디테일드 라우팅 유전자 알고리즘의 설계와 구현

Design and Implementation of a Genetic Algorithm for Detailed Routing

송호정, 송기용
Ho-Jeong Song, Gi-Yong Song
충북대학교 컴퓨터공학과

요 약

디테일드 라우팅은 VLSI 설계 과정중의 하나로, 글로벌 라우팅을 수행한 후 각 라우팅 영역에 할당된 네트들을 트랙에 할당하여 구체적인 네트들의 위치를 결정하는 문제이며, 디테일드 라우팅에서 최적의 해를 얻기 위해 left-edge 알고리즘, dogleg 알고리즘, greedy 채널 라우팅 알고리즘등이 이용된다. 본 논문에서는 디테일드 라우팅 문제에 대하여 유전자 알고리즘(genetic algorithm; GA)을 이용한 해 공간 탐색(solution space search) 방식을 제안하였으며, 제안한 방식을 greedy 채널 라우팅 알고리즘과 비교, 분석하였다.

Abstract

Detailed routing is a problem assigning each net to a track after global routing. The most popular algorithms for detailed routing include left-edge algorithm, dogleg algorithm, and greedy channel routing algorithm. In this paper we propose a genetic algorithm searching solution space for the detailed routing problem. We compare the performance of proposed genetic algorithm(GA) for detailed routing with that of greedy channel routing algorithm by analyzing the results of each implementation.

Keywords : genetic algorithm, detailed routing, greedy channel routing

I. 서론

최근 VLSI 회로 설계는 전자설계자동화(EDA; Electronic Design Automation) 툴을 사용하여 효과적으로 이루어지고 있다. 이러한 EDA 툴은 회로의 레이아웃을 위하여 분할, 배치, 배선등의 여러 단계의 과정을 수행하게 된다.

배치(placement) 단계에서 회로 블록과 핀들의 위치가 결정되면, 각 블록들 사이에 라우팅 영역이 생성된다. 배치 단계에서 생성된 라우팅 영역에 네트들의 위치를 결정하는 것을 라우팅(routing)이라 한다.

라우팅 문제는 연결선의 구체적인 위치를 결정하지 않고 각 네트들을 라우팅 영역에 할당시키는 글로벌 라우팅(global routing)과 각각의 라우팅 영역에 할당된 네트들의 구체적인 위치를 결정하는 디테일드 라우팅(detailed routing)의 두 단계로 나눌 수 있다[1][2].

디테일드 라우팅 문제는 VLSI 회로의 physical design 단계에서 모든 네트들을 최소한의 트랙에 할당하여 라우팅 영역을 최소화하는 문제이다. 라우팅 영역은 VLSI 칩의 면적에 영향을 미치므로, 라우팅 영역의 최소화는 VLSI 회로 레이아웃을 위한 EDA 툴에서 매우 중요하다.

본 논문에서는 VLSI의 설계 과정 중 글로벌 라우팅을 수행한 후 각 라우팅 영역에 할당된 네트들을 트랙에 할당하여 구체적인 네트들의 위치를 결정하는 디테일드 라우팅 문제에 대하여 유전자 알고리즘을 이용한 해 공간 탐색 방식을 제안하였으며, 이 방식을 greedy 채널 라우팅 알고리즘과 비교, 분석하였다.

본 논문의 구성은 다음과 같다. II장에서 디테일드 라우팅 문제에 대하여 알아보고, III장에서는 본 논문에서 제안한 디테일드 라우팅 유전자 알고리즘의 데이터 표현 방법과 알고리즘에 대하여 설명한다. IV장에서는 제안한 디테일드 라우팅 알고리즘과 greedy 채널 라우팅 알고리즘을 비교하였고, 마지막으로 V장에서는 결론과 향후 연구 방향에 대하여 기술한다.

II. 디테일드 라우팅 문제

디테일드 라우팅 문제는 글로벌 라우팅을 수행한 후 각 라우팅 영역에 할당된 네트들을 최소한의 트랙에 할당하여 구체적인 네트들의 위치를 결정하는 문제이다. 즉 최적의 라우팅은 모든 네트들을 연결하였을 때 사용된 트랙의 수가 최소가 되도록 연결시키는 것으로써, 라우팅 결과에 따라 칩 면적이 영향을 받는다.

디테일드 라우팅 문제를 해결하기 위한 알고리즘으로는 left-edge 알고리즘[4], dogleg 알고리즘[5], greedy 채널 라우팅 알고리즘[6]등이 있다.

각 알고리즘을 살펴보면, left-edge 알고리즘은 네트들을 핀의 순서에 따라서 정렬을 한 후, 가장 왼쪽의 네트부터 트랙에 할당 가능여부를 조사하여 가능한 트랙에 할당하거나 새로운 트랙에 할당하는 알고리즘으로서, 수행시간은 상당히 빠르나 지역해에 빠지기 쉽다. dogleg 알고리즘은 left-edge 알고리즘과 비교했을 때 dogleg을 사용하여 트랙의 개수를 줄일 수 있다는 특징이 있으나 마찬가지로 지역해에 빠지기 쉽다. greedy 채널 라우팅 알고리즘은 column by column으로 진행을 하면서, 해당 column에서 최적의 라우팅을 탐색하는 방법으로 매우 우수한 발견적 탐색 방법의 알고리즘이다.

디테일드 라우팅 알고리즘은 사용 가능한 layer의 개수에 따라 달라질 수 있으며, 본 논문에서는 2-layer를 사용한 VH(또는 HV) 모델에서의 라우팅 알고리즘을 기술한다.

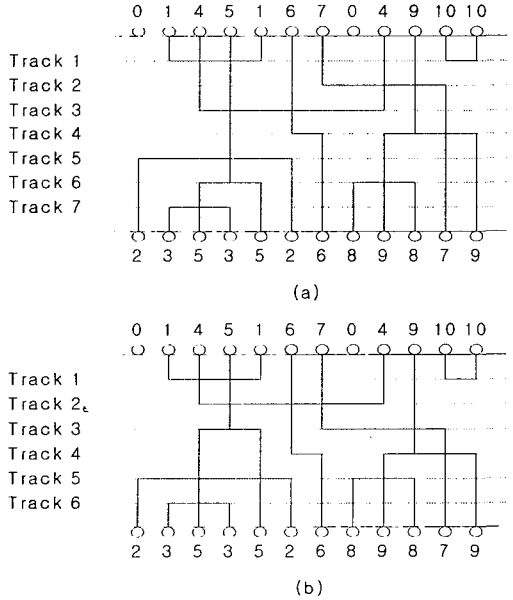

그림 1. (a) 7개의 트랙에 할당된 네트
(b) 6개의 트랙에 할당된 네트
Fig. 1 (a) The nets assigned to 7 tracks.
(b) The nets assigned to 6 tracks.

그림 1은 주어진 네트를 트랙에 할당한 결과를 보인다. 그림 1에서 각 라우팅 결과 사용한 트랙의 수는 1(a)는 7이고, 1(b)는 6이다. 이와 같이 네트를 트랙에 할당하는 방법에 따라 사용된 트랙의 개수가 달라질 수 있다. 여기서 채널의 위와 아래에 표시된 숫자들은 같은 네트에 연결된 포트들을 의미한다. 즉, 위쪽 2번째와 5번째의 '1'은 1번 네트에 연결된 포트로서 채널을 사용하여 연결되어야 한다.

그림 2는 dogleg을 사용하지 않은 라우팅과 dogleg를 사용한 라우팅을 보이고 있다. 그림 2(a)에서는 dogleg을 사용하지 않고 하나의 네트를 하나의 트랙에 할당을 한 결과를 보이며, 그림 2(b)는 굵은 선으로 표시한 2번 네트를 dogleg을 사용하여 라우팅한 결과를 보인 것이다. 이와 같이 dogleg를 사용하면 라우팅에 필요한 트랙의 수를 줄일 수 있다.

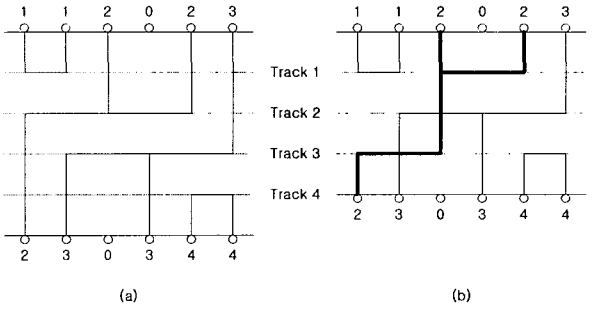

그림 2. (a) dogleg을 사용하지 않은 라우팅 예
(b) dogleg을 사용한 라우팅 예
Fig. 2 (a) Routing example without dogleg.
(b) Routing example using dogleg.

III. 디테일드 라우팅 유전자 알고리즘

유전자 알고리즘은 진화 과정에서 유도된 탐색방법으로, 이 알고리즘은 염색체와 유사한 자료구조를 사용하여 해공간을 부호화하며, 부호화한 자료 구조에 재조합 연산자를 적용하여 염색체들을 진화시킨다.

유전자 알고리즘은 처음에 임의로 선택된 염색체 집단(population of chromosome)에서 시작하며, 이러한 염색체 집단 중에서 일정한 방식으로 부모 염색체를 선택하고 이들 부모 염색체를 교배시켜 자식 염색체를 생성한다. 새로 생성된 자식 염색체는 평가함수에 의해 평가되며 좋은 평가 결과를 가지는 염색체가 다음 세대에 살아남을 확률이 높게 된다. 이와 같은 방식으로 유전자 알고리즘은 염색체 집단의 진화를 통하여 최적해에 근접할 수 있으므로, 최적 해를 구하기 어려운 여러 NP-문제에 적용될 수 있다.

1. 염색체의 표현

디테일드 라우팅 문제는 글로벌 라우팅을 수행한 후

라우팅 영역에 할당된 네트들을 트랙에 할당하는 문제로서, 각 네트들이 할당된 트랙을 표현할 수 있어야 하며, 또한 dogleg을 사용하여 다른 트랙에 할당이 된 것도 표현 가능해야 한다. 디테일 라우팅 유전자 알고리즘의 염색체를 표현하기 위하여 그림 3과같이 각 트랙과 열을 각각 숫자와 알파벳순으로 표기하였다.

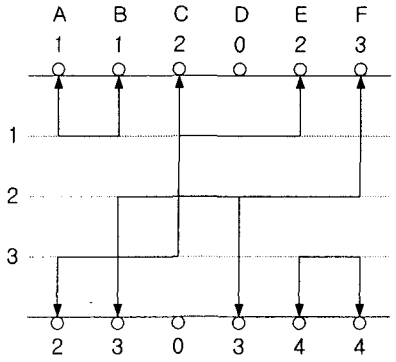

그림 3. 네트리스트와 트랙
Fig. 3 Netlist and tracks.

그림 4는 그림 3의 네트리스트를 본 논문에서 제안한 디테일드 라우팅 유전자 알고리즘의 염색체 표현 방법으로 표현한 것이다. 여기서는 간단한 설명을 위하여 최대 2번의 dogleg만이 가능한 경우를 예로 보이고 있다.

그림 4의 염색체 표현에서 각 5개의 유전인자는 하나의 네트를 나타내며, 첫 번째 유전인자는 네트의 시작 트랙을 나타내고 시작 열의 위치는 네트리스트에서 확인할 수 있다. 또한 나머지 유전인자는 dogleg된 열과 트랙을 나타내는 2개의 유전인자 쌍으로 이루어져있다.

즉, 1번 네트는 처음에 1번 트랙을 사용하고, 계속 1번 트랙만을 사용하는 것을 나타내며, 2번 네트는 처음에 3번 트랙을 C열에서 1번 트랙을 사용한 것을, 3번 네트는 2번 트랙을, 4번 네트는 3번 트랙을 사용한 것을 나타내고 있다.

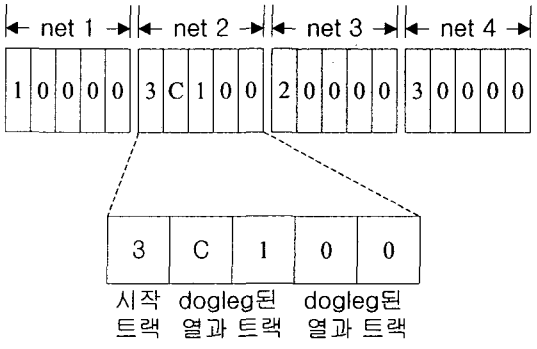

그림 4. 염색체의 표현
Fig. 4 Representation of chromosome.

2. 교배 연산자(Crossover Operator)

유전자 알고리즘에서 각 세대의 모집단은 교배를 통해 각 부모의 유전 정보를 상속받게 된다. 이러한 교배로 생성되는 자손들 중 좋은 형질을 상속받은 자손은 다음 세대에 살아남을 높은 확률을 가지게 되고, 그렇지 않은 자손은 낮은 확률을 가지게 된다.

본 논문에서는 2-점 교배 연산자를 사용하여 교배 연산을 수행하였다. 2-점 교배 연산자는 2개의 교배점 사이의 유전인자를 서로 교환하는 방법으로 이루어진다.(그림 5).

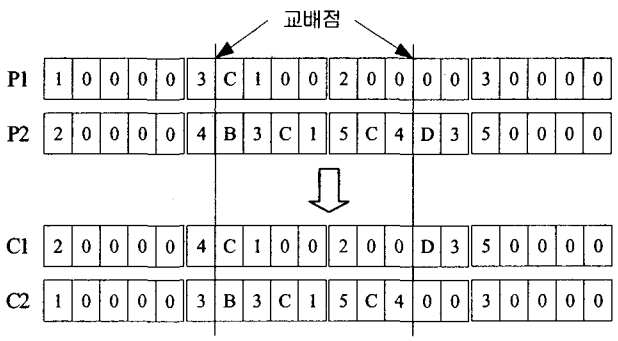

그림 5. 2-점 교배 연산
Fig. 5 2-point crossover operation.

그림 5의 교배연산 결과 *C1*의 3번 네트와 같이 유전자의 중간에 dogleg 정보가 없는 유전인자가 존재할 경우 이 유전인자를 뒤로 밀어 유전인자를 재정렬 한다(그림 6).

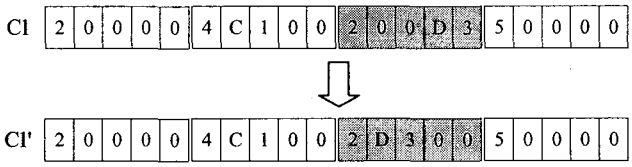

그림 6. 유전인자의 재정렬
Fig. 6 Realign of genes.

염색체의 표현 방법에서 첫 유전인자를 제외하면 나머지 유전인자들은 dogleg된 열과 트랙을 나타내는 2개의 쌍으로 이루어져있다.

만일 교배로 생성된 자손이 dogleg된 열과 트랙 중 하나의 정보만을 가지고 있을 경우 그 유전인자의 사용 여부를 임의로 결정하고, 만일 해당 유전인자를 사용한다면 나머지 유전인자는 임의의 값을 할당하고, 그렇지 않을 경우 '0'을 할당한다(그림 7).

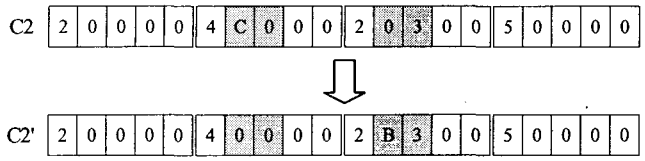

그림 7. 유전인자의 재할당
Fig. 7 Reassign of genes.

3. 돌연변이(Mutation)

유전자 알고리즘에서 각 세대의 모집단은 진화를 진행하면서 얻고자 하는 해에 가까운 염색체들로 구성된 모집단으로 수렴하게 되지만, 그 결과가 최적해가 아닌 지역해로 수렴할 수도 있으며, 이러한 지역해로의 수렴을 막기 위하여 돌연변이 연산을 수행하게 된다.

본 논문에서 사용한 돌연변이 연산 방법은 염색체의 각 유전인자를 돌연변이율 P_m에 의해서 dogleg을 사용할 열과 사용 트랙을 변경시키는 방법으로 사용하였다.(그림 8).

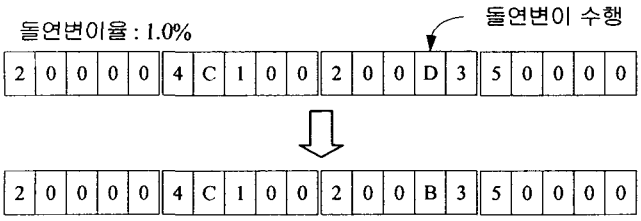

그림 8. 돌연변이 연산
Fig. 8 Mutation operation.

4. 평가함수(Evaluation Function)

유전자 알고리즘이 진행되는 동안 현재 모집단의 개체들은 특정 평가함수에 의해 평가된다. 디테일드 라우팅 문제에서 최종 목적은 라우팅에 사용된 트랙의 개수를 줄이는 것이기 때문에, 디테일드 라우팅 유전자 알고리즘의 평가함수는 모든 네트를 트랙에 할당한 후, 오버랩된 길이와 사용된 트랙의 개수를 사용하여 계산된다.

$$F = \frac{1}{\alpha \cdot O_v + \beta \cdot N_t} \quad (1)$$

여기서, O_v는 오버랩된 네트의 길이를 나타내며, N_t는 사용된 트랙의 개수를 나타낸다. 또한 α와 β는 페널티(penalty) 비율로써 α=[0,1], β=1-α의 값을 갖는다.

5. 간격 그래프(Interval Graph)

디테일드 라우팅에서 사용가능한 최소 트랙수는 다음과 같은 interval 그래프를 사용하여 구할 수 있다.

그림 9에서 (a)위치에서 최대로 5개의 네트가 겹치게 된다. 이것은 적어도 5개의 트랙이 필요하다는 것을 의미한다.

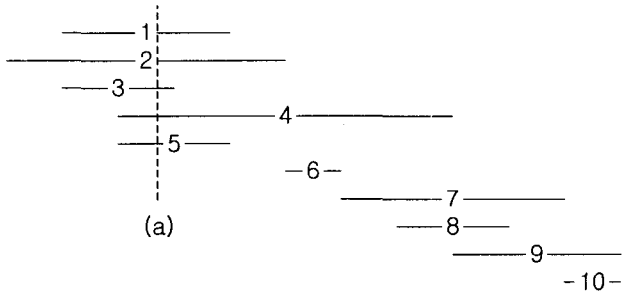

그림 9. 간격 그래프
Fig. 9 Interval graph.

6. 디테일드 라우팅 유전자 알고리즘

알고리즘 A는 본 논문에서 제안한 디테일드 라우팅 유전자 알고리즘을 나타낸다.

알고리즘 A. 디테일드 라우팅 유전자 알고리즘

단계 0 : 파라미터들의 설정
pop_size : 객체들의 수
P_m : 돌연변이율
max_gen : 최대 생성 횟수
S_t, E_t, N_t=S_t, : 최소, 최대, 현재 트랙의 수
N_c : 사용가능한 열의 개수
num_dogleg : 최대 가능한 dogleg 횟수
α, β, γ (γ>[1,2])

단계 1 : 초기 모집단의 생성
pop_size 만큼의 각기 다른 객체 S_i(i=1, …, *pop_size*)를 임의로 생성한다. 각 유전인자는 N_c와 N_t 범위내에서 선택.

단계 2 : 평가함수를 이용하여 모든 객체들의 적합도를 계산한다.

단계 3 : 교배
모집단으로부터 두 개의 부모 염색체 *P1*, *P2*를 임의로 선택하고, 2-점 교배 연산을 수행하여 새로운 자식 염색체 *C1*, *C2*를 생성한다.

단계 4 : 돌연변이
염색체의 각 위치를 임의로 생성한 $\varepsilon \in (0,1]$과 돌연변이율 P_m을 비교하여 만일 $\varepsilon < P_m$이면 돌연변이 연산을 수행한다.

단계 5 : 만일 생성된 자식 염색체 *C1* 또는 *C2*가 모집단 내의 어느 객체와 같다면, 같은 객체가 없는 새로운 객체가 될 때까지 돌연변이 연산을 수

행한다.

단계 6 : 새로운 세대의 구성
모집단 내의 가장 낮은 적합도를 갖는 두 개의 염색체를 삭제하고 생성된 자식 염색체 *C1*과 *C2*를 모집단 내에 추가한다.

단계 7 : 새로운 염색체의 생성 횟수가 max_gen이 될 때까지 단계 3부터 6까지를 반복한다.

단계 8 : N_t의 값이 E_t가 될 때까지 다음을 수행하고,
$N_t = N_t + 1$
$max\_gen = max\_gen * \gamma$
단계 1에서 8을 반복한다.

단계 9 : 최적해는 모집단 내에서 가장 높은 적합도를 갖는 염색체이다.

그림 10은 디테일드 라우팅 유전자 알고리즘의 흐름도를 나타낸다.

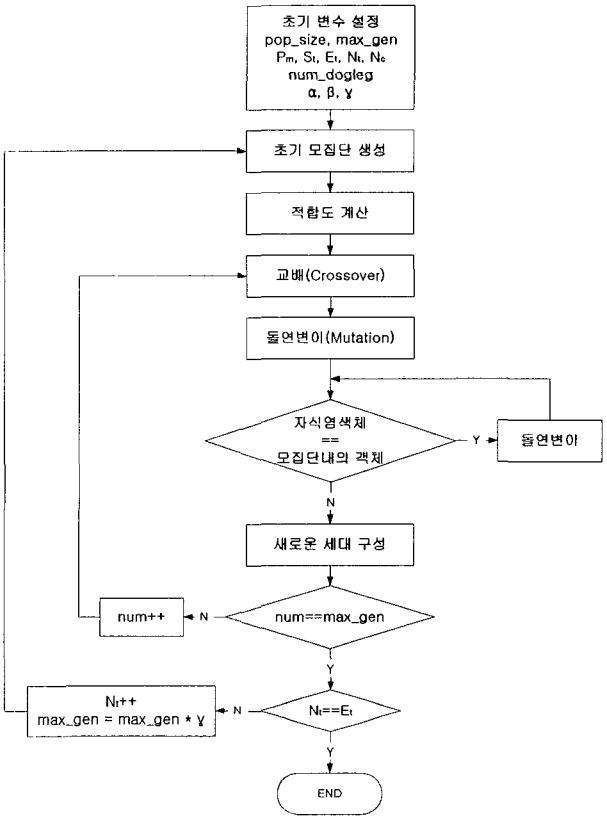

그림 10. 알고리즘 흐름도
Fig. 10 Flow chart for algorithm

Ⅳ. 시뮬레이션

본 논문에서 제안한 디테일드 라우팅 유전자 알고리즘의 시뮬레이션을 위하여 임의의 네트리스트 데이터를 생성하고, 생성된 데이터에 greedy 채널 라우팅 알고리즘과 디테일드 라우팅 유전자 알고리즘을 적용하여 그 결과를 분석하였다. 또한 벤치마크 회로인 Deutsch's difficult problem 회로에 대하여 동일 알고리즘을 적용하여 결과를 비교하였다.

본 시뮬레이션에서는 디테일드 라우팅 유전자 알고리즘의 돌연변이율 P_m=1.0, 모집단내의 개체의 수 *pop_size*=20, 최대 수행 횟수 *max_gen*=100, 최대 dogleg 횟수 *num_dogleg*=5, α=0.8, β=0.2, γ=1.2를 사용하였다. 또한 시뮬레이션 환경은 AMD Athlon 1GHz, 512MB RAM에서 Visual C++ 6.0을 사용하여 수행하였다.

그림 11은 임의로 생성된 네트리스트 데이터에 greedy 채널 라우팅 알고리즘과 디테일드 라우팅 유전자 알고리즘을 적용한 결과를 가지고 각 알고리즘을 비교한 것을 보이고 있다. 자세한 수치 결과는 Appendix의 표 2에 보였다.

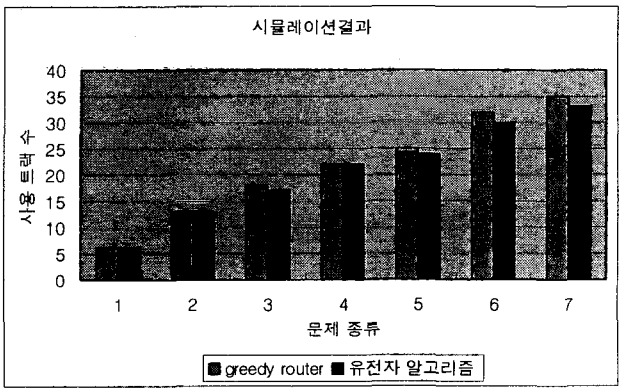

그림 11. 임의 생성 네트리스트 데이터 시뮬레이션 결과
Fig. 11 Simulation result for random netlist data.

또한, 그림 12는 벤치마크 회로인 Deutsch's difficult problem 회로에 대하여 동일 알고리즘을 적용한 결과를 보이고 있다. 자세한 수치 결과는 Appendix의 표 3에 보였다.

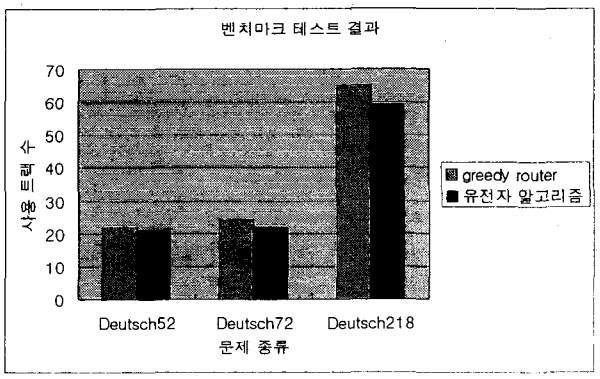

그림 12. 벤치마크 테스트 결과
Fig. 12. Benchmark test result.

표 1은 Deutsch's difficult problem 벤치마크 회로에서 greedy 채널 라우팅 알고리즘에 대한 디테일드 라우팅 유전자 알고리즘의 개선도를 나타낸 것이다. 개선도를 얻기 위해 사용된 식은 다음과 같다.

$$개선도(\%) = (Greedy-GA) / Greedy * 100 \quad (2)$$

표 1. GA의 개선도

Table 1. Improvement by percentage of GA

벤치 마크	네트 개수	개선도
Deutsch52	52	4.55
Deutsch72	72	8.33
Deutsch218	218	9.23

그림 13은 표1의 문제중 8개의 네트를 가지는 임의 생성 데이터에 대하여 디테일드 라우팅 유전자 알고리즘을 적용한 결과를 보이고 있다.

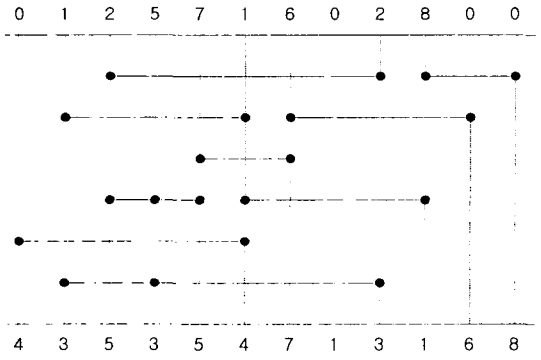

그림 13. 디테일드 라우팅 유전자 알고리즘 결과

Fig. 13 Result of detailed routing genetic algorithm.

두 알고리즘의 시뮬레이션 결과 디테일드 라우팅 유전자 알고리즘이 greedy 채널 라우팅 알고리즘보다 약 5%-10%정도로 더 좋은 결과를 얻는 것을 알 수 있다. 시뮬레이션 결과 적은 개선도를 보이지만, greedy 채널 라우팅 알고리즘이 매우 우수한 알고리즘이므로 만족할 만한 결과라 볼 수 있다.

Greedy 채널 라우팅 알고리즘은 column by column으로 진행을 하면서, 해당 column에서 최적의 라우팅을 탐색하는 방법으로 매우 우수한 발견적 탐색 방법의 알고리즘이나 지역해에 빠지기가 쉽다. 또한 WEAVER등의 인공지능 기법의 알고리즘들은 과도한 수행시간이 요구되므로 문제가 커지면 실현성이 없어진다는 단점이 있다. 반면에 유전자 알고리즘은 교배와 돌연변이 연산을 사용한 이웃해로의 이동범위 광역화에 의한 효과적인 해공간 탐색으로, 인공지능 기법의 알고리즘에 비해 비교적 빠른 시간 내에 해를 찾게 되며, greedy 채널 라우팅 알고리즘에 비해 더 좋은 해를 찾게 된다.

V. 결론

본 논문에서는 VLSI 설계 과정 중 디테일드 라우팅 문제에 대하여 유전자 알고리즘을 제안하였으며, 제안한 방식을 greedy 채널 라우팅 알고리즘과 비교, 분석하였다.

제안한 디테일드 라우팅 유전자 알고리즘과 greedy 채널 라우팅 알고리즘을 임의로 생성한 회로와 Deutsch's difficult problem 벤치마크 회로에 적용하여 비교, 분석한 결과 제안한 디테일드 라우팅 유전자 알고리즘이 greedy 채널 라우팅 알고리즘보다 더 효과적으로 최적해에 근접하는 것을 알 수 있었다.

앞으로 다층(multi-layer) 채널 라우터와 스위치박스 라우터를 포함한 디테일드 라우팅 문제에 대한 좀더 효과적인 유전자 알고리즘에 대한 연구가 필요하다고 생각된다.

Appendix

표 2. 임의 생성 네트리스트 데이터 테스트 결과

Table 2. Test results for random netlist data.

문제 종류	네트 개수	사용 트랙 수	
		greedy router	유전자 알고리즘
1	8	6	6
2	20	13	13
3	40	18	17
4	60	22	22
5	80	25	23
6	100	32	30
7	120	35	33

표 3. 벤치마크 테스트 결과

Table 3. Benchmark test results.

벤치 마크	네트 개수	사용 트랙 수	
		greedy router	유전자 알고리즘
Deutsch72	52	22	21
Deutsch72	72	24	22
Deutsch218	218	65	59

접수일자 : 2002. 6. 11 수정완료 : 2002. 7. 20

참고문헌

[1] S. M. Sait, H. Youssef, *VLSI Physical Design Automation Theory and Practice*, World Scientific Publishing, 2001.

[2] Naveed A. Sherwani, *Algorithms for VLSI Physical Design Automation. 3rd Edition*, Kluwer Academic Publishers, 2001.

[3] S. M. Sait, H. Youssef, *Iterative Computer Algorithms with Applications in Engineering*, Computer Society, 1999.

[4] A.Hashimoto and J.Stevens. Wire routing by optimizing channel assignment within large apertures. *Proceedings of 8th Design Automation Conference*, pages 155-169, 1971.

[5] D.N.Deutch. A dogleg channel router. *Proceedings of 13th Design Automation Conference*, pages 425-433, 1976.

[6] R.L.Rivest and C.M.Fiduccia. A greedy channel router. *Proceedings of 19th Design Automation Conference*, pages 418-424, 1982.

[7] T.Ho, S.S.Iyengar, and S.Zheng. A general greedy channel routing algorithm. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 10(2):204-211, February 1991.

[8] T.Yoshimura and E.S.Kuh. Efficient algorithms for channel routing. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, CAD-1:25-35, Jan 1982.

[9] U.Yoeli. A robust channel router. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 10(2):211-219, February 1991.

[10] G.Vijayan, H.H.Chen, and C.K.Wong. On VHV routing in channels with irregular boundaries. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 8(2):146-152, February 1989.

송호정 (Ho-Jeong Song)
準會員
1994년 배재대학교 물리학과 이학학사
1996년 청주대학교 전자공학과 공학석사
2001년 충북대학교 컴퓨터공학과 박사수료
관심분야 : VLSI 설계, High-level Synthesis

송기용 (Gi-Youn Song)
正會員
1974~80년 서울대, 동대학원(전자공학)
1995년 Univ. of Southwestern Louisiana 공학박사
1983년~현재 충북대학교 공과대학 컴퓨터공학과 재직중
관심분야 : 컴퓨터구조, VLSI 설계등