UM NOVO ALGORITMO DE ROTEAMENTO DE PROPÓSITO GERAL

STÈPHANO GONÇALVES; LEOMAR DA ROSA JR.; FELIPE MARQUES

Universidade Federal de Pelotas – {smmgoncalves, leomarjr, felipem}@inf.ufpel.edu.br

1. INTRODUÇÃO

Com o avanço da tecnologia, os circuitos integrados tornaram-se cada vez mais complexos, exigindo o uso de ferramentas CAD para automatizar seu processo de concepção. Este processo é composto por várias etapas, sendo a síntese física uma das principais fases. A síntese física é composta pela fase de posicionamento, onde as células do circuito são posicionadas seguindo determinados critérios, e a fase de roteamento, que consiste em realizar as conexões dos componentes por meio de fios. Dessa forma, a etapa de roteamento busca os menores caminhos entre os componentes, desviando de todos obstáculos.

O primeiro algoritmo de roteamento, chamado *Maze Router*, foi proposto por Lee (1961). Este algoritmo é ótimo, em relação ao comprimento do fio, mas possui um elevado custo em tempo de processamento e consumo de memória. Mais tarde, o algoritmo A * (HART, 1968) foi trazido ao problema do roteamento em circuitos integrados, por Rubin (1974), resultando em uma versão melhorada do *Maze Router*. Mikami et al. (1968) propôs um roteador de busca em linha, que é mais rápido que os roteadores de busca por *maze*, mas não garante o resultado ótimo. Finalmente, Hetzel (1998) propôs uma técnica que consiste em rotular um intervalo inteiro de vértices, em vez de apenas um vértice, como o algoritmo de Rubin funciona. Assim, o algoritmo resultante é equivalente ao algoritmo A* com expansões realizadas por meio de segmentos, o que é mais rápido que qualquer algoritmo de busca por *maze* e ainda é ótimo. Como não existem trabalhos que propõe melhorias no algoritmo Hetzel, este pode ser considerado o estado da arte dos algoritmos de roteamento de propósito geral.

Apesar do algoritmo de Hetzel ser mais rápido do que os algoritmos de busca por *maze*, ele possui um problema. Ao rotear uma área congestionada, o algoritmo se comporta da mesma maneira que o algoritmo A *, aumentando muito o tempo de processamento e o consumo de memória. Diante desse problema, este trabalho propõe um novo algoritmo de roteamento de propósito geral. O termo “propósito geral” indica que o algoritmo não possui uma aplicação específica, pois pode ser aplicado em qualquer situação que se encaixe em seu escopo. O algoritmo trabalha sobre um plano modelado por um grafo de grade. Por trabalhar em um escopo bidimensional, sua aplicação mais adequada e útil é na visualização de circuitos. Ainda assim, é possível utilizá-lo na síntese física, desde que se restrinja ao uso de duas camadas de roteamento. O algoritmo utiliza uma busca em linha guiada por nós com custos associados, da mesma forma que o algoritmo de Hetzel, porém, possui características muito específicas.

2. METODOLOGIA

2.1 Exemplo Motivador e Heurística Utilizada

Consideremos um caso em que a área de roteamento está congestionada (Fig. 1 (a)). Se o retângulo tracejado é completamente interceptado por um obstáculo, o caminho deve contornar o obstáculo para alcançar o destino. Para

fazer isso, o algoritmo de Hetzel expande a área de busca em todas as direções, visitando todos os pontos da área em cinza claro. No entanto, como o destino se encontra à direita, neste caso, não seria necessário realizar a busca em todas as direções. Bastaria apenas que, o algoritmo, ao encontrar o obstáculo, avaliasse as duas possibilidades de contorno, optando pela melhor escolha. Essa é a estratégia básica do algoritmo proposto. É evidente que o espaço de busca da heurística do algoritmo proposto (nós em cinza escuro) é muito menor do que o espaço de busca do algoritmo comparado. Além disso, o custo de acesso aos nós cinza escuro não rotulados é menor que o custo dos nós rotulados, uma vez que estes representam nós no conjunto de nós abertos (também chamado de fronteira) e os não rotulados são apenas os nós que precisam ser visitados para atingir o limite do obstáculo. Todos os nós em cinza claro são nós da fronteira do algoritmo de Hetzel.

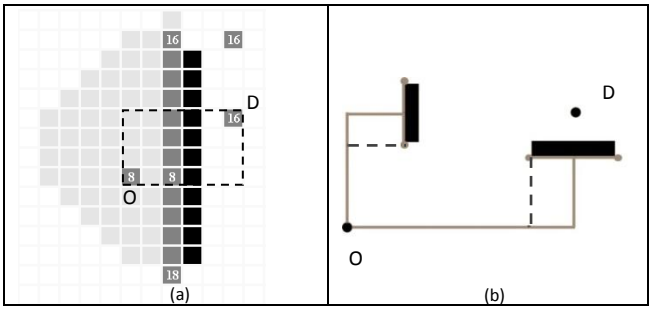

Figura 1. (a) Exemplo motivador. (b) Possíveis atalhos em linhas tracejadas.

O exemplo da Fig. 1 (a) é apenas um caso dentre muitos outros possíveis. Se o algoritmo atuasse exatamente como a heurística apresentada, ele ainda estaria longe de ser ótimo. A Fig. 1 (b) mostra um exemplo em que o caminho ótimo não é obtido. Para evitar estes casos, é necessário que haja um mecanismo de criação de atalhos. Cada vez que um segmento é criado para contornar um obstáculo, o algoritmo de detecção de atalhos é executado. Para detectar um atalho o algoritmo percorre os segmentos do caminho criando possíveis segmentos de atalho. Em seguida, o algoritmo escolhe o segmento mais promissor e tenta criar um atalho com o segmento escolhido. Se o atalho não intercepta algum obstáculo, o caminho é atualizado com o atalho encontrado. Se o atalho intercepta algum obstáculo, o algoritmo de roteamento proposto é chamado recursivamente passando como parâmetros os pontos de origem e destino do segmento de atalho. O caminho retornado é adicionado como atalho. Até então, a heurística de atalhos utilizada toma todos os cuidados para que o caminho seja ótimo. No entanto, existem raros casos onde existe algum atalho possível e o algoritmo retorna sem detectá-lo. Esta peculiaridade existe de forma intencional, pois permite um bom ganho de desempenho em troca de uma perda de qualidade praticamente desprezível.

2.2 Algoritmo de Roteamento Proposto

O algoritmo proposto trabalha com nós, associados a pontos. Cada nó tem um custo, calculado da mesma forma como o algoritmo A *. Além disso, temos uma direção (para cima, baixo, esquerda, direita), que representa a direção de expansão. Existem dois tipos de nós: nós comuns e nós de contorno. Um nó comum expande criando um segmento que se inicia no ponto do nó e termina no ponto mais próximo do destino. Se o segmento intercepta um obstáculo, dois nós de contorno são criados no ponto de intersecção, com sentidos opostos, ambos

ortogonais em relação ao nó comum. Caso contrário, um nó comum é criado, com direção ortogonal, apontando para o destino. Nós de contorno possuem outra direção associada, que é a direção do obstáculo que se deseja contornar. Quando expandido, um nó de contorno desloca-se na direção de expansão, até o limite do obstáculo ser atingido ou outro obstáculo ser encontrado. Em ambos os casos, um novo nó de contorno (e, possivelmente, um nó comum) é criado. Toda vez que um novo nó é criado, seja por expansão de um nó comum ou de um nó de contorno, o algoritmo de detecção de atalhos deve ser executado.

3. RESULTADOS E DISCUSSÃO

Os experimentos foram realizados em um computador com um processador Intel Core i5 3.2 Ghz e memória de 8GB DDR3 1333Mhz. O algoritmo proposto, denominado ST-Router, e o algoritmo de Hetzel foram implementados em Java. Ambos os algoritmos foram integrados ao visualizador de circuitos de uma ferramenta de CAD desenvolvida em nosso grupo de pesquisa. O posicionador de células do visualizador de circuitos foi utilizado para fornecer os obstáculos iniciais de roteamento. O posicionador utilizado tende a fornecer áreas de roteamento com alto congestionamento, para grandes circuitos. O roteamento foi realizado para um subconjunto dos *benchmarks* MCNC (LISANKE, 1988). Para cada fio, ambos algoritmos realizaram o roteamento sobre o mesmo conjunto de obstáculos. Após ambos algoritmos rotearem um fio, o caminho resultante do ST-Router foi adicionado ao conjunto de obstáculos. Para cada *benchmark*, o processo de roteamento foi realizado 5 vezes a fim de obter o tempo médio de execução e desvio padrão para ambos os algoritmos.

A Tab. 1 apresenta os resultados. A coluna “WL” representa quanto o comprimento total de fios do ST-Router excede o comprimento de fios do algoritmo de Hetzel. A coluna “Expandidos” mostra quantas vezes algoritmo de Hetzel expande mais nós do que o ST-Router. A coluna “Speedup” mostra quantas vezes ST-Router foi mais rápido. A relação desvio-padrão/média (em relação ao tempo de processamento) foi calculada para cada *benchmark*. A média

Tabela 1. Resultados de desempenho.

Benchmark	Tempo (ms)		WL (st/h -1)%	Speedup (h/st)
	Hetzel	ST		
cmb	213	8	0	28
frg1	490	18	0	27
ldd	7.815	43	0	180
majority	0,21	0,22	0	0,97
mux	208	8	0	25
pcler8	667	16	0	42
term1	65.327	40	0	1.631
unreg	2.897	81	0	35
x2	184	10	0	19
z4ml	24	1,04	0	23
9symml	235.419	939	0	250
alu2	2.435.382	4.466	0,009	545
apex6	13.211.168	124.922	0,008	105
C1908	1.221.586	1.215	0	1.005
C3540	49.739.358	1.508.480	0,0003	32
C432	22.172	31	0,004	710
cht	117.083	4.482	0,005	26
i2	556	10	0	57
ttt2	29.553	510	0	57
vda	2.048.551	61.456	0,001	33

dessa relação entre todos os *benchmarks* é 4,59% para o algoritmo de Hetzel e 5,85% para o ST-Router.

Os resultados mostram que o algoritmo proposto é muito mais rápido, mantendo um comprimento de fio sub-óptimo. Quanto maior é o tamanho do *benchmark*, maior tende a ser o *speedup*. Isto se deve ao fato de que, quanto maior o circuito, mais o algoritmo de Hetzel se encontra nos casos onde seu desempenho é reduzido drasticamente, enquanto o ST-Router sofre um queda de desempenho muito menor.

4. CONCLUSÕES

Este artigo apresentou um novo algoritmo de roteamento de propósito geral. O algoritmo proposto visa melhorar a forte queda de desempenho do algoritmo de Hetzel em casos frequentes. Os resultados mostram que o algoritmo proposto é extremamente mais rápido, mantendo um comprimento de fio sub-ótimo. Isso o torna mais adequado para uma escolha de algoritmo de roteamento, se o tempo é essencial e o escopo é bidimensional, como o roteamento na visualização de circuitos.

Como trabalhos futuros, pretende-se ampliar o ST-Router para o escopo tridimensional e compará-lo com o algoritmo de Hetzel, utilizando *benchmarks* de roteamento da síntese física. Além disso, pretende-se aprofundar os estudos sobre o comportamento do algoritmo proposto, a fim de fazer o algoritmo garantir o melhor caminho, mantendo o seu desempenho.

5. REFERÊNCIAS BIBLIOGRÁFICAS

HART, P. E. A Formal Basis for the Heuristic Determination of Minimum Cost Paths. In: **IEEE TRANS. ON SYST. SCI. AND CYBERN.** 1968. p. 100-107.

HETZEL, A. A sequential detailed router for huge grid graphs. In: **DESIGN, AUTOMATION AND TEST IN EUROPE.** 1998. p. 332-338.

LEE, C. Y. An Algorithm for Path Connections and Its Applications. In: **IRE TRANS. ON ELECTRONIC COMPUTERS.** 1961. p. 346-365.

LISANKE, R. **Logic synthesis and optimization benchmarks User Guide Version 3.0**. 1988. Acessado em 29 jul. 2014. Online. Disponível em: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.49.591&rep=rep1&type=pdf

MIKAMI, K.; TABUCHI, K. A computer program for optimal routing of printed circuit connectors. In: **PROC. INT. FEDERATION FOR INFORM. PROCESS**. 1968. p. 1475-1478.

RUBIN, F. The Lee Path Connection Algorithm. In: **IEEE TRANS. ON COMPUT**. 1974. p. 907-914.