-Driven System Partitioning", 30th DAC, pp. 761-765, 1993.

[7] M. Shih and E.S. Kuh, "Circuit Partitioning by Capacity and I/O constraints", IEEE CICC, pp. 659-662, 1994

[8] Xilinx, Inc., The Programmable Gate Array Data Book. Xilinx, San Jose, 1992.

[9] Aptix, Inc., System Data Book, Aptix, 1993.