Síntese de layout

Posicionamento

CIn-UFPE/GRECO

Manoel Eusebio de Lima

Síntese de layout - posicionamento

- Posicionamento (*placement*) é a tarefa de estabelecer as posições precisas de componentes sobre um plano, considerando a otimização de área, temporização e potência.
- De uma forma geral dados os parâmetros:
  - Um conjunto de células (módulos) com geometria fixa, pinos fixos, atrasos e potência dissipada;
  - restrições quanto a posição de alguns módulos;
  - restrições quanto a área total e a relação alturaxlargura do chip;
  - uma netlist especificando quais pinos devem ser interconectados;
  - estimativa de atraso por unidade de tamanho das interconexões.
- Determine a posição final de todos os módulos considerando que todas as restrições são satisfeitas e que a área total do chip, tamanho das redes e a dissipação de potência sejam minimizados.

Posicionamento

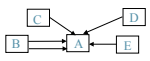
- Tipos de heurísticas:
  - construtivas
  - Iterativas
- **Heurísticas construtivas**
  - Constroem uma solução progressivamente colocando uma célula por vez a partir de uma célula semente (seed).
- **Heurísticas iterativas**
  - A partir de um pré-posicionamento, tenta-se otimizar o posicionamento final movendo-se células

Posicionamento

- Tipos de algoritmos(exemplos)
  - Simulated annealing
  - Force-Directed algoritm
  - Posicionamento por particionamento
  - Técnicas de otimização numéricas
  - Posicionamento utilizando redes neurais
  - Posicionamento utilizando algoritmos genéticos
  - .....

Posicionamento

- Force-Directed
  - A idéia básica desta classe de algoritmo é representar as interações entre as células baseada na lei de Hooke.
  - Assuma que as células (unidades funcionais) que são conectados por redes exerçam uma certa força atrativa entre si. A magnitude desta força é diretamente proporcional à distância entre as células. A constante de proporcionalidade é dada como um certo peso (k) com relação a todas as redes diretamente conectadas às células.
    - $F = k_{ij} \Delta s_{ij}$
  - com k_{ij} = soma das redes que conectam a célula i à célula j.
  - Δs_{ij} = vetor que conecta o centro da célula i à célula j, proporcional à distância entre elas.



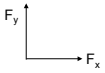
Posicionamento

- As células possuem certas posições no plano de coordenadas (x,y), e a distância entre duas células i e j, d_{ij}, pode ser calculado por:
- 1. Distância Euclidiana $|d_{ij}| = ((x_i - x_j)^2 + (y_i - y_j)^2)^{1/2}$
- 2. Distância de Manhattan $|d_{ij}| = |x_i - x_j| + |y_i - y_j|$
- Normalmente o ponto de referência de cada módulo é dada pelo centro do módulo.
- **Idéia do algoritmo**

  Se as células de um dado sistema podem mover-se livremente, elas se moveriam na direção da força resultante mínima sobre elas, até o sistema entrar em equilíbrio, em um estado de mínima energia. Isto é, equivalente a termos molas com a mínima tensão ou em circuitos, tamanho mínimo para as redes. A força resultante no estado de equilíbrio deve ser zero sobre cada célula.

Posicionamento

- Considere peso w_{ij} = no. de redes entre duas células M_i e M_j
- Considere também s_{ij} a distância entre M_i e M_j
- A força da rede sobre a célula é dada por:
  $F = \Sigma\, w_{ij} . s_{ij}$
- Se a força resultante deve ser zero, isto significa que, considerando as duas componentes ortogonais teríamos:
  $\Sigma\ w_{ij} (x_i - x_j) = 0$, $\Sigma\ w_{ij} (y_i - y_j) = 0$



- Assim, as coordenadas do módulo i para a força resultante ser zero seriam dadas por:
  $| x_i |_{F=0} = \Sigma\ w_{ij} . x_j / \Sigma\ w_{ij}$ com j= 1 ..n
  $| y_i |_{F=0} = \Sigma\ w_{ij} . y_j / \Sigma\ w_{ij}$

Posicionamento

- Algoritmos baseados em força podem ser implementados em vários estilos:
  - Métodos construtivos
  - Métodos iterativos
- Método Construtivos
  - Neste método não existe um posicionamento inicial das células. As coordenadas de cada célula são tratadas como variáveis e a força das redes exercida sobre cada célula por todos as outras células é igual a zero.
  - A posição final das células sujeita as forças existentes no sistema pode ser encontrada resolvendo um sistema de equações do tipo:
    $\Sigma\ F_i = 0$
    $\Sigma\ F_{i+1} = 0$
    .....

Posicionamento

- Considerando que nenhum módulo tem uma posição fixa, então a solução trivial para o problema é dada por:
  $x_1 = x_2 = x_3 = x_4$ ..... x_n ;
  $y_1 = y_2 = y_3 = y_4$ ..... y_n ;
  Os módulos ficarão sobrepostos, o que não é uma solução viável.

  Para evitar sobre-posição (*overlapping*), algumas restrições podem ser consideradas, tais como:
  1. Estabelecer-se que dois módulos ou células não podem sobrepor-se.
  2. Com a colocação de "pads" na periferia do chip e considerando a força exercida pela conexões entre os pads e as células evitará temporariamente o completo *overlapping* de todas as células. No entanto, em geral, um pós-processamento para eliminar o resto dos *overlapping* é necessário.
  3. Um outro conceito para ajudar o posicionamento e reduzir *overlapping* é a idéia de forças repulsivas, as quais atuam nos módulos que não possuem conexão entre si.

Posicionamento

- Força repulsiva
  - Pode ser constante
  - Pode ser inversamente proporcional à distância entre células
- Força de repulsão
  $F_i = \Sigma - k_{ij} . \Delta s_{ij} + R_{ij}$

Considere:
k_{ij} = Número de redes entre células i e j.
Δs_{ij} = Total dos tamanhos das redes entre células i e j
R = Força de repulsão proporcional a célula j que não possui conexão com a célula i.

Onde
$| R_{ij} | = 0$ para $k_{ij} \neq 0$ ou
$| R_{ij} | = 0$ para i=j ou
$| R_{ij} | = R$ para $k_{ij} = 0$ (sem conexões)

Posicionamento (exemplo)

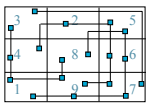
- Método Iterativo
  - Em geral o método construtivo não gera um posicionamento satisfatório, devido em geral a simplicidade das considerações, quanto a posição de pinos e o fato que apenas translações são consideradas na pesquisa de configurações de espaço.
  - O método iterativo tenta refinar o resultado de um posicionamento previamente desenvolvido por um algoritmo construtivo, ou através de uma geração inicial aleatória.
- **Exemplo**
  - Force-direct relaxation

Posicionamento - exemplo

- **Force- direct relaxation**
  1. Uma célula é selecionada de cada vez. Como selecionar a primeira célula?
     a. Randomicamente
     b. Considerando a célula com maior força sobre ela
     c. Relacionando o módulo em função de sua conectividade
  2. Onde colocar a célula selecionada?
     a. Move-se a célula para o slot mais próximo à posição de força zero. Se esta posição já está ocupada, move-se a célula para a região mais próxima livre. Este tipo de solução pode colocar a célula numa região longe da posição ideal.
     b. Computa-se a posição desejada da nova célula e faz-se uma estimativa de ganho na troca de células se a posição estiver ocupada, em função do tamanho de interconexões. A mudança do tamanho das interconexões ou custo quando as células são trocadas é avaliada e, se houver redução no comprimento das interconexões a troca é aceita, caso contrária é rejeitada.
  3. Executa um "ripple move"
     a. Selecione a célula que ocupa o ponto destino para o próximo movimento. Este processo continuará até o ponto destino ser um espaço vazio ou a mesma posição. Então uma nova semente é selecionada.

Posicionamento

- Exemplo
  - Considere o circuito abaixo com um pré-posicionamento. Utilize Um algoritmo baseado em força para otimizar o posicionamento das células.



net 1 = { 1,3,4,8,9 }
net 2 = {1,5,6,7,8,9}
net 3 = {2,4,5,6,7,9}
net 4 = {3,7}

Matriz conectividade

célula	1	2	3	4	5	6	7	8	9	Σ
1	0	0	1	1	1	1	1	2	2	9
2	0	0	0	1	1	1	1	0	1	5
3	1	0	0	1	0	0	1	1	1	5
4	1	1	1	0	1	1	1	1	2	9
5	1	1	0	1	0	2	2	1	2	10
6	1	1	0	1	2	0	2	1	2	10
7	1	1	1	1	2	2	0	1	2	11
8	2	0	1	1	1	1	1	0	2	9
9	2	1	1	2	2	2	2	2	0	14

Somatório de redes

- Algoritmo "Force-directed placement"; (Shahookar and Mazumber, 1991)

```
gerar matriz conectividade da netlist;
ordenar as células em ordem decrescente de suas conectividades->L;
interation_limit = x; / inteiro definido pelo projetista - no. de iterações;
abort_count = y; / inteiro definido pelo projetista - no. de abortes;
abort_limit = w; /  inteiro limite - no. abortes limite;
end_ripple = false; / indica quando uma célula encontrou uma posição;
interation_count = 0; / contador
While (interation_count < interation_limit)
   Seed = próximo módulo da lista L; (maior conectividade)
   declare a posição do módulo Seed como vaga; (VACANT)
   While end_ripple = false
     compute o ponto destino para o módulo selecionado e
     arrendonde valor das coordenadas para o inteiro mais próximo;
     Case target point;
        VACANT;
              Mova Seed para o ponto selecionado e trave (lock);
              end_ripple <- true;
              abort_count <- 0;
        SAME LOCATION (unlock)
              end_ripple <- true;
              abort_count <- 0;
```

continua


```
        LOCKED:
              Mova a célula selecionada para a posição vaga mais
              próxima;
              end_ripple <- true;
              abort_count <- abort_count+1;
              If abort_count > abort_limit Then
                    UNLOCK todas as posições das células;
                    iteration_count <- iteration_count+1;
              ENDIF;
        OCCUPIED: (*e não LOCKED*)
              Ocupe célula como ponto destino (target) e selecione a
              célula target  para o próximo movimento;
              end_ripple <- false;
              abort_count <- 0;
     EndCase;
   EndWhile;
 EndWhile;
End
```


Posicionamento

Modelo para o cálculo das coordenadas de cada nova posição de cada célula

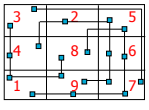
- Considere peso w_{ij} = no. de redes entre duas células Mi e Mj
- Considere também s_{ij} a distância entre Mi e Mj
- A força da rede sobre a célula é dada por:
  $F = \Sigma\ w_{ij} . s_{ij}$
- Se a força resultante deve ser zero, isto significa que, considerando as duas componentes ortogonais teríamos:
  $\Sigma\ w_{ij} (x_i - x_j) = 0$, $\Sigma\ w_{ij} (y_i - y_j) = 0$
  Assim, as coordenadas do módulo i para a força resultante ser zero seriam dadas por:

$| x_i |_{F=0} = \Sigma\ w_{ij} . x_j / \Sigma\ w_{ij}$ com j= 1 ..n

$| y_i |_{F=0} = \Sigma\ w_{ij} . y_j / \Sigma\ w_{ij}$

Posicionamento

- Force-direct relaxation (Exemplo)



net 1 = { 1,3,4,8,9 }
net 2 = {1,5,6,7,8,9}
net 3 = {2,4,5,6,7,9}
net 4 = {3,7}

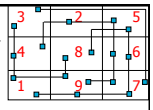
Considerações:
abort_limit = 3
iteration_count = 2

Matriz conectividade

célula	1	2	3	4	5	6	7	8	9	Σ
1	0	0	1	1	1	1	1	2	2	9
2	0	0	0	1	1	1	1	0	1	5
3	1	0	0	1	0	0	1	1	1	5
4	1	1	1	0	1	1	1	1	2	9
5	1	1	0	1	0	2	2	1	2	10
6	1	1	0	1	2	0	2	1	2	10
7	1	1	1	1	2	2	0	1	2	11
8	2	0	1	1	1	1	1	0	2	9
9	2	1	1	2	2	2	2	2	0	14

Célula 9 é a célula de maior conectividade

Posicionamento inicial



Posicionamento

Iteração	Módulo selecion.	Ponto destino	Status	Posição final	Resultado
1	9(seed)	(1.1,1)	occupied ->	(1,1)	3 2 5
			ação -> lock	(1,1)	4 9 6
			end_ripple = false		1 - 7
	8	(0.9,0.9)	locked ->	(1,1)	3 2 5
			abort_count=1	(1,0)	4 9 6
			end_ripple = true		1 8 7
	7(seed)	(1.1,1.2)	locked ->	(1,1)	3 2 5
			abort_count=2	(2,0)	4 9 6
			end_ripple = true		1 8 7
	6(seed)	(1.2,0.9)	locked ->	(1,1)	3 2 5
			abort_count=3	(2,1)	4 9 6
			end_ripple = true		1 8 7

Posicionamento

Iteração	Módulo selecion.	Ponto destino	Status	Posição final	Resultado
2	9(seed)	(1.1,0.9)	the same ->	(1,1)	3 2 5 4 9 6 1 8 7
			ação->(unlock)	(1,1)	
			end_ripple = true		
	7(seed)	(1.1,1.2)	occupied ->	(1,1)	3 2 5 4 7 6 1 8 -
			ação -> lock	(1,1)	
			end_ripple = false		
	9	(0.9,1.0)	locked ->	(1,1)	3 2 5 4 7 6 1 8 9
			abort_count=1	(2,0)	
			end_ripple = true		
	6(seed)	(1.2,0.9)	locked ->	(1,1)	3 2 5 4 7 6 1 8 9
			abort_count=2	(2,1)	
			end_ripple = true		

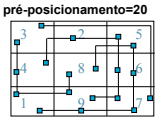
Posicionamento

Iteração	Módulo selecion.	Ponto destino	Status	Posição final	Resultado
2	5(seed)	(1.2,0.7)	locked ->	(1,1)	3 2 5 4 7 6 1 8 9
			abort_count=3	(2,2)	
			end_ripple = true		

- **Resultado final do posicionamento após duas iterações**

*** Posicionamento escolhido**

Σtamanho das redes no pré-posicionamento=20



Σtamanho das redes após do posicionamento=17