COMPARAÇÃO DE POSICIONAMENTO SIMULATED ANNEALING E QUADRATURA NO GERADOR AUTOMÁTICO DE MACRO-CÉLULAS TROPIC

Renato Hentschke[1], Diogo Fiorentin[1], Fernando Moraes[2], Ricardo Reis[1]

[1] *Instituto de Informática - UFRGS - Universidade Federal do Rio Grande do Sul*
CP15064 - Campus do Vale - CEP 915011-970 - Porto Alegre - Brasil

[2] *Faculdade de Informática - PUCRS - Pontifícia Universidade Católica do Rio Grande do Sul*
Av. Ipiranga, 6681 P 30/BL 4 – CEP 90619-900 - Porto Alegre - Brasil

ABSTRACT

This paper compares two placement algorithms: quadratic partitioning and simulated annealing. The automatic layout synthesis tool, TROPIC [9], is used to generate the benchmarks, allowing area and wirelength evaluation. Experimental results show that the simulated annealing algorithm can reduce the wirelength up to 10% in small circuits (less than 3000 cells), and it is expected better results for complex modules. The reduction on wirelength reduces the final silicon area, delay and power. The simulated annealing algorithm drawback is the long CPU time when compared with a determinist algorithm such as quadratic partitioning. The simulated annealing algorithm is up to 3 orders of magnitude slower.

RESUMO

Este artigo trata da comparação de dois algoritmos de posicionamento: Particionamento em Quadratura e *Simulated Annealing*. A comparação é feita no ambiente de geração automática de layout TROPIC [9]. Os resultados experimentais mostram que o *Simulated Annealing* pode reduzir o tamanho dos fios em 10% para os circuitos menores (menos de 3000 células), enquanto que nos circuitos maiores há perspectiva de crescimento desta vantagem. A redução de tamanho dos fios leva a melhora de densidade (menor área), atraso e dissipação de potência. O custo deste ganho é o excessivo tempo de CPU requerido pelo *Simulated Annealing*, que é até 3 ordens de grandeza mais lento.

COMPARAÇÃO DE POSICIONAMENTO SIMULATED ANNEALING E QUADRATURA NO GERADOR AUTOMÁTICO DE MACRO-CÉLULAS TROPIC

Renato Hentschke[1], Diogo Fiorentin[1], Fernando Moraes[2], Ricardo Reis[1]

[1] Instituto de Informática - UFRGS - Universidade Federal do Rio Grande do Sul
CP15064 - Campus do Vale - CEP 915011-970 - Porto Alegre - Brasil

[2] Faculdade de Informática - PUCRS - Pontifícia Universidade Católica do Rio Grande do Sul
Av. Ipiranga, 6681 P 30/BL 4 – CEP 90619-900 - Porto Alegre - Brasil

ABSTRACT

This paper compares two placement algorithms: quadratic partitioning and simulated annealing. The automatic layout synthesis tool, TROPIC [9], is used to generate the benchmarks, allowing area and wirelength evaluation. Experimental results show that the simulated annealing algorithm can reduce the wirelength up to 10% in small circuits (less than 3000 cells), and it is expected better results for complex modules. The reduction on wirelength reduces the final silicon area, delay and power. The simulated annealing algorithm drawback is the long CPU time when compared with a determinist algorithm such as quadratic partitioning. The simulated annealing algorithm is up to 3 orders of magnitude slower.

RESUMO

Este artigo trata da comparação de dois algoritmos de posicionamento: Particionamento em Quadratura e *Simulated Annealing*. A comparação é feita no ambiente de geração automática de layout TROPIC [9]. Os resultados experimentais mostram que o *Simulated Annealing* pode reduzir o tamanho dos fios em 10% para os circuitos menores (menos de 3000 células), enquanto que nos circuitos maiores há perspectiva de aumento desta vantagem. A redução de tamanho dos fios leva a uma melhora da densidade (menor área), atraso e dissipação de potência. O custo deste ganho é o excessivo tempo de CPU requerido pelo *Simulated Annealing*, que é até 3 ordens de grandeza mais lento.

1. INTRODUÇÃO

A etapa de posicionamento é de significativa importância para a síntese física do circuito. Seu papel é de definir a posição das células ao longo do espaço no circuito integrado. Desta forma, o posicionamento influi diretamente no roteamento, definindo o congestionamento e tamanho mínimo das conexões.

Nas tecnologias modernas, o problema de congestionamento é de crescente importância, já que há uma quantidade muito grande de elementos a serem roteados, demandando mais espaço para conexões. Ao mesmo tempo, o aparecimento de novos níveis de metal para rotear aumenta as possibilidades de roteamento. Porém, o custo de área é crescente e exige-se que haja máxima densidade de elementos, evitando-se, assim, canais de roteamento.

Ainda, as tecnologias modernas definem alguns novos problemas elétricos que devem ser tratados no nível de posicionamento. O atraso das conexões é da mesma ordem de grandeza que o atraso das células, fazendo com que a análise de atraso das conexões faça parte do processo de posicionamento. Ao mesmo tempo, a análise de dissipação de potência e sua distribuição ao longo do chip são também problemas tratados no posicionamento, como em [4].

Desta forma, as novas tecnologias exigem um posicionamento com múltiplos objetivos (minimização do tamanho dos fios, congestionamento, potência e atraso,).

Os algoritmos de posicionamento existentes na literatura se classificam em construtivos e iterativos. Os algoritmos construtivos têm como função de gerar um posicionamento completo a partir apenas da descrição do *netlist* do circuito. Os algoritmos iterativos, por outro lado, necessitam de um posicionamento inicial (gerado por algum posicionamento construtivo). Entre os algoritmos construtivos mais conhecidos, destaca-se o posicionamento aleatório (usado como posicionamento inicial), posicionamento direcionado a forças [6] e posicionamento baseado em particionamento (onde se destaca a Quadratura) [5]. Entre os algoritmos iterativos, destacam-se as meta-heurísticas (*Simulated Annealing*, Algoritmos Genéticos, Busca Tabu).

Reconhecidamente, os algoritmos construtivos são mais rápidos que as meta-heurísticas. Com o crescimento da quantidade de células para posicionar, gradualmente a pesquisa em posicionamento se direcionou para

algoritmos construtivos. Hoje, há um esforço por incluir, nestes algoritmos, novas estruturas que possam levar em conta as questões de tecnologias modernas (congestionamento, *timing* e potência), como em [6][7]. Porém, ainda são muito poucos os trabalhos que conseguem reunir um posicionamento construtivo com múltiplos objetivos.

As meta-heurísticas, por outro lado, possuem uma função objetivo genérica, sendo naturalmente adaptadas a múltiplos objetivos, como em [3]. A sua principal limitação é no tempo de execução, que é ordens de grandeza superior ao tempo de um algoritmo construtivo, como o posicionamento em quadratura. Por outro lado, são propostos uma série de trabalhos que visam a otimização do tempo de CPU do Simulated Annealing com *schedules* adaptativas [8][7], perturbações inteligentes [1][7], e funções de custo otimizadas [7].

O gerador de macro-células chamado de Tropic3 [9] utiliza em sua versão atual o algoritmo de posicionamento em quadratura. Este trabalho busca verificar quais são os ganhos do uso de posicionamento com *Simulated Annealing* em relação a Quadratura. Deseja-se verificar se há vantagem de tamanho total de fios e densidade de layout no uso do *Simulated Annealing*. A implementação utilizada tem como função de custo a minimização do tamanho total dos fios. Não é considerada nenhuma técnica de função multi-objetivo.

Este artigo está organizado como segue. A seção 2 discute as características da implementação de *Simulated Annealing* utilizada. A sessão 3 comenta a metodologia de comparação utilizada. A sessão 4 apresenta os resultados experimentais. Finalmente, a sessão 5 mostra as conclusões e trabalhos futuros.

2. ALGORITMO SIMULATED ANNEALING

O algoritmo *Simulated Annealing* baseia-se em modificações (ou perturbações) aleatórias sobre o estado atual de posicionamento, até que se atinja um estado aceitável. A cada iteração, uma perturbação é seguida de uma função de aceitação, que diz se o novo estado deve ser descartado ou mantido. A função deve ser escrita de tal forma que no começo do processo ela aceite uma quantidade maior de perturbações (inclusive as que pioram o estado atual), enquanto que no final do processo ela aceite somente modificações que melhoram o estado atual. Este processo é implementado através de um parâmetro externo, chamado de temperatura. No começo a temperatura é alta, e deve ser gradativamente diminuída, a cada iteração.

Assim, o *Simulated Annealing* é definido por uma função de perturbação, uma função de *schedule* da temperatura e uma função de aceitação.

A função de **perturbação** utilizada é descrita com mais detalhe em [7]. Basicamente, ela se baseia no movimento de uma célula para outra parte do layout. Este movimento é realizado por quatro funções diferentes, que são chamadas alternadamente. A primeira função, chamada de *single perturbation*, escolhe uma célula aleatoriamente, e move esta célula para uma posição aleatória. A segunda função é chamada de *double perturbation*, e faz uma simples troca de posição de duas células selecionadas aleatoriamente. As outras duas funções de perturbação são gulosas, por realizarem movimentos heuristicamente determinados. O uso das quatro funções alternadamente garante ao algoritmo uma rápida convergência sem ficar preso em máximos locais.

A função de **aceitação** baseia-se em uma equação básica, como é mostrada na Figura 1 [8][2]. Observa-se que ela é dependente de duas variáveis: delta e temperatura. O delta é a variação do custo que o posicionamento foi submetido através da perturbação. Em outras palavras, dependendo da variação do custo obtida pela perturbação, é decidido se o estado atual será mantido ou não. A função de custo utilizada neste trabalho leva em conta somente o tamanho total dos fios. Ela é implementada de tal forma que se compute somente os fios que foram modificados pela perturbação anterior, de forma a otimizar o tempo total de processamento.

$$prob = e^{\frac{-delta}{temperatura}}$$

Figura 1 - Função de aceitação.

A função de ***schedule*** define uma temperatura inicial e uma variação da temperatura. A temperatura inicial é adaptada ao circuito utilizado. Sabe-se que, quanto maior o circuito, maior deve ser a temperatura inicial para produzir o mesmo efeito. Além disto, deve ser escolhido um método de otimização. O processo conhecido por **high-annealing** é baseado no uso de uma temperatura alta no começo do processo, fazendo com que a solução inicial seja destruída e evitando que o algoritmo fique preso em otimizações parciais. O processo de **low-annealing** parte da solução inicial e, portanto, deve encontrar uma temperatura adequada para que a solução inicial não seja destruída.

O método de *high-annealing* exige o uso de mais iterações que o método de *low-annealing* para compensar o tempo perdido com a destruição da solução inicial. Neste trabalho, a temperatura inicial adequada para *low-annealing* foi determinada experimentalmente. É proposto como trabalho futuro um método automático para sua determinação.

A temperatura final é fixa em zero. Usam-se um número fixo e pré-determinado de iterações até que se

atinja o final do processo, diferentemente de outras *schedules* de temperatura, como [8].

A variação da temperatura é feita através de uma curva de terceiro grau com ponto de inflexão variável. No processo de *high-annealing*, é desejado que a curva tenha rápida convergência, para diminuir a temperatura inicial rapidamente. A figura 2.a mostra dois exemplos de curvas adequadas para high-annealing. No processo de *low-annealing*, esta curva deve ser mais próxima de uma reta conectando a temperatura inicial (na figura 2: ti) com a temperatura final (que vale zero), como mostra a figura 2.b.

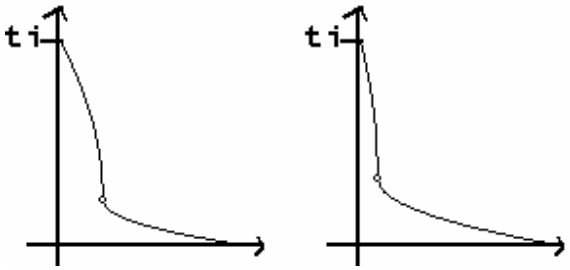

Figura 2.a – Variações de temperatura adequadas para *high-annealing*.

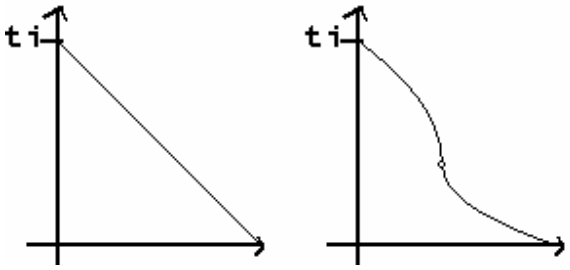

Figura 2.b – Variações de temperatura adequadas para *low-annealing*.

3. METODOLOGIA DE EXPERIMENTOS

O gerador de layout Tropic é totalmente automatizado, sem a utilização de bibliotecas *Standard Cell*. A descrição de entrada é um *netlist* em formato SPICE, no nível lógico, com a utilização de sub-circuitos. A layout de saída é no formato CIF. O desenho do layout é baseado em células de altura variável, posicionada em bandas. O processo de geração do layout inicia-se com a leitura e planificação do *netlist* de entrada. A partir de então, é feito o posicionamento e geração do layout das células. Finalmente, é realizado o roteamento com a utilização de 3 níveis de metal. A maior parte do roteamento é realizado em canais de roteamento entre as bandas de células. O roteador do Tropic permite que se abram espaços entre as bandas de células para a passagem de conexões. Desta forma, tem-se que a altura total das bandas será dependente da demanda por conexões. Sabe-se que a demanda por conexões é função da qualidade do posicionamento. Assim, a densidade do layout é diretamente proporcional à qualidade do posicionamento.

O experimento realizado substitui o algoritmo de posicionamento da versão atual do Tropic (baseado em Quadratura) pela implementação do Simulated Annealing vista na sessão 2 deste artigo e também em [7]. Foram comparados os seguintes parâmetros:

- Área do Bloco, em μm^2
- Dimensões do layout do bloco gerado – XxY, em µm
- Comprimento Médio das Conexões – C, em µm
- Porcentagem de conexões menores que 100 µm - % 0-100
- Porcentagem de conexões menores que 200 µm e maiores que 100 µm - % 100-200
- Densidade – D, em transistores por milímetro quadrado
- Número de Bandas

Todas as métricas avaliam a qualidade do posicionamento, em diferentes perspectivas. Tanto área, quanto dimensão e densidade são medidas de roteabilidade, pois estão relacionadas com a demanda por recursos de roteamento. Comprimento Médio das Conexões também é uma métrica de roteabilidade com relação indireta a *timing* e dissipação de potência. A porcentagem de conexões menores que 200µm está relacionada com a proximidade de células conectadas, e indiretamente com o desempenho do circuito, uma vez que conexões longas demais podem significar muito atraso para as conexões críticas.

4. RESULTADOS

A tabela 1 mostra os resultados obtidos com a utilização de alguns benchmarks. Os algoritmos utilizados na comparação são: Quadratura (Tropic), *Simulated Annealing* usando **low-annealing** (300 iterações e 3 repetições da mesma iteração por célula) e *Simulated Annealing* usando **high-annealing** (400 iterações e 9 repetições por célula).

Considere, inicialmente, a porcentagem de conexões menores que 200µm como métrica para qualidade do posicionamento. O gráfico da figura 3 mostra as porcentagens para os circuitos da tabela 1 ordenados por complexidade. Observe a importância desta métrica para a avaliação do posicionamento. Conexões muito grandes demandam a inserção de *buffers*, aumentando o consumo de potência e área do circuito. Assim, é desejado que a porcentagem de conexões menores que 200µm seja maior possível.

Tabela 1 – Comparação entre os algoritmos de posicionamento Quadratura (Tropic) e *Simulated Annealing* (High/Low).

Bench	Algoritmo	Area	XxY	C	%0-100M	%100-200M	D	Bandas
C432_4x4 134 células	Low 3			-				8
	High 9	0,01006	77,25 x 130,20	58,15	83,3%	15,5%	68,8K	8
	Tropic	0,01115	101,70 x 109,60	66,11	79,5%	18,4%	62,0K	6
C499 405 células	Low 3	0,02281	193,00 x 118,20	58,14	84,7%	11,5%	68,2K	7
	High 9	0,02102	192,45 x 109,20	54,03	87,9%	8,8%	74,0K	7
	Tropic	0,02279	194,45 x 117,20	57,62	85,5%	11.9%	68,2K	7
Bw 473 células	Low 3	0,01972	168,25 x 117,20	48,22	91,8%	5,5%	63,6K	7
	High 9	0,01833	169,45 x 108,20	45,60	93,7%	4,0%	68,5K	7
	Tropic	0,01993	168,60 x 118,20	47,98	91,6%	4,6%	63,0K	7
C1908_3x3 783 células	Low 3	0,04228	259,70 x 162,80	59,92	89,1%	9,9%	57.9K	9
	High 9	0,04030	259,00 x 155,00	53,91	92,8%	6,8%	61K	9
	Tropic	0,04556	258,00 x 176,60	61,69	86,5%	12,3%	53,9K	9
Mult 1468 células	Low 3	0,13550	433,45 x 312,60	71,82	88,5%	7,7%	63,3K	16
	High 9	0,13795	433,00 x 318,60	71.08	90,0%	6,2%	62,2K	16
	Tropic	0,14757	432,00 x 341,60	80,59	79,0%	15,6%	58,1K	16
C53 2307 células	Low 3	0,24761	530,45 x 466,80	86,11	85,6%	5,6%	43,0K	18
	High 9	0,21566	531,45 x 405,80	73,07	88,3%	4,9%	49,4K	18
	Tropic	0,26206	530,70 x 493,80	103,09	81,1%	8,1%	40,6K	18

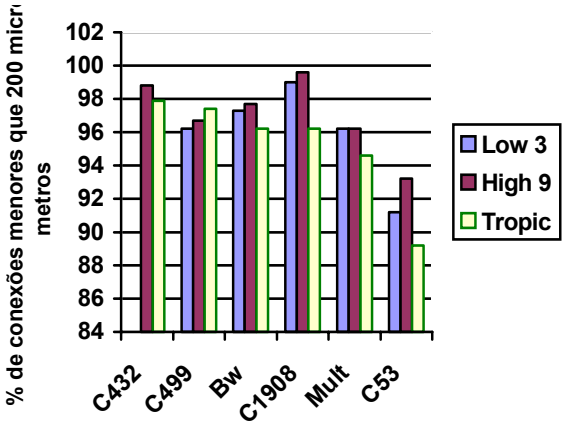

Figura 3 – Porcentagem de conexões com comprimento inferior a 200 μm.

Observando isoladamente o posicionamento Quadratura, observa-se que com o crescimento da complexidade do circuito diminui a qualidade do posicionamento. Isto demonstra que o particionamento em quadratura implementado no Tropic começa a apresentar problemas conforme cresce o circuito. A quadratura é uma heurística incompleta, pois um número excessivo de particionamentos pode afastar demasiadamente elementos que devem se conectar. Assim, conforme cresce o tamanho do circuito, a quadratura começa a ter dificuldades, pois necessita de mais partições.

Por outro lado, o *Simulated Annealing* é mais regular, apresentando um forte declínio somente no circuito C53, que é bastante complexo. Observe como o desempenho do Mult é comparável com o C499.

A coluna 'C' da tabela 1 indica o comprimento médio das conexões. Observe que este dado é praticamente independente do número de células. Esta informação é muito importante, pois permite uma avaliação prévia de atraso para uma dada tecnologia, sem a síntese do layout.

O gráfico da figura 4 foca em outra métrica de avaliação do posicionamento: a densidade os circuitos. O gráfico da figura 4 mostra que com o uso de *Simulated Annealing* pode-se obter menor ocupação da área. O gráfico da figura 5 reproduz o gráfico anterior, apresentando a taxa de ganho de densidade do Simulated Annealing em relação ao Quadratura.

Observa-se que o ganho do Simulated Annealing em relação ao posicionamento Quadratura fica próximo de 10% para os *benchmarks* mais simples, enquanto que no C53 atinge 18%. A tendência, como foi visto, é que este ganho seja cada vez maior.

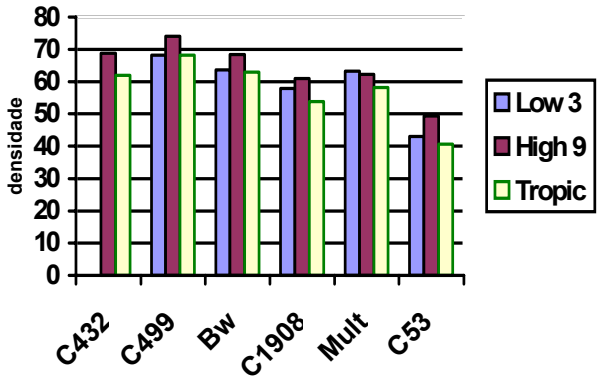

Figura 4 – Densidade, em 1000 transistores por milímetro quadrado.

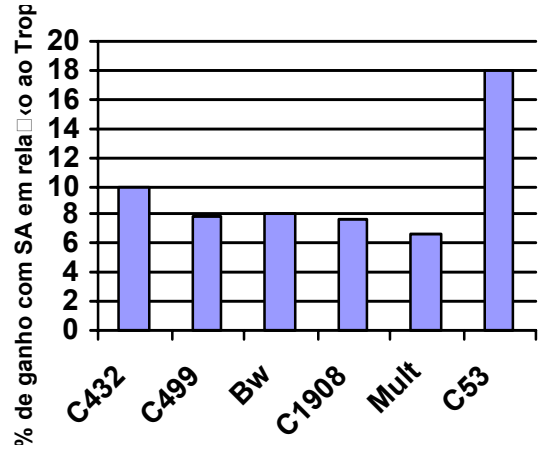

Figura 5 – Porcentagem de ganho em densidade (tr/mm^2) utilizando o posicionamento *Simulated Annealing* e não quadratura no gerador de layout Tropic.

A tabela 2 compara o tempo do Simulated Annealing rodando em um Pentium IV 1,8 GHz contra o Quadratura rodando em uma SUN UltraSparc 10. A comparação é imprecisa, pois são CPUs diferentes, porém permite ilustrar a ordem de grandeza da diferença de tempos entre os algoritmos.

Tabela 2 – Comparação do tempo de execução dos algoritmos de posicionamento *Simulated Annealing* e quadratura.

Circuito	Low3	High 9	Quadratura
C499	180 s	600 s	1 s
C53	7800 s	30600 s	11 s

5. CONCLUSÃO E TRABALHOS FUTUROS

Em uma análise geral dos gráficos, o *Simulated Annealing* mostra uma vantagem na qualidade do posicionamento em relação ao Quadratura. Ou seja, à custa de maior tempo de processamento (até 3 ordens de grandeza), obtem-se um resultado de posicionamento melhor. Observe que o *Simulated Annealing* propicia que o desempenho seja ainda melhor, bastando aumentar o número de repetições. Dentre as duas técnicas de *Simulated Annealing*, observa-se que o high-annealing possui um desempenho melhor às custas de mais iterações (400 iterações com 9 repetições por célula no high-annealing, 300 iterações e 3 repetições por célula para low). Porém, os dados do low-annealing são interessantes por mostrar que é possível gastar menos tempo com *Simulated Annealing* com resultados ainda bons, melhores em média que o quadratura. O desenvolvimento de *schedules* adaptativas à solução de posicionamento inicial podem melhorar os ganhos do método de *low-annealing*.

Como trabalhos futuros, inicialmente deve ser respondida a seguinte pergunta: a vantagem obtida é devido a uma limitação do algoritmo de quadratura ou devido a uma limitação da heurística de particionamento utilizada? Será que melhores algoritmos para particionamento não iriam propiciar um melhor desempenho da quadratura?

Um segundo trabalho futuro está focado no uso de *Simulated Annealing* com objetivo de minimização de potência, atraso e congestionamento.

Por fim, deseja-se encontrar um método para determinação automática da probabilidade inicial a ser usada pelo processo de *low-annealing*.

6. REFERÊNCIAS

[1] SU, Lixin; BUNTINE, Wray; NEWTON, Richard. **Learning as Applied To Stochastic Optimization for Standard-Cell Placement**. IEEE Transactions on Computer-Aided Design of Integrated Circuits And Systems, v.20(**4**), April 2001**,** p. 516 -527

[2] SHERWANI, Naveed A. **Algorithms for VLSI Physical Design Automation.** Third Edition, Kluwer Academic Publishers, 1998.

[3] SAIT S, YOUSSEF H, KHAN J, MALEH A. **Fuzzy Simulated Evolution For Power And Performance Optimization Of VLSI Placement**. In: IJCNN'01 (International Joint Conference on Neural Networks), 2001, p. 738 -743.

[4] TSAI, Ching-Han and KAND, Sung-Mo. **Cell-level Placement for Improving Substrate Thermal Distribution**. IEEE Transactions on Computer-Aided Design of Integrated Circuits And Systems, Feb. 2000, p. 253-266.

[5] PARAKN P., BROWN R., SAKALLAH K. **Congestion Driven Quadratic Placement.** In: Design Automation Conference, 1998, p. 275-278.

[6] CHOU, Yih-Chih and LIN, Yon-Long. **Effective Enforcement of Path-Delay Constraints in Performance-Driven Placement**. IEEE Transactions on Computer-Aided Design of Integrated Circuits And Systems, January 2002, p. 15-22.

[7] HENTSCHKE R., REIS R. **Random and Greedy Mixed Moves aplyied to Simulated Annealing Placement.** Submetido para avaliação ao DAC 2003, Anaheim, EUA.

[8] AARTS E. H. L. and LAARHOVEN P.J.M.. **A New Polynomial-Time Cooling Schedule**. In: ICCAD'85, 1985.

[9] MORAES, Fernando Gehm; ROBERT, Michel; AUVERGNE, Daniel. **A Virtual CMOS Library Approach for Fast Layout Synthesis.** In: VLSI: Systems on a Chip, 1999, p. 415-426.