XXXV JUGOSLOVENSKA KONFERENCIJA ETAN-a, OHRID, 3-7. JUNA 1991.

Roman Kužnar,Baldomir Zajc
Fakulteta za elektrotehniko in računalništvo
61000 Ljubljana, Tržaška 25

Algoritem za avtomatsko razmestitev standardnih celic v integriranem vezju

An Algorithm for Placement Standard Cells in Integrated Circuit

POVZETEK — **V delu je predstavljen algoritem za avtomatsko razmestitev standardnih celic v integriranem vezju. Reševanje problema je razdeljeno v dve fazi: začetno razmestitev celic in iterativno izboljševanje začetne razmestitve. Za dosego boljših rezultatov je v vsaki fazi je uporabljena kombinacija večih metod.V prvi fazi je uporabljena metoda združevanja manjših komponemt vezja v večje na podlagi medsebojne povezanosti komponent, tako da so celice z večjo povezanostjo razmeščene bližje skupaj. Končna začetna razmestitev je dosežena z uporabo modificirane metode rasti kristala. V drugi fazi, na osnovi popolnješe informacije o topoloških lastnostih vezja, ki je dobljena iz začetne razmestitev, začetno razmestitev iterativno izboljšujemo. Uporabljena optimizacijska metoda temlji na izračunu gravitacijskih centrov, pri čemer sta za zamenjavo lokacij celic uporabljeni dve perturbacijski funkciji. Na ta način je zmanjšan problem zaustavitve optimizacije v lokalnem minimumu kriterijske funkcije.**

ABSTRACT — **This paper deals with an algorithm for placement of standard cells in integrated circuit. The algorithm for solving the problem is devided into two phases namely inital placement and iterative improvement. To achieve better results, we combine more than one method in each phase of the algorithm. One clustering method is implemented to solve the problem of relative placement, where cells with higher connectivity are placed closed together. Final initial placement is is generated by implemented cluster growth method. In second phase, the initial placement in iteratively improved, based on the information of topologic properties of a circuit that are given from initial placement. An optimization method which involve calculation of gravitational centers of cells and combine two perturbation function,x is used. In that way we decrease a problem of stoppage of the optimization process in a first local minimum of a criterion function.**

1. Uvod

Problem razmestitve standardnih celic v integriranem vezju je v literaturi dobro znan [1, 2]. V splošnem je to zaradi velikega števila celic, ki jih vezje vsebuje, težek kombinatorični problem, ki ga ponavadi rešujemo s hevrističnimi metodami. V strokovni literaturi lahko zasledimo več različnih pristopov k reševanju tega problema. Mi smo sledili ideji, kjer problem iskanja optimalne razporeditve razdelimo v dve fazi. Najprej na podlagi medsebojne povezanosti in velikosti celic generiramo začetno razmestitev celic. Ker je množina informacije o topoloških lastnostih vezja nezadostna, v splošnem z začetno razmestitvijo ne dobimo optimalnih razmer. Generiranje začetne razmestitve je razdeljeno v dva koraka: Najprej na podlagi medsebojne povezanosti med celicami gradimo binarno združevalno drevo. Na osnovi dobljenega drevesa je določeno zaporedje, po katerem bodo celice razvrščene v vrstice po principu "rasti kristala", kjer je seme (začetna celica) postavljeno v sredo sredinske celice.

V drugi fazi začetno razmestitev iterativno izboljšujemo. Ker gre v bistvu za optimizacijski proces, lahko optimizacija obtiči v lokalnem optimumu. Za rešitev tega problema je bila predlagana metoda, ki simulira počasno ohlajevanje telesa (Simulated annealing). Vendar je ta metoda časovno zelo požrešna, saj temelji na zelo velikem številu perturbacij lokacij celic. V našem algoritmu smo problem zaustavitve optimizacije v lokalnem optimumu zmanjšali tako, da smo definirali dve perturbacijski funkciji. Kakor hitro pride do zaustavitve optimizacije v lokalnem optimumu, je uporabljena druga funkcija. Proces teče

tako dolgo, dokler ne pride do zaustavitve optimizacije v končnem optimum, ki je blizu globalnega. V vsakem koraku iteracije bodo izmed vseh možnih perturbacij izbrane le tiste, ki imajo veliko verjetnost, da bodo minimizirale kriterijsko funkcijo.

Topološki model integriranega vezja s standardnimi celicami, na katerem temelji celotna zasnova algoritma izhaja iz naslednjih predpostavk:

- Celice razmeščamo v horizontalne vrstice z višino H in podano maksimalno dolžino. Višina vseh celic je enaka ($h_i = H$), širina (w_i) pa v splošnem različna. Za podano število vrstic (N_r) lahko ocenimo maksimalno potrebno dolžino celice:

$$W_{vrstice} = Round(\frac{1}{N_r}\sum_{i=1}^{N} w_i + \frac{1}{N}\sum_{i=1}^{N} w_i) \tag{1}$$

  kjer je N število celic in N_r število vrstic celic.

- Med vrsticami celic poteka $N_r - 1$ kanalov povezav, ki imajo spremenljivo širino. Širina kanala D_i je sorazmerna maksimalnemu številu trakov v kanalu.

- Dolžina povezav med celicami je merjena od sredine celic. Celotna dolžina povezav je enaka celotni dolžini povezav prirejenega Steiner-jevega drevesa in je enaka:

$$L(x,y) = \sum_{i=1}^{N-1}\sum_{j=1}^{N} c_{ij}(|x_i - x_j| + |y_i - y_j|) \tag{2}$$

  kjer so (x_i, y_i) koordinate lege celice i.

- Ocenjena površina vezja je enaka:

$$P_{vezja} = W_{vrstice} H N_r + \sum_{i=1}^{N_r-1} D_i W_{vrstice} \tag{3}$$

  kjer je $W_{vrstice}$ maksimalna dolžina vrstice.

2. Začetna razmestitev

2.1 Relativna začetna razmestitev

Podana naj bo pravokotna simetrična matrika $C = [c_{ij}]$ reda N, z vrednostjo elementov $c_{ij} = k$, pri čemer je k število povezav med celico i in j. Matriko C imenujemo matrika povezav (Connection Matrix). Z njo so definirane vse povezave med celicami vezja. Red matrike N je enak številu vseh logičnih celic vezja.

Definirati želimo množico $S = (s_1, s_2, ..., s_{2N-1})$. Element s_i imenujemo komponenta vezja. Elementi za katere velja $i < N$ so celice vezja in jih imenujemo osnovne komponente vezja. Za $i = N + 1..2N - 1$ so elementi s_i skupine komponent, ki so sestavljene iz dveh komponent:

$$s_i = (s_p, s_q); (p < i) \wedge (q < i) \tag{4}$$

Za vsak element s_i je potrebno določiti oba elementa s_p in s_q v pravilnem zaporedju. V bistvu gradimo binarno drevo, z določeno orientacijo vej. Elemente definiramo na sledeči način:

- K matriki C določimo pridruženo matriko C_s, kjer elementi matrike določajo število povezav med komponentami vezja. Za množico S imamo na začetku definiranih prvih N elementov, zato je $C_s = C$. Za $i = 1..N^*$ in $j = i + 1..N^*$ poiščimo največji element v matriki C_s (N^* je tekoči red matrike C_s). Naj bo to element $c_{s_{pq}}$. Določen je element: $s_{2N-N^*} = (s_p, s_q)$. Reduciramo matriko C_s t.j. zmanjšamo red matrike za ena tako, da seštejemo koeficiente stolpcev in vrstic p in q razan koeficienta $c_{s_{pq}}$ in zapišemo novo vrstico in stolpec. Postopek ponavljamo tako dolgo, dokler ni $N^* = 1$.

- Množica S je binarno drevo. Iz vsakega vozlišča drevesa izstopata dve veji. Imenujmo jih "leva veja" in "desna veja". Orientirati binarno drevo pomeni, da moramo za vsako vejo drevesa določiti ali je veja "leva" ali "desna".
  Predpostavimo, da so vse celice razvrščene v eno vrstico po skupinah, kakor so združene v binarnem drevesu. Pri določanju orientacije vej je izbira vozlišč določena v obratnem vrstnem redu združevanja. V prvi iteraciji je orientacija vej povsem naključna. V vsakem koraku zamenjamo za vsako vozlišče orientacijo vej. Če se dolžina povezav zmanjša, ohranimo novo orientacijo sicer je orientacija vej enaka prvotni.

Ponazorimo si postopek na preprostem zgledu. Podana naj bo matrika povezav C, reda $N = 10$.

$$C = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 & 2 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2 & 1 & 1 \\ 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 2 \\ 0 & 1 & 2 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 2 & 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 2 \\ 0 & 0 & 0 & 1 & 1 & 2 & 0 & 0 & 2 & 0 \end{bmatrix}$$

V vsakem koraku združimo par skupin komponent z največjim koeficientom $c_{s_{ij}}$. C_e je takšnih skupin več (naprimer v prvem koraku so to pari: $(3,7),(5,8),(6,10),(9,10)$, izberemo tisti par, ki ima vsoto koeficientov $c_{s_{ik}} + c_{s_{jk}}$; za $k = 1..N$ največjo. Za podano matriko povezav C_s dobimo zaporedje združevanja, kakor je prikazano na sliki 2.1 . Če je red matrike povezav C enak N , je število vseh elementov množice S enako $2N - 1$

$$\begin{aligned} 1.s_{11} &= (s_{10}, s_9) \\ 2.s_{12} &= (s_{11}, s_6) \\ 3.s_{13} &= (s_8, s_5) \\ 4.s_{14} &= (s_{12}, s_{13}) \\ 5.s_{15} &= (s_{14}, s_1) \\ 6.s_{16} &= (s_7, s_3) \\ 7.s_{17} &= (s_4, s_{16}) \\ 8.s_{18} &= (s_{17}, s_2) \\ 9.s_{19} &= (s_{18}, s_{15}) \\ S &= (s_1, s_2, .., s_{19}) \end{aligned}$$

Slika 2.1. Zaporedje združevanja

2.2 Določitev začetnih lokacij celic

Po določitvi relativne razmestitve bodo celica razvrščene v vrstice. Lokacija celice je določena, če je določena vrstica in lokacija znotraj vrstice, kamor bo celica postavljena. Določitev lokacij celic je generirana z modificirano metodo "rasti kristala" (cluster growth method).

Orientirano binarno drevo razdelimo na dve poddrevesi: S_p in S_k, kjer velja $S_{2*N-1} = (S_p, S_k)$. Pravtako razdelimo površino celotnega vezja z vertikalnim rezom na dva dela, tako da je površina vsakega dela enaka površinam P_{s_p} in P_{s_k}. Predpostavimo, da je na začetku širina vseh povezovalnih kanalov enaka in je enaka D. Za obe površini lahko določimo potrebno širino:

$$w_{s_p} = \frac{P_{s_p}}{(N_r H + D(N_r - 1))} ; w_{s_k} = \frac{P_{s_k}}{(N_r H + D(N_r - 1))} \tag{5}$$

Vse celice, ki pripadajo poddrevesu S_p, bodo razvrščene na površini, ki leži levo od vertikalnega reza, celice, ki pripadajo podrevesu S_k pa na površini desno od vertikalnega reza. Seme t.j. začetna celica bo postavljena ob vertikani rez v sredinsko. Celice bodo razvrščene po naslednjih pravilih:

- Prva celica bo izbrana iz tistega poddrevesa, ki ima več osnovnih komponent (celic). Če je to S_p, gremo od vrha poddrevesa po vseh desnih vejah, dokler ne pridemo do osnovne celice. Ta celica bo seme. Naslednja celica bo izbrana iz drugega drevesa, kakor je bila izbrana predhodna celica. Po drevesu S_p iščemo z vrha navzdol tako, da gremo v vsakem vozlišču po desni veji, dokler ne pridemo do nerazvrščene osnovne komponente (celice). Če v tekočem vozlišču v desni veji ni več nerazvrščenih komponent, gremo po levi veji do vozlišča, ki vsebuje desno vejo z nerazvrščeno osnovo komponento. Izkanje po drevesu S_k je dualno; leve veje zamenjamo z desnimi.
- Prva celica bo postavljena v sredinsko vrstico levo ali desno polek vertikalnega reza, glede na to kateremu drevesu pripada. Naslednje celice bomo izmenjaje postavljali levo od reza za celice, ki so v drevesu S_p in desno od reza za celice drevesa S_k. Postavljene bodo v vrstico, kjer bo dolžina povezav do že razmeščenih celic najmanjša. Levo od reza vrstica ne sme biti daljša od w_{s_p}, desno od reza pa ne daljša od w_{s_k}.

3. Iterativno izboljševanje začetne razmestitve

Z matriko povezav C so določene vse povezave med celicami. Z začetno razmestitvijo celic so določene koordinate lege vsake celice (x_i, y_i). Na osnovi teh podatkov lahko določimo potrebno širino vsakega kanala, celotno dolžino vseh povezav ter gravitacijske centre celic.

V optimizacijskem procesu minimiziramo funkcijo (2). Pri tem moramo upoštevati, da se širina kanala v procesu spreminja. Zaradi tega se y komponeneta lege celic spreminja in je določena po enačbi:

$$y_i = \frac{H}{2} + (r-1)H + \sum_{j=1}^{r-1} D_j \tag{6}$$

kjer je D_j število trakov v kanalu j.

Na ta način posredno minimiziramo tudi širino kanalov. Funkcijo (2) bomo minimizirali, če bomo minimizirali člene znotraj oglatih oklepajev. Pri gravitacijskih metodah vpeljemo še dodaten kriterij. Funkcijo (2) minimiziramo tako, da minimiziramo člene

$$K_i = (|x_i - x_{gi}| + |y_i - y_{gi}|) \tag{7}$$

pri čemer imenujemo par (x_{gi}, y_{gi}) gravitacijski center za celico i in je:

$$x_{gi} = \frac{1}{e_j} \sum_{i=1}^{N} c_{ij} x_i; \; y_{gi} = \frac{1}{e_j} \sum_{i=1}^{N} c_{ij} y_i; \tag{8}$$

kjer je e_j:

$$e_j = \sum_{i=1}^{N} c_{ij} \tag{9}$$

Gravitacijski center celice i je lokacija, kjer je dolžina povezav do celic s katerimi je celica i povezana, najmanjša.Celotna dolžina povezav med celicami se bo zmanjševala, če bodo lokacije celic pomaknjene bližje gravitacijskim centrom.

Definirajmo množico 'ugodnih' lokacij za vsako celico. Dolžina povezav se bo zmanjšala, če bo celica pomaknjena iz trenutne lokacije na eno izmed 'ugodnih' lokacij. Množico 'ugodnih' lokacij v bistvu tvorijo razpoložljive lokacije v okolici gravitacijskega centra. Izmed vseh 'ugodnih' lokacij za eno celico, naj bo izbrana tista, ki v največji meri minimizira dolžino povezav. Imenujmo jo 'optimalna lokacija'. Cilj optimizacijskega postopka je izkanje 'optimalnih lokacij' ter pomik celic na te lokacije. Generiranje 'ugodnih' lokacij in zaporedje pomikanja celic na nove lokacije je definirano s perturbacijsko funkcijo. Zgodi se lahko, da smo s perturbacijsko funkcijo za vse celice generirali 'ugodne' lokacije in celice pomikali na 'optimalne' lokacije ter na ta način minimizirali kriterijsko funkcijo (2), vendar smo s tem v bistvu dosegli lokalni minimum kriterijske funkcije. Problem nastane, če je lokalni minimum zelo oddaljen od globalnega, ker bo končni rezultat v bistvu slabo optimiziran. Da zmanjšamo ta problem, smo definirani dve perturbacijski funkciji. Kakor hitro dosežemo lokalni minimum z eno perturbacijsko funkcijo, uporabimo drugo in z minimizacijo funkcije (2) nadaljujemo do naslednjega lokalnega minimuma. Postopek ponavljamo dokler ne pride do zaustavitve procesa optimizacije v končnem lokalnem minimumu.

A. Perturbacijska funkcija 1 (PF1)
Definirana naj bo množica koeficientov $K = (k_1, k_2, ..k_N)$, kjer so vrednosti k_i določeni po enačbi (7). Koeficient k_i je mera, ki pove koliko je celica i oddaljena od gravitacijskega centra. Celice, uredimo po zaporedju, tako da bodo koeficienti k_i urejeni po padajočem zaporedju. S tem je določeno zaporedje pomikanja celic na nove lokacije. V enem koraku izberemo še neizbrano celico z največjim koeficentom k_i. Imenujmo jo primarna celica A. Lokacija primarne celice je nato zamenjana z lokacijami ϵ celic, za katere je pogoj:

$$\epsilon_i = |x_A - x_{g_{ci}}| + |y_A - y_{g_{ci}}| + |x_{ci} - x_{gA}| + |y_{ci} - y_{gA}| \tag{10}$$

najmanjši. Obdržimo tisto zamenjavo lokacij celic, ki je v največji meri minimizirala kriterijsko funkcijo (2). Če nobena izmed zamenjav ne minimizira (2), obdržimo razporeditev celic pred zamenjavo. Zamenjavo lokacij celic po PF1 izvajamo za v naprej določeno stevilo celic, ki imajo največje vrednosti koeficentov k_i.

B. Perturbacijska funkcija 2 (PF2)
Po PF2 celice ne zamenjujejo lokacije z lokacijami drugih celic, temveč celice pomikamo bližje k gravitacijskim centrom. Pri tem razlikujemo dva načina:

1. **način** : Celica A je pomaknjena na novo lokacijo, ki se nahaja v isti vrstici, kot je trenutna lokacija celice. Celica A je vrinjena med celici i in j, ki izpolnjujeta pogoj:

$$x_i < x_{gA} < x_j \tag{11}$$

   Če se je s pomikom zmanjšala vrednost funkcije (2), obdržimo novo razporeditev, sicer celico A vrnemo na staro lokacijo.

2. **način** : Celico A pomaknemo na novo lokacijo, ki se nahaja v drugi vrstici, kot je trenutna lega celice A. Izbrana je vrstica r, za katero je pogoj: $|y_{gA} - y_r|$ najmanjši. Celica je v vrstici r vrinjena med celici i in j, ki izpolnjujeta pogoj (11). Novo lokacijo obdržimo le, če se je s tem zmanjšala vrednost funkcije (2).

Za pomikanje na nove lokacije so najprej izbrane celice, ki imajo y-komponento gravitacijskega centra bližje vrstici celic, v kateri ne ležijo. Te celice so pomaknjene na nove lokacije po 2. načinu. V naslednji iteraciji so celice pomaknjene na nove lokacije po 1. načinu. Celice so izbrane v zaporedju, kakor ležijo v vrsticah od začetne (prve) vrstice do zadnje vrstice celic.

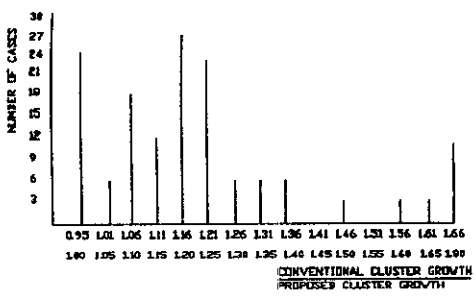
Načina 1 in 2 zaporedoma ponavljamo, dokler se optimizacija z uporabo PF2 ne zaustavi v lokalnem minimumu.

4. Testni rezultati

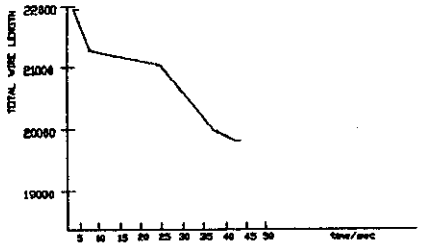
Program zasnovan na opisanem algoritmu je napisan v programskem jeziku Turbo Pascal 5.5 ter preizkušan na osebnem računalniku IBM PC - AT 80286, 16 MHz. Predstavljen je primer relativno majhnega vezja, ki vsebuje 150 standardnih logičnih celic. Majhno vezje je izbrano zato, ker smo metodo primerjali z drugimi metodami, ki so časovo zelo požrešne. Rezultate za začetno razmestitev so primerjani z rezultati metode "rasti kristala", kjer je zaporedje razvrščanja celic določeno s funkcijo IOC, predstavljeno v [6]. Po tej metodi so celice razvrščane v vrstice na prvo prosto mesto v vrstici, in sicer v tisto vrstico, kjer je dolžina povezav

do razmeščenih celic najmanjša. Pomanjkljivost te metode je v tem, da je začetna razmestitev zelo odvisna od izbire semena t.j. začetne celice. V našem algoritmu tega problema ni, kajti začetna celica je eksaktno določena. Iz primerjave rezultatov lahko ugotovimo, da je 5-15% začetnih razmestitev po primerjani metodi boljše od rezultatov, dobljenih z predstavljeno metodo. Pomanjkljivost primerjalne metode je v tem, da ne moremo v naprej vedeti, katera semena bodo dala dobre rezultate (Slika 4.1.).

Rezultati opisane iterativne metode so primerjani z rezultati klasične FDPR (Force Direct Pairwise Relaxation) metode. Obe metodi so uporabljeni za izboljšanje začetne razmestitve, ki je generirana z opisano metodo. Iz slike 4.2. je razvidno, da dobimo s kombinacijo dveh gravitacijskih lokalni minimum pomaknjen bližje k globalnemu, kakor z uporabo klasične metode FDPR.



slika 4.1



slika 4.2

5. Zaključek

V članku je predstavljen algoritem za avtomatsko razmeščanje standardnih celic na površini integriranega vezja. Predstavljeni sta dve metodi: za generiranje začetne razmestitve in za optimiranje začetne razmestitve. Z metodo graditve združevalnega drevesa je problem izbire začetnih celic (semena) zmanjšan. Celice se združujejo v večje skupine na podlagi medsebojne povezansti, zato je pričakovati, da bodo v končni fazi ležala bližje skupaj. Pomanjkljivost je lahko v tem, da celico dodamo k večji grupi celic, zato lega celice ne bo najbolj ugodna, ker je lega omejena z razporedom in velikostjo drugih celic. Pomanjkljivost je tudi v tem, da je začetna lega celice določena le z lego že razporejenih celic. Zato bodo nadaljne raziskave usmerjene v definiranje kriterijske funkcije, ki bo kompletneje zajemala informacijo dobljeno na podlagi združevalnega drevesa.

Z predstavljeno optimizacijsko metodo je problem zaustavitve optimizacije v lokalnem optimumu zmanjšan v toliko, da pride do zaustavitve optimizacije v lokalnem minimumu, ki je pomaknjen bližje globalnemu optimumu, kakor bi bilo to v primeru uporabe le ene perturbacijske funkcije. Zato v celoti ne rešuje tega problema. Opisana metoda predstavlja alternatrivo metodi, ki simulira počasno ohlajevanje telesa ("Simulated Annealing") v tem, da je potrebno dosti manj perturbacij med legami celicam, za minimizacijo celotne dolžine povezav, s čimer je poraba časa majša. Ugotovimo lahko, da v splošnem ne moremo doseči globalnega optimuma.

Literatura

[1] S.Nahar, S.Sahni, E.Shragowitz:*Simulated Annealing and combinatorial optimization.* 23^{th} Design Automation Conference, pp. 293-299, 1986.

[2] S.Mallela and L.K. Grover:*Clustering based Simulated Annealing for Standard Cell Placement.* 25^{th} ACM/IEEE Design Automation Conf., pp. 312-317, 1988.

[3] B.T.Preas and P.G.Karger:*Automatic Placement A Rewiew of Current Techniques.* 23^{th} Design Automation Conference, pp. 622-629, 1986.

[4] M.Hanan, P.K.Wolff and B.J.Anguli:*Some Experimental Results on Placement Techniques.* 13^{th} Design Automation Conference (San Francisco), pp. 214-224,1976

[5] S.Goto:*An Efficient Algorithm for the Two-Dimensional Placement Problem in Electrical Circuit Layout.* IEEE Transactions on circuit and systems, vol. cas-28,No. 1, pp. 12-18, 1981.

[6] D.G. Schweikert: *A 2-dimensional Placement Algorithm for Layout of Electrical Circuit.* 13^{th} Design Automation Conference (San Francisco), pp. 408-416, 1976