ALGORITMO ESTRUTURAL PARA GERAÇÃO DE REDES DE TRANSISTORES EFICIENTES

POSSANI, Vinicius Neves; MARQUES, Felipe de Souza;
DA ROSA JUNIOR, Leomar Soares

Universidade Federal de Pelotas - UFPel
{vnpossani, felipem, leomarjr}@inf.ufpel.edu.br

1. INTRODUÇÃO

No desenvolvimento de circuitos integrados (CIs), o número total de transistores necessários para implementar cada célula lógica, está diretamente relacionados com o atraso de propagação do sinal, consumo de potência e área dos circuitos integrados Muitos dos CIs desenvolvidos atualmente são compostos por bilhões, o que demanda o uso de ferramentas *Computer Aided Design (CAD)* para automatizar o fluxo de projeto.

Atualmente existem alguns métodos disponíveis na literatura que implementam diferentes técnicas para geração de redes de transistores otimizadas. Os mais tradicionais são baseados em fatorações a (MARTINS, 2012). Outros métodos são baseados em otimizações através de grafos. Inicialmente, uma expressão Booleana é traduzida para um grafo, em sequência são realizadas operações no grafo visando compartilhar arestas (DA ROSA JUNIOR, 2007) (KAGARIS 2007).

As técnicas existentes para geração de redes de transistores normalmente utilizam estratégias gulosas. Porém, sabe-se que os algoritmos gulosos estão diretamente relacionados com mínimos locais e, portanto, podem não ser a melhor estratégia. Nesse sentido, este trabalho apresenta um método para geração de redes de transistores eficientes, o qual visa evitar as escolhas gulosas no inicio do processo de otimização. De fato, é de extrema importância evitar as escolhas gulosas durante os primeiros passos do processo de otimização, pois tais escolhas apresentam grande influencia em otimizações futuras. Consequentemente, essas escolhas estão diretamente relacionadas à qualidade da rede de transistores gerada ao final do processo.

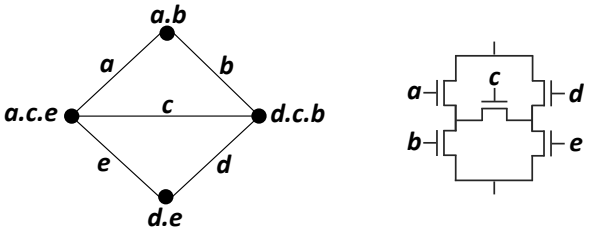
2. METODOLOGIA

O método proposto parte de uma Soma de Produtos Irredundante (ISOP) e realiza combinações C_4^n dos cubos da ISOP com o objetivo de construir estruturas de grafos predefinidas, as quais foram chamadas de *NSP Kernels* e *SP Kernels*. Tais estruturas estão relacionados, respectivamente, a associações Não-Série-Paralelas (NSP) e associações Série-Paralelas (SP) de transistores. Dessa forma, é possível obter arranjos eficientes em número de chaves sem realizar escolhas gulosas. Para cada combinação, o algoritmo tenta construir um grafo onde os vértices representam cubos da função de entrada e as arestas representam literais em comum entre cada par de cubo. Posteriormente, os rótulos das arestas dos *kernels* são mapeados para uma rede de transistores através de um processo de reposicionamento das arestas.

Exemplo NSP Kernel: Considere a Equação (1) como entrada do método proposto. Os cubos da função serão representados através dos vértices do grafo ilustrado pela Fig. 1(a) e as intersecções entre os cubos são calculadas através das arestas do grafo. Note que todos os cubos compartilharam todos os seus literais através das arestas do grafo. Assim, aplicando um reposicionamento nas

arestas do grafo é possível obter a rede de transistores não-série-paralela mínima ilustrada pela Fig. 1(b), sem aplicar escolhas gulosas.

$$f = a.b + a.c.e + d.e + b.c.d \quad (1)$$

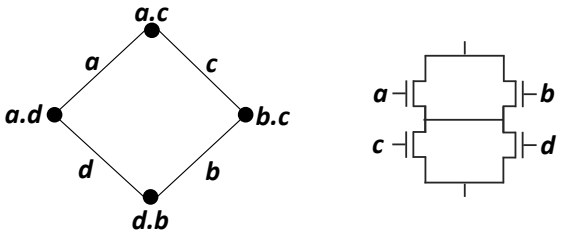


(a) (b)

Figura 1. *NSP Kernel* (a) e rede de transistores obtida (b) através da Equação (1).

Exemplo SP Kernel: Considerando como entrada do método a Equação (2), o algoritmo constrói o grafo ilustrado pela Fig. 2(a) verificando a intersecção entre os cubos através das arestas do grafo. A Fig. 2(b) apresenta a rede série-paralela mínima obtida pelo processo de reposicionamento de arestas.

$$f = a.c + a.d + b.c + b.d \quad (2)$$



(a) (b)

Figura 2. *SP Kernel* (a) e rede de transistores obtida (b) através da Equação (2).

Os exemplos anteriores são bastante simples, devido as expressões de entrada serem compostas por poucos cubos. Porém, para funções mais complexas, o algoritmo é capaz de encontrar diversos *NSP* e/ou *SP kernels*. Assim, os *kernels* encontrados são compostos através de uma técnica de compartilhamento de arestas previamente desenvolvida (POSSANI, 2012). Dessa forma, as subestruturas mais otimizadas são encontradas no inicio do processamento através dos *NSP* e *SP kernels* e, por fim, são compostas para atingir a solução final para a função alvo.

3. RESULTADOS E DISCUSSÃO

O primeiro experimento foi realizado sobre o conjunto de funções P-class de quatro entradas, composto por 3982 funções lógicas. Assim, foram geradas portas lógicas para cada uma das funções deste conjunto, e comparadas com as soluções de outros métodos da literatura. A Tabela 1 apresenta os resultados obtidos considerando o número total de chaves para implementar portas lógicas para todas as funções.

O segundo experimento foi realizado através de um conjunto de 53 funções lógicas. Esse conjunto de funções foi escolhido devido as características dessas funções, e por consequência, a dificuldade de encontrar as redes de transistores mínimas para implementar tais funções. Assim, cada uma das 53 funções foi

aplicada aos métodos propostos e aos outros métodos presentes na literatura. Os resultados obtidos através de cada método foram comparados com a solução mínima de cada rede disponível no catálogo (LOGICS, 2012). Os dados obtidos através desse experimento estão descritos na Tabela 2.

Tabela 1. Número total de transistores para gerar portas lógicas para o conjunto de funções lógicas P-class de 4 entradas.

	(MARTINS, 2010)	(DA ROSA JUNIOR, 2007)	(KAGARIS, 2007)	***Método proposto***
Número total de transistores	102.668	103.049	97.174	**95.595**

Tabela 2. Número total de transistores para implementar as redes do catálogo de 53 funções lógicas.

	Ótimo	(MARTINS, 2010)	(DA ROSA JUNIOR, 2007)	(KAGARIS, 2007)	***Método proposto***
Número total de transistores	**356**	487	503	543	**359**

Por fim, o último experimento realizado foi através do catálogo de funções e redes presente em (HARRISON, 1965). Esse catálogo é composto por 402 funções e suas respectivas redes desenhadas manualmente, consideradas pelo autor como mínimas. Ao longo das últimas décadas, diversos autores utilizaram esse catálogo como métrica para avaliação dos métodos de geração de redes propostos. Recentemente, Tanaka e Kagaris compararam o número de chaves das redes geradas pelos seus respectivos métodos (TANAKA, 2004) e (KAGARIS, 2007) com o número de chaves das redes do catálogo de (HARRISON, 1965).

O método proposto não foi capaz de gerar a rede mínima para todas as funções do catálogo. Porém, a principal contribuição do algoritmo propostos em relação a esse experimento, foi o fato de que o método encontrou duas redes com um número menor de chaves do que as respectivas redes das funções N 12' e N 54 do catálogo de Harrison. A função N 12' foi implementada no catálogo de Harrison pela rede da Fig. 3(a). No entanto, a Fig. 3(b) ilustra a rede mínima derivada pelo método proposto. De forma similar, a função N 54, foi implementada no catálogo de Harrison pela rede ilustrada na Fig. 4(a). Porém, a Fig. 4(b) ilustra a solução encontrada pelo método proposto.

Assim, além de demonstrar que nem todas as redes do catálogo de Harrison são mínimas (como Tanaka e Kagaris demonstraram para uma delas), o métodos proposto neste trabalho demonstrou que nem o método de (TANAKA, 2004) e nem o método de (KAGARIS, 2007) foram capazes de encontrar essas duas redes mínimas. Já que, para as funções N 54 e N 12' ambos os métodos derivaram redes com o mesmo número de chaves do que as redes correspondentes no catálogo de (HARRISON, 1965).

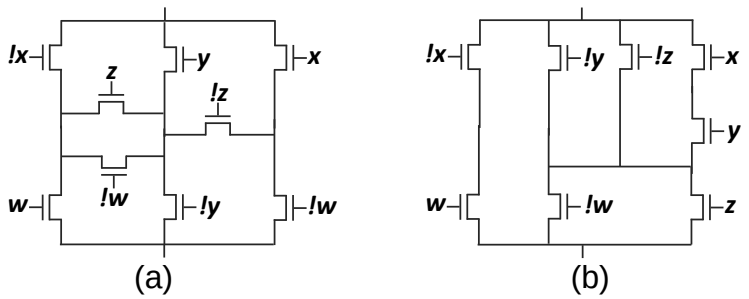
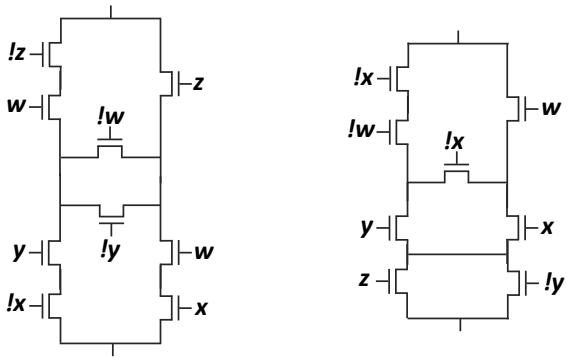


Figura 4. Redes para N 12': (a) Harrison e (b) método proposto (mínima).



(a) (b)

Figura 4. Redes para N 54: (a) Harrison e (b) método proposto (mínima).

4. CONCLUSÕES

Este trabalho apresentou um método para geração automática de redes de transistores otimizadas. O algoritmo desenvolvido é capaz de evitar escolhas gulosas durante o inicio do processo de geração da rede, o que contribui significativamente para que redes mais eficientes sejam obtidas. A qualidade dos resultados obtidos através do conjunto de experimentos realizados demonstra o impacto positivo de evitar as escolhas gulosas.

5. REFERÊNCIAS BIBLIOGRÁFICAS

MARTINS, M. G. A.; DA ROSA JUNIOR, L. S.; RASMUSSEN, A.; RIBAS, R. P. e REIS, A. I.. Boolean Factoring with Multi-Objective Goals. **IEEE Int. Conf. on Computer Design.** Amsterdam, 2010. p. 229-234.

DA ROSA JUNIOR, L. S.; MARQUES, F. S.; SCHNEIDER, F.; RIBAS, R. P. e REIS, A. I. A Comparative Study of CMOS Gates with Minimum Transistor Stacks. **20th ACM Symposium on Integrated Circuits and Systems Design (SBCCI)**. Rio de Janeiro, 2007. p. 93-98.

KAGARIS, D. e HANIOTAKIS, T. A Methodology for Transistor-Efficient Supergate Design. **IEEE Trans. On Very Large Scale Integration (VLSI) Systems**, p. 488-492, 2007.

POSSANI, V. N.; DE SOUZA, R. S.; DOMINGUES JUNIOR, J. S.; AGOSTINI. L. V.; MARQUES, F. S. E DA ROSA JUNIOR, L. S. Optimizing Transistor Networks Using a Graph-Based Technique. **Analog Integrated Circuits and Signal Processing**, v. 73, p. 841-850, May 2012.

LOGICS. Re-trieved October 15, 2012, from Logic Circuit Synthesis Labs, Federal University of Rio Grande do Sul. **Catalog of 53 Handmade Optimal Switch Networks**, Porto Alegre, 15 October 2012. Disponivel em: <http://minerva.ufpel.edu.br/~leomarjr/53_NSP_Catalog.pdf>. Acesso em: 28 Agosto 2013.

HARRISON, M. **Introduction to Switching and Automata Theory**. New York: McGraw-Hill, 1965.

TANAKA, K. e KAMBAYASHI, Y. Transduction method for design of logic cell structures. **Proc. IEEE Asia–Pacific Design Autom. Conf. (ASP-DAC)**. 2004. p. 600–603.