論文 89-26-6-16

IC 밀집화를 위한 최적기술에 대한 연구

(A Study on the Optimization Technique for IC Compaction Problem)

李 天 熙*

(Cheon Hee Yi)

要 約

본 논문은 사용자가 정의한 stick 도형으로부터 혼합된 정수 선형프로그래밍 문제를 공식화시키는 새로운 mask compaction 방법을 제시하였는데 이 혼합된 정수 프로그램을 풀므로써 밀집화되고 디자인 rule에 맞는 layout을 얻을 수 있다. 또한 constraint graph에서 최장경로 문제를 풀 수 있는 새로운 효율적인 알고리듬도 기술하였다.

Abstract

This paper describes a new method of mask compaction to formulate a mixed integer linear programming problem from a user defined stick diagram. By solving this mixed integer program, a compacted and design rule violation free layout is obtained. Also, a new efficient algorithm is given which solves the longest path problem in the constraint graph.

I. 서 론

콤팩션(compaction)은 IC 설계에서 초기 설계도나 심볼릭(symbolic) 도표를 밀집화 시키는 CAD tool로서 1970년에 DA Workshop[1]에서 처음으로 사용되었으며 1977년에 만들어진 FLOSS[2]와 SLIP[3]은 손으로 하는 것보다 설계 주기를 크게 단축시켰다.

*正會員, 淸州大學校 電子工學科
(Dept. of Elec. Eng., Chongju Univ.)

接受日字 : 1989年 1月 19日
(※본 논문은 87년도 과학재단 일반연구 지원으로 이루어진 것임.)

J.D.Williams가 1977년에 stick도형의 개념을 소개한 이래 stick도형을 geometrical layout으로 자동변환시키기 위한 연구가 활발히 진행되었다. 1978년에 Williams가 STICKS compiler[4]를 만들었고 1979년에는 M.Hsueh가 CABBAGE 시스템을[5], 1980년에는 A.E.Dunlop가 SLIM 시스템을[6] 만들었고 R.C.Mosteller가 REST 시스템을 고안 하였으며 1981년에는 N.Weste가 MULGA 시스템을 개발하였다. 1983년에는 Y.Z.Liao와 C.K.Wong이[7] IC 디자인의 symbolic layout을 compaction하는 그래프 알고리듬을 제안하였고 G.Kedem과 H.Watanabe는[8] compaction에 대한 그래프 최적화 기법을 발표하였다. 또한 W.L.Schiele

도[9] 전체 배선길이를 줄일 수 있는 최적화 설계에 대한 알고리듬을 발표하였으며 W.H.Wolf도[10] compaction 알고리듬보다 pitch를 최소화 할 수 있는 면적최소화의 heuristic 방법을 발표하였다.

C.Kingsley가[11] 1984년에 계층적으로 디자인되는 모든 칩을 compaction하는데 실제적이며 셀 설계시스템과 칩 어셈블리 tool로써 사용할 수 있는 compaction을 개발하였다. 1985년에는 M.Ishikawa, S.Goto 등이[12] 빌딩블럭 LSI를 위한 설계면적을 최소화하는 compaction 방법을 발표하였고 1986년에는 H.Shin, C. H.Sequin 등이[13] 최적위치에 소자를 배치하기 위하여 필요한 유연성을 만들어 줌으로써 compaction 단계사이에 배선을 할시에 *jog*를 삽입하는 2차원의 설계기법을 발표하였다.

현재까지의 시스템은 두개의 1차원 문제로서 compaction을 수행하기 때문에 수직과 수평의 compaction이 교대로 이루어 진다. 따라서 한 방향의 compaction이 다른 방향에서 다음의 compaction을 방해하는 수도 있다. 또 알고리듬이 번갈아 수행되기 때문에 해결 수순이 임의로 되고 앞에서 결정이 나중 해결의 가능성을 제한하기도 한다. 그러므로 주어진 모델에서 정의된 제한된 영역에서 최적설계가 만들어 질수 없는 경우도 생긴다. 또 다른 문제점은 compaction 작업이 단지 국부정보만으로 수행되며(STICKS compiler, SLIP) SLIM과 CABBAGE에서는 compaction이 그래프 모델의 최장경로를 사용하여 얻어진 전체적인 관점에서 유도되지만 compaction작업 자신은 기본적으로 국부작업이다.

Compaction 작업으로 만든 최종 layout은 제어할 수 있는 요소가 적으며 수행된 작업이 목표 지향적이 아니다. 목표 지향적인 compaction에서 시스템에 주어진 목표는 compaction 진행과 최종 셀의 모양을 유도하는 것이다. 각 작업은 전체의 layout 면적을 감소시키는데 아무렇게나 적용되었다. 그러나 만일 회로가 큰 시스템의 sub-block이라면 비록 layout의 면적이 최소가 되지 않더라도 좋은 모양이 될 수도 있다. 어떤 모양내에 sub-block을 설계함으로써 sub-block사이에 생길 수 있는 못쓰는 공간을 줄일 수 있으며 전체 시스템을 위한 더 작은 layout을 만들 수 있다. 또 고려할 점은 연결점의 위치이다. 현존하는 시스템은 단순히 이 요소을 고려함이 없이 설계면적만을 최소화하려고 노력하였다.

계층적인 디자인 방법에서 layout generator를 이용하기 위해서 시스템은 목표를 구체화 시켜야 한다. 시스템은 배치와 연결정보로 부터 각 sub-block을 위한 목표를 조정하여야 한다.

본 연구의 연구목표는 앞에서 설명한 기존시스템의 결점을 일반적으로 극복할 수 있는 시스템의 개요를 구성하는 데에 있다.

Ⅱ. Layout 생성을 위한 수학적 최적화

1. 선형 프로그래밍

디자인 rule은 두 회로요소와 내부 연결배선의 최소폭과 다른 layer의 최소 겹침사이에 필요한 최소공간을 고려하여 정의하였다. 그러므로 부등식으로 디자인 rule을 표현하는 것이 가능하다. 간략히 하기위해 그림(1)에서 두개의 이웃하는 심볼에 알맞는 layout의 조건을 다음의 부등식으로 표현하였다.

$$(X2-W2)-(X1+W1)>H12 \qquad (1)$$

상수를 오른쪽으로 옮기면

$$X2-X1>H12+W1+W2 \qquad (2)$$

모든 상수를 d12 표시하면

$$X2-X1>d12 \qquad (3)$$

유사한 부등식이 compaction 진행동안 모든 회로요소에 대해서 명확히 표현될 수 있다. 이 제한조건하에서 최적화된 목적함수는 layout의 면적이 되며 $A=W\cdot H$이다. H는 가장 높은 segment의 좌표이고 W는 layout내에 가장 오른쪽의 좌표이다. Otten이[14] 제안한 근사값 W+H를 사용하여 선형 프로그램 문제를 명확히 표현할 수 있다. W+H를 최소화 하면

$$Ax>b, \qquad x>0 \qquad (4)$$

A는 (3)식과 같은 부등식의 왼편에서 계산된 matrix의 계수이며 x는 값이 layout을 표시하는 벡터이고 b는 (3)식과 같은 부등식의 오른편에서 계산된 벡터이다. 이 선형 프로그래밍 문제는 이 제한조건들하에서 최적 layout을 만들어 낸다. 그러나 선형 프

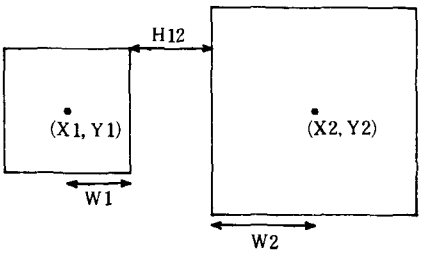

그림 1. 심볼사이의 제한조건
Fig. 1. Constrains between symbols.

로그램은 일반적으로 compaction 작업의 2D와 관련된 다음의 두 경우를 취급하기에 충분하지 않다.

1) 그림 2에서 심볼 B는 심볼 A에서 볼 때 수평이거나 또는 수직방향으로 움직일 수 있다. 주위의 조건에 따라서 두 움직임의 하나는 다른 것보다 설계의 전체적인 compaction에는 더 좋을 수가 있다. 이 경우에 디자인 rule 요구는 다음과 같이 표현된다.

$$Xb-Xa>Hab+Wa+Wb \quad \text{수평 제한조건} \quad (5)$$
$$Ya-Yb>Vab+Wa+Wb \quad \text{수직 제한조건} \quad (6)$$

필요조건은 위의 두 조건중에 최소 한개가 만족되어야 한다. 그러나 이것은 선형이 아니다. B(xb, yb)를 중심점으로 하는 지정된 영역은 그림 2에서 빗줄친 부분으로써 모양이 convex가 아니다. 만일 위의 조건이 정규의 선형 제한조건으로 처리된다면 심볼 B를 위한 합법적인 영역은 그림 3에 표시된 것과 같다. 이것은 지나친 제한조건이고 compaction을 위하여 적당하지 않다. 기껏해야 부등식의 계산시간에서 심볼 B의 이동방향을 선택해야 하며 이것은 단지 두 조건중에 하나가 만들어진 것이다. 그림 4에서 수직의 제한조건이 만들어 졌고 심볼 B는 단지 수평방향에서 움직일 수 있다.

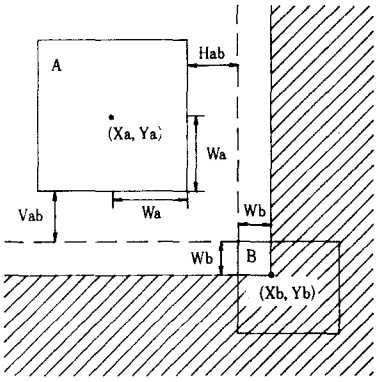

그림 2. Compaction 작업의 비선형성
Fig. 2. Non-limearity of compaction operation.

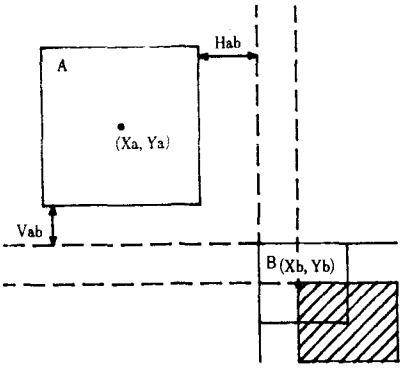

그림 3. 선형화된 겹침 제한조건
Fig. 3. Linearized but over-constrained.

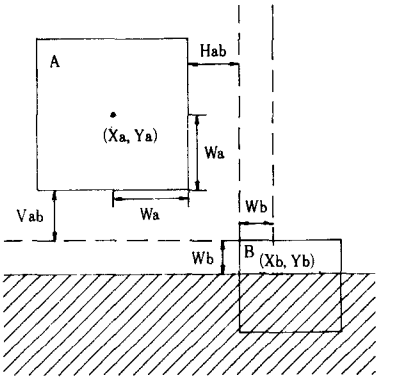

그림 4. 수평방향의 선택
Fig. 4. Only horizontal direction is chosen.

선형성이 유지되었지만 compaction 조작은 아직 문제계산 시간에서 만들어진 임의의 결정 때문에 크게 제한된다.

2) 두번째 문제는 수평과 수직방향이 교대로 고려될 때 일어나며 그림 5에 묘사된 것과 같다.

심볼 C를 위한 수평의 제한조건은 심볼 A와 B까지 심볼 C의 연관된 수직위치에 따라 다르다. 심볼 A 또는 B로 부터 제한조건의 하나가 적용되며 그 조건은 다음과 같이 표현된다. 만일

$$yc-yb>vbc+Wb+Wc$$

그러면

$$xc-xa>Hac+Wa+Wc \quad (7)$$

아니면

$$xc-xb>Hbc+Wb+Wc$$

위의 부등식의 set는 선형이 아니며 심볼 C의 중

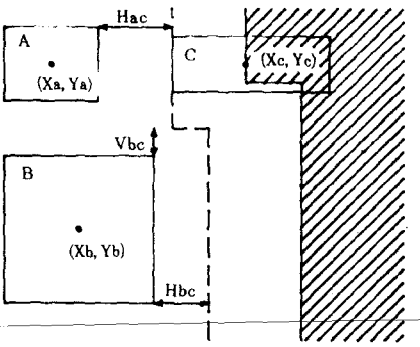

그림 5. 조건부 제한조건
Fig. 5. Conditional constraints.

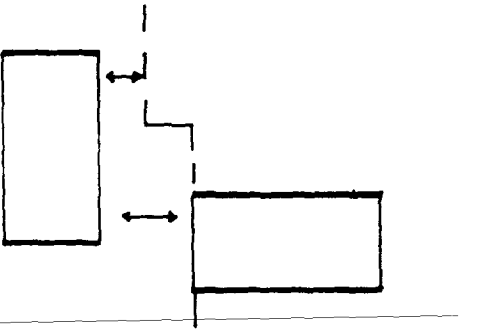

그림 7. 수평공간 요구의 변화
Fig. 7. Varying horizontal space requirement.

심점을 위한 지정된 영역은 convex가 아니다. 이것을 그림 5에 줄친 면적으로 나타냈다. 만일 이 두개의 수평 제한조건이 수직관계를 고려함이 없이 번갈아 일어난다면 조건은 선형이 되며 그림 6과 같다. 이것은 compaction을 위하여 적당하지 않으며 만일 같은 방향으로 부터 두개 이상의 제한조건이 있다면 가장 심한것은 모든 위치에 대한 제한조건으로써 실제 디자인 rule 요구를 고려함이 없이 항상 일어난다. 비슷한 상황이 그림 7에 보인것처럼 심볼사이에 필요한 수평공간이 심볼의 연관된 수직의 위치에 따라서 변할때 두 심볼사이에서 일어난다.

Layout 생성과 compaction을 위한 선형 프로그래밍을 사용하기 위하여 문제계산 시간에 compaction의 방향에 대하여 어떤 결정을 내려야 한다. 이것은 compaction 작업에서 일반성을 잃게되는 원인이 되며 못쓰게된 면적은 위의 두번째 경우에서 처럼 어느 경우에는 불가피하다.

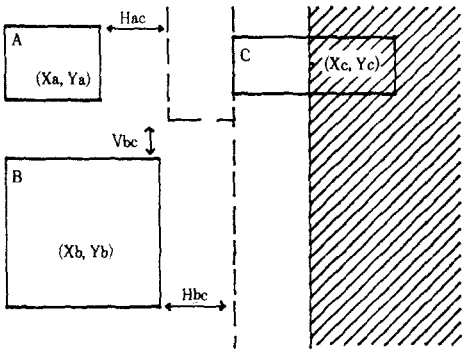

그림 6. 선형화된 조건부 제한조건
Fig. 6. Linearized conditional constraints.

2. 혼합된 정수 프로그래밍

위 문제들을 처리하기 위하여 혼합된 정수 프로그래밍의 기술을[7] 이용할 수 있다. 여기에 나타낸 공식은 compaction 방법의 기초를 이루는 모델을 제공한다. $f(x)$를 최소화하는 문제를 고려해 보면

$$x \in s = \{x \mid Ax > b, x > 0\} \tag{8}$$

이며 $g(x) > 0$이거나 $h(x) > 0$ 또는 양쪽 다이다. 이 문제는 이분법이기 때문에 선형이 아니다. 그러나 이 분법은 kg, kh가 g와 h상에 한정된 하한경계이고 δ가 0, 1인 정수변수일 때 다음식이 성립한다.

$$\begin{aligned} &g(x) > \delta Kg \\ &h(x) > (1-\delta) Kh \\ &\delta = 0, 1 \end{aligned} \tag{9}$$

이 방법을 사용하면 앞절의 첫번째 경우는 Kh, Kv가 아래 식의 왼쪽편의 하한경계인 곳에서 다음과 같은 공식으로 나타내어 질 수 있다.

$$\begin{aligned} &xb - xa - (Hab + Wa + Wb) > \delta Kh \\ &ya - yb - (Vab + Wa + Wb) > (1-\delta) Kv \\ &\delta = 0, 1 \end{aligned} \tag{10}$$

실제로 하한경계를 위해 충분히 작은 수를 사용할 수 있다. 두번째 경우는 (11)식의 조건표현이 (12)식과 등가이기 때문에 같은 방법으로 공식화 할 수 있다.

만일

$$g(x) > 0 \text{이면 } h1(x) > 0 \text{이고 아니면 } h2(x) > 0 \tag{11}$$

(11)식은 다음식처럼 표시할 수 있다.

$$g(x)>0, h1(x)>0 \text{ 또는 } g(x)<0, h2(x)>0 \quad (12)$$

이 표현은 Kg, Kh1, Kh2가 g, h1, h2의 한정된 하한경계이며 Cg가 g의 상한경계일 때 앞의 경우와 비슷하며 다음식과 등가이다.

$$\begin{aligned} &g(x)>\delta Kg, h1(x)>\delta Kh1 \\ &g(x)<(1-\delta)Cg, h2(x)>(1-\delta)Kh2 \quad (13) \\ &\delta=0, 1 \end{aligned}$$

계산목적을 위하여 $g(x)<(1-\delta)Cg$보다 $g(x)<(1-\delta)Cg$표현을 결과에 영향을 미침이 없이 사용할 수 있다. 두번째 경우에서 디자인 rule에 맞는 조건표현은 K가 아래의 세표현의 하한경계 모두가 K와 같거나 크며 C가 충분히 큰 수인 경우에 다음식으로 표현된다.

$$\begin{aligned} &yc-yb-(Vbc+hb+hc)>\delta K \\ &xc-xa-(Hac+Wa+Wc)>\delta K \\ &yc-yb-(Vbc+hb+hc)<(1-\delta)C \quad (14) \\ &xc-xb-(Hbc+Wb+Wc)>(1-\delta)K \\ &\delta=0, 1 \end{aligned}$$

이 공식의 이점은 계산이 두개의 분리된 일차원 문제라기 보다는 단일 이차원의 문제로서 수행된 것이다. 또한 국부적인 모든 제한조건이 전체적인 최적조건을 완성시키기 위하여 교대로 고려되었다. 그러므로 모든 국부적인 움직임이 전체적인 목표의 내용으로 수행되었다. 단점은 혼합된 정수 프로그래밍이 선형 프로그래밍보다 풀기가 더 어렵다는 것이다.

선형 프로그래밍을 위한 단순한 방법에 필적할 만한 정수 프로그래밍을 위한 효과적인 알고리듬이 없으며 게다가 0-1 변수의 각 삽입은 탐색공간을 2배로 한다.

Ⅲ. 그래프 최적화

1. 그래프내의 arc와 디자인 rule

앞절에서 arc를 계산하였으며 선형부등식에 일치하는 arc들이 연결요구를 표현하기 위하여도 사용된다. 그래프에는 2종류의 arc가 있는데 그 중 하나는 layout내에 요소사이에 최소거리 요구와 일치하며 분리 arc라 부른다. 다른 arc는 심볼과 배선사이에 연결을 표현하며 연결 arc라 부른다. 전자의 arc는 다시 단일 arc와 이중 arc로 갈라진다.

그래프에서 분리 arc는 두개의 요소사이에 최소의 거리와 일치한다. Arc의 머리노드와 꼬리노드는 포함된 두 요소를 표현하며 거리는 arc의 무게로서 표현되었다. 최소거리 요구는 arc가 그룹순서 매김에 따라 전임자 그룹내의 노드로 부터 후임자 그룹내의 노드까지 지적하도록 정의한다. 그래프가 디자인 rule 를 따르도록 만들고 arc에 의해서 표현된 모든 요구가 만족했다면 layout 결과는 디자인 rule을 충족시킨다. 그래프 생성 알고리듬은 심볼들이 너무 근접해 있으므로써 디자인 rule을 위반할 수도 있는 심볼쌍 사이에 arc를 만들어야 한다. 만일 두 심볼이 단일 직선의 배선으로 직접 연결되었다면 거리 요구는 그림 8 에서 보인 것처럼 한 방향내에 있다. 이것은 디자인 rule로 부터 생성된 간단한 arc의 예가 된다.

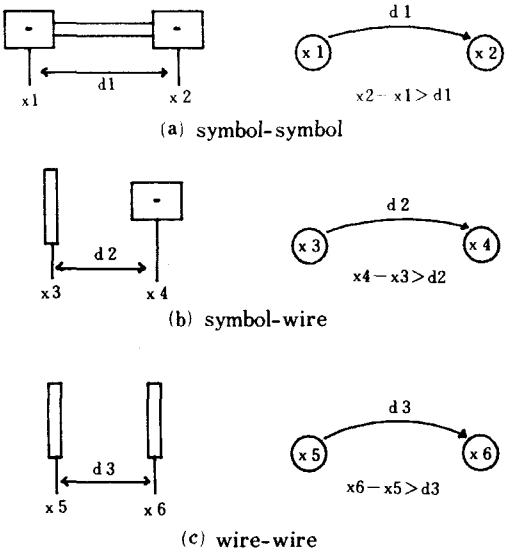

그림 8. 간단한 arc의 예
Fig. 8. The example of simple arcs.

간단한 arc는 또한 배선과 심볼 (그림 8 의(b))사이와 배선사이에 (그림 8 의 (c)) 생성된다. 두 심볼의 단일 직선의 배선에 의해서 연결되지 않았으나 서로서로 가까이 있을때 그림 9 의 (a)에 보인 것처럼 두개의 제한조건이 있는데 하나는 수직방향내에 있고 다른 하나는 수평방향내에 있다. Arc의 쌍은 하나는 수직의 그래프내에 다른 하나는 수평의 그래프내에 구성되어야 한다.

Arc의 쌍을 이중 arc라 부른다. Compact된 layout의 결과가 디자인 rule을 충족시키기 위해서는 두 제한조건중의 하나가 만족되어야 한다. 이것은 사용된 매트릭스 (즉 맨하탄 거리)와 모든 심볼을 최소로 둘러쌓은 구형에 의해서 표현되었기 때문이다. 이중의 arc와 일치하는 비슷한 제한조건이 배선부분과

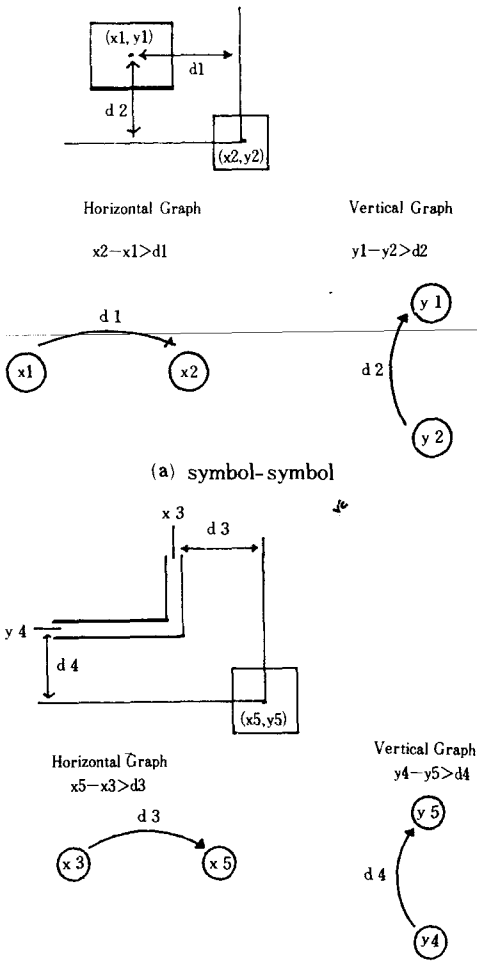

그림 9. 이중 arc사이의 연결
Fig. 9. Connection between dual arcs.

심볼사이에 그리고 그림 9의 (c)에 보인 것처럼 두개의 배선 segment사이에 생성된 배선 segment 가 단지 한 방향으로 좌표변수를 갖기 때문에 단말 요소의 좌표변수는 다른 방향으로 사용하여야 한다.

2. 심볼과 배선사이에 연결제한조건

Compaction 과정에서 심볼과 심볼의 연결배선 및 연결된 배선사이의 연결이 보존되어야 한다. 제한조건의 쌍은 단일 연결요구을 표현하기 위하여 사용되었는데 이 방법을 사용하고 또한 심볼의 연결노드가 충분히 넓은 경우에 배선과 연결된 심볼이 서로서로에 연관되게 이동시킬 수 있다. 그림10에서 보인 예를 사용하여 배선과 심볼사이에 연결요구의 표현을 설명하면 다음과 같다. x1, x2, x3를 배선과 심볼 및 심볼의 연결노드 각각의 중심좌표라하고 Wn, W1을 심볼과 배선의 연결노드의 넓이라 하자. 이제 d=(Wn−W1)/2라 정의하면 연결요구가 |x1−x3|<d이며 다음식과 등가이다.

$$x1-x3<d$$
$$x1-x3>-d \qquad (15)$$

Arc의 무게가 최소거리를 표시하는 관례에 의하면 위식을 다음과 같이 고쳐 쓸 수 있다.

$$x3-x1>-d$$
$$x1-x3>-d \qquad (16)$$

만일 노드 x3를 연결한 것의 중심이 심볼 x2의 중심과 동시에 동일공간을 차지한다면 위 부등식의 쌍은 조건이 필요하다.

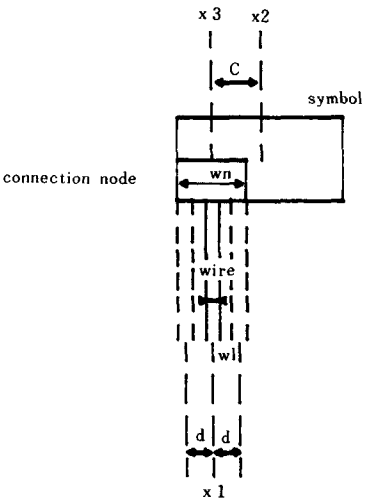

그림10. 제한조건의 연결
Fig. 10. Connection constraints.

배선사이에 연결요구는 약간 다르다. 그림11은 수평배선 L2에 연결된 수직배선 L1을 보여준다. 수평배선이 심볼 A와 B에 연결되었다고 수직의 배선 L1의 중심좌표는 연결을 유지하는 선분 L2의 단말요소 사이어야 한다. 그러므로 L1과 L2사이에 연결요구는 다음의 두 부등식으로 표현된다.

$$xl1 - xa > da$$
$$xb - xl1 > db \qquad (17)$$

여기서 da와 db는 각기 심볼 A와 배선 L1사이와 두 선분사이에 연결을 견고히 할 뿐만 아니라 또한 수직배선과 두개의 단말의 심볼을 취급한다. 연결 arc가 그래프의 acyclic 특성을 파괴하기 때문에 compaction 알고리듬의 최장길이 부분에서 특별히 취급하여야 한다.

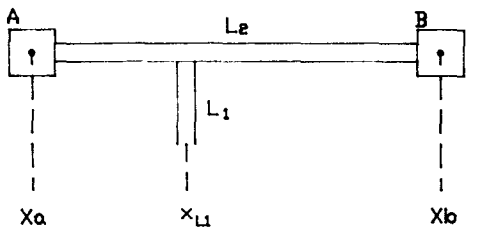

그림11. 배선사이의 연결
Fig. 11. Connection between wires.

3. Constraint graph

Compaction의 주요접근 방법에는 constraint graph와[2,7,8,19] virtual grid 방법등이[20,21] 있는데 그 중에서 constraint graph 방식이 가장 많이 사용된다. Constraint graph 방식은 virtual grid 방식보다 속도는 늦으나 면적을 작게 만들 수 있다. 앞에서 설명한 것처럼 constraint graph 모델에서 nobe는 설계 요소(mark layer 또는 심볼)를 표시하며 arc는 이 요소 사이에 거리제한을 표시한다. 따라서 compaction 과정에서 심볼과 이들의 연결배선 및 연결된 배선사이에 연결을 유지하여야 한다.

최장경로가 발견되면 critical node와 critical arc set이[16] 결정되며 최장경로와 연관되지 않은 모든 배선의 길이를 최소화하여야 한다. Critical node와 arc는 sink node로 부터 breadth first search에 의하여 찾을 수 있으며 배선길이를 최소화할 때 critical no-

```
Algorithm for longest path
Definitions:
  V:node set
  A:arc set
  var S1, S2:node set
  v:node name
  s:source node
  t:drain node
  xi:constant for vi ∈Vc
  vi:vi∈Vc(critical nodes)
  Arc(vi):all ancestor of vi
  bij:the weight of arc aij
  len(a):minimum length of arc a
  loc(v):physical x or y location of node v
  l0:length of the longest path
  E:set of arcs in a digraph
PROCEDURE Longest path(l0)
  for each vi∈V do xi←−∞;
  x0←0;S1←s; {initialization complete}
  begin
    n:=|A|;
    l0:=0;
    for all nodes v in a digraph do
        loc(v):=0
    end;
    call update(s, n, l0);
  while S1≠0 do begin
        S2←0;
          {do breadth first search}
        for each vi∈S1, vi∈S2 do begin
          if xi<xi+bij then begin
            xj←xi+bij;
            pj←Vi;
            S2←S2+vj;
          end
        end
          {check for positive cycles}
        for each vj∈S2 do begin
          for each va∈Anc(vj) do begin
            if v0=vj then
              positive cycle in ancestry path;
          end
        end
          S1←S2;
    end
  end
PROCEDURE update(vl, n, l0)
  (update node locations for longest path calculations)
  for all arcs a∈I (v) do
    if a∈I+(v) then
      (consider only nodes whose in-comming arcs were
       already marked)
      if mark (a)≠0 return;
    else(a∈I−(v))
      if mark (a)≠0 then
      begin
        n:=n−1
        mark (a):=0;
        find a terminal node u:a∈I+(u);
        if u=t then
          l0:=max{l0, loc(v)+len(e)};
          if n=0 return
        else
          loc(u):=max{loc(u), loc(v)+len(e)};
          call update(u, n, l0);
        end;
      end;
    end;
end;
```

그림12. 최장경로 알고리듬
Fig. 12. Longest path algorithm.

de가 아닌것 만을 옮길 수 있기 때문에 모든 critical node와 arc는 무시한다. 배선길이 최소화 알고리듬은[9,18] 여러가지 있으나 최근의 Hench Hegen이 만든 알고리듬은[16] simplex 방법을 기초로 하여 만들어 졌다.

Compaction에서 constraint graph 방식이 중요한 접근방식이며 constraint graph 방식에서는 최장경로를 발견하기 위한 그래프 이론적인 알고리듬이 중요한 부분을 차지하기 때문에 이를 계선하기 위한 연구들이 많이 이루어 졌으며 본 연구에서도 그림12와 같은 효율적인 알고리듬을 개발하였다.

최장경로 알고리듬으로 constraint graph를 풀므로써 칩면적을 작게 할 수 있으며 symbolic layout과 mask layout에서 배선길이를 고려한 최적배치도 이루어 진다. 그러나 positive cycle이 존재하면 최장경로 문제를 풀기가 어려우나 이러한 over-constrained compaction 문제를 freezing 방법[17], relaxation 방법[11], jog insertion 방법[15,22] 등으로 풀 수 있다. 따라서 positive cycle이 탐지되면 위의 방법으로 해결한 후에 다시 최장경로 알고리듬을 수행하게 된다.

Ⅳ. 결 론

현존하는 시스템의 결함을 극복하기 위하여 선형 프로그래밍과 혼합된 정수 프로그래밍을 layout 생성과 compaction에 적용하는 문제를 검토하였다. 여기에 묘사된 공식은 충분히 일반적이어서 모든 결함들이 혼합된 정수 프로그래밍의 framework내에 균일하게 처리된다. Sub-block들의 모양과 위상수학적인 배치가 주어졌다는 가정하에 절연된 영역을 결정하기 위하여 혼합된 정수 프로그래밍과 선형 프로그래밍이 사용되었다.

또한 compaction의 중요한 접근방법으로 constraint-graph가 사용되며 constraint 방식에서 최장경로를 발견하기 위한 그래프 이론적인 알고리듬이 중요한 부분을 차지하기 때문에 본 연구에서도 효율적인 최장경로 알고리듬을 개발하였다. 앞으로의 연구과제는 compaction 시스템이 계층적으로 이루어지도록 하여 본 연구에서 구상한 시스템을 효율적으로 구현시키도록 하며 silicon compiler에서도 사용할 수 있도록 하는 것이다.

參 考 文 獻

[1] Akers, S.B. Gyer, J.M: and Roberts, D.L. "IC Mask layout with a single conducting layer." *Proc of the 7th Design Automation Workshop* 88.7-16, 1970.

[2] Y.E. Cho, A.J. Korenjak. and D.E. Stockton. "FLOSS: an approach to automated layout for high-volume designs." *Proc 14th D.A. Conference*. pp. 138-141, 1977.

[3] A.E. Dunlop. "SLIP: sybolic layout of integrated circuits with compaction." CAD. vol. 10, no. 6. pp. 387-391, 1978.

[4] J.D.. Williams. "STICKS: a graphical compiler for high-level LSI design." AFIP Conference Proc. vol. 47. pp. 289-295, 1978.

[5] M. Hsueh, "Symbolic layout and compaction of integrated circuit," Memorandom No. UCB/ERL M 79/80, Electronics, Research Lab. University of California, Berkeley, 1979.

[6] A.E. Dunlop, "SLIM-The translation of symbolic layout into mask data," *Proc. of 17th DAC*, pp. 595-002, 1980.

[7] Y.Z. Liao, C.K. Wong, "An algorithm to compact a VLSI simbolic layout with mixed constraints." *Proc. 20th D.A. Conference*. pp. 107-112. 1983.

[8] G. Kedem, H. Wantanabe. "Graph-optimization techniques for IC layout and compaction," *Proc. 20th D.A. Conference*. pp. 113-120. 1983.

[9] W.L. Schiele. "Improved compaction by minimiged length of wires." Proc. 20th D.A. Conference. pp. 121-127. 1983.

[10] W.H. Woll. "Two-dimensional compaction strategies," *Proc. IEEE International Conference on CAD*. pp. 90-91, 1983.

[11] C. Kingsiey. "A Hierarchical. errortolerant compaction," *Proc. 21th D.A. Conference*. pp. 126-132, 1983.

[12] M. Ishikawa, T. Matsuda, T. Yoshimura and Goto, "An automatic compaction method for building blocks LSIs," *Proc. of ISCAS*. pp. 203-206, 1985.

[13] H. Shin A.L. sangiovanni-Vincentelli and C. Hsequin. "Two-dimensional compaction by Zone refining." *Proc. 23rd D.A. Conference*, pp. 115-112, 1986.

[14] R.H.J.M. Other and M. C. van Lier, "Automatic JC-layout: the geometry of the islands," *Proc. IEEE Int. Sym . on Circuits and Systems*, pp. 231-234, 1975.

[15] W.L. Schiele, "Compaction with incremental over-costraint resolution." *Proc. 25th DAC*. pp. 390-395, 1988.

[16] D. Marple, M. Smuldero, H. Hegen, "An

efficient compactor for 45 layout." *Proc. 25th DAC.* pp. 396-402, 1988.
[17] Do J. Dawson W, "SPACER: A well-behaved IC layout compactor." *Proc. VLSI* 85. pp. 283-291, 1985.
[18] Lakhani G. Varadarajan R, "A wire-length minization algorithm for circuit layout compaction," *Proc. ISCAS,* pp. 276-279, 1987.
[19] J. Bums and A.R. Nemtion. "SPARCS: A new constraint-based symbolic layout spacer." *Proc. IEEE CICC.* pp. 534-539, 1986.
[20] Wested. N. "Virtual grid symbolic layout. *Proc. 18th DAC.* pp. 225-233, 1981.
[21] Boyer, D.G. and Weste, N. "Virtual grid compaction using the most recent layer algorithm." *Proc. ICCAD,* 1192-93, 1983.
[22] M. Ishikawa, S. Goto. "Compaction-based custom LSI layout design method." *IEEE Trans. on CAD,* vol. CAD-6, no. 3, pp. 374-382, 1987. *

著 者 紹 介

李 天 熙(正會員)
1945年 6月 6日生. 1968年 한양대학교 전자공학과 졸업. 동 대학원 졸업. 1975年 성균관대학교 대학원 전자자료처리과 졸업. 1986年 성균관대학원 전자공학과 공학박사학위 취득. 1971年 (주) 한국마벨 근무. 1977年 동양공업전문대학 전자과 근무. 1979年～현재 청주대학교 전자공학과 부교수. 1983年～1985年 미국 캘리포니아 산호세 주립대학교 전산과 객원교수. 주관심분야는 VLSI Layout, ASIC, DRAM, CAD Tool 개발등임.