BRANCH AND BOUND APLICADO NO PROBLEMA DE COBERTURA DE FUNÇÕES BOOLEANAS

Tercio Alberto dos Santos[1], Alexandre César Rodrigues da Silva[2] e Carlos Eduardo da Silva Santos[3]

Resumo — *Este trabalho apresenta um método para a obtenção de cobertura mínima para uma função booleana empregando o algoritmo Branch and Bound modificado. A partir da formulação do problema de cobertura de funções booleanas como um problema de programação linear inteira 0 e 1, cuja formulação é apresentada a seguir, a solução ótima global é obtida.*

MIN CT. X de modo que

$$AX \geq b$$
$$X \geq 0$$

Onde: $X \in Rn$; $C \in Rn$; $b \in Rn$ e $A \in R^{n.n}$ *com posto completo. Vários métodos para a geração de implicantes primos podem ser utilizados para a geração da formulação da cobertura. Empregou-se neste trabalho o algoritmo Expander que utiliza-se do método de expansão de Shannon. Uma grande variedades de funções booleanas foram estudadas, cujos resultados comprovaram a eficiência computacional deste tipo de abordagem. O algoritmo implementado em linguagem de programação está sendo comparado com outros algoritmos já implementados.*

Palavras chave — Programação Matemática, Otimização de Funções Booleanas, Branch and Bound, Plano de Corte, Dual Simplex.

INTRODUÇÃO

A otimização lógica a dois níveis tem suas raízes há muitas décadas atras e está baseada na teoria clássica sobre álgebra de chaveamento. Vários textos tratam do assunto [1], [2], [3]. Os primeiros algoritmos para a minimização exata foram propostos por Quine [4] e por McCluskey [2]. Uma grande variedade de métodos foram propostos ao longo dos anos como, por exemplo, o método de Rudell e Sangiovanni [5] denominado Espresso-Exact, o método de Dagenais et ali [6] denominado de McBoole, o método de Coudert e Madre [7] e muitos outros.

Com o aumento da complexidade dos circuitos digitais a minimização lógica exata tornou-se impraticável até algumas décadas atrás. Com o aumento da velocidade e da capacidade de memória das estações de trabalho, aliado ao desenvolvimento de eficientes métodos de cobertura para uma solução exata, revitalizou-se o interesse por essa área de pesquisa.

Atualmente, tem-se como um dos mais eficientes algoritmos de minimização de funções booleana o programa denominado BOOM (A Heuristic Boolean Minimizer) [8], cuja principal característica é a combinação da geração dos implicantes primos em conjunto com a solução do problema de cobertura, o que leva a uma redução no número de implicantes primos gerados.

Este trabalho trata sobre a resolução do problema de cobertura de funções booleanas formulada como um problema de programação matemática. Implementou-se três algoritmos descritos na literatura com o objetivo de avaliar o que melhor soluciona o problema de cobertura. Foram implementados o algoritmo Branch and Bound, o Plano de Corte de Gomory e o Dual Simplex. Utilizou-se como critério de eficiência a quantidade de memória utilizada, o tempo de execução, a quantidade de iterações e a solução obtida.

PROBLEMA DE COBERTURA

Toda cobertura mínima de uma função booleana é solução de um problema de programação linear inteira 0 e 1, dessa forma todos os avanços da área da programação matemática podem ser utilizados com o objetivo de solucioná-la. Muitos métodos clássicos podem ser empregados, porém nesse trabalho implementou-se três métodos, o Branch and Bound, o Plano de Corte de Gomory e o Dual Simplex Canalizado [9].

O método Simplex é a base para o Plano de Corte de Gomory e para o Branch and Bound. O Dual Simplex Canalizado é uma forma alternativa para solucionar o problema de cobertura sem a necessidade de se utilizar o método Simplex.

Resolver o problema de cobertura consiste em encontrar uma solução que satisfaça todas as restrições do problema e que o custo da função objetivo, que representa o custo de implementação da função booleana, seja o menor possível.

Na grande maioria dos exemplos tratados neste trabalho obteve-se solução inteira no conjunto binário 0 e 1. Contudo, em alguns casos encontrou-se uma solução otimista (casos cíclicos). A identificação de casos cíclicos tornou-se possível, pois a solução é fracionária. Para esses casos empregou-se o método Branch and Bound ou Plano de Corte

[1]Tercio Alberto dos Santos, FESURV Fundação do Ensino Superior de Rio Verde , Fazenda Fontes do Saber, 75901-970, tercioas@fesurv.br, tecioas@dee.feis.unesp.br

[2]Alexandre César Rodrigues da Silva, UNESP Universidade Estadual Paulista, Júlio de Mesquita Filho, Faculdade de Engenharia, Av. Brasil, Nº 56 Centro, 15385-000, Ilha Solteira S.P. Brasil, acrsilva@dee.feis.unesp.br

[3]Carlos Eduardo da Silva Santos, UNESP Universidade Estadual Paulista, Júlio de Mesquita Filho, Faculdade de Engenharia, Av. Brasil, Nº 56 Centro, 15385-000, Ilha Solteira S.P. Brasil , carloses@dee.feis.unesp.br

de Gomory para encontrar a solução ótima global do problema.

O algoritmo de Branch and Bound é um algoritmo enumerativo, cuja estrutura de resolução baseia-se na construção de uma árvore onde os nós representam os problemas candidatos e os ramos representam as novas restrições que devem ser consideradas. Por intermédio dessa árvore, todas as soluções inteiras da região viável do problema são enumeradas de modo implícito ou explicito o que garante que todas as soluções ótimas serão encontradas.

O algoritmo de Plano de Corte [10] é um dos principais algoritmos de programação linear inteira, o seu uso é pós otimização de PL internamente. Nesse algoritmo de uma iteração para outra resolve-se um PL com uma restrição adicional. Assim, os PL's sucessivos são diferentes em uma restrição adicional. Portanto, a estratégia mais evidente é usar dual simplex que aproveita o quadro ótimo de um PL para resolver o PL seguinte com a simples adição no quadro da nova restrição corretamente atualizada para a base atual.

O algoritmo dual simplex canalizado é o mais adequado para reotimizar subproblemas de programação linear (PL) gerados pelos algoritmos de Branch and Bound na resolução de um problema de programação linear inteira.

No algoritmo de Branch and Bound cada problema (PL) é diferente de seu antecessor em uma restrição adicional da seguinte forma:

$$x_j \geq k + 1 \qquad (\beta_1)$$

ou

$$x_j \leq k \qquad (\beta_2)$$

Sendo k um número inteiro.

Embora as restrições do tipo β_1 e β_2 podem ser adicionadas como uma restrição padrão e usar um algoritmo dual simplex padrão, essa estratégia aumenta sem necessidade o tamanho do quadro e da base. Como as restrições β_1 e β_2 são restrições sobre uma única variável então a melhor estratégia é usar um algoritmo dual simplex canalizado que leva em conta esse tipo de restrições somente de maneira implícita.

Vários métodos podem ser empregados para a formulação do problema de cobertura. Neste trabalho utilizou-se o algoritmo Expander [11] que implementa Método de Expansão de Shannon.

O Método de Expansão de Shannon

A geração de implicantes primos pelo Método de Expansão de Shannon é uma técnica iterativa que permite obter implicantes primos de uma dada função boolena através de simples operações sobre o conjunto de mintermos e/ou estados "don't care", representados em notação decimal, que descreve a função original. Tal método pode ser empregado em função com um grande número de variáveis. Todos ou, quase todos os implicantes primos que não estariam no conjunto solução são eliminados, simplificando, desse modo, a busca pela solução mínima.

Como exemplo considere a função booleana $f(A,B,C,D) = \Sigma m(0, 6, 8, 10, 15) + d(1, 2, 7, 9, 11, 14)$. Para a formulação do problema de cobertura utilizou-se o algoritmo Expander cujo arquivo de entrada e de saída estão apresentados na Figura 1 e na Figura 2, respectivamente. A tabela de cobertura proposta por McCluskey, consiste de colunas contendo todos os mintermos da função e de linhas que representam todos os implicantes primos gerados. A Tabela I apresenta a tabela de cobertura de McCluskey cujos implicantes primos foram gerados pelo algoritmo Expander.

Com o auxílio da Tabela I, o problema de cobertura dos mintermos transforma-se em um problema de programação linear inteiro binário 0 e 1.

FIGURA. 1
TELA DO ARQUIVO DE ENTRADA DO PROGRAMA EXPANDER.

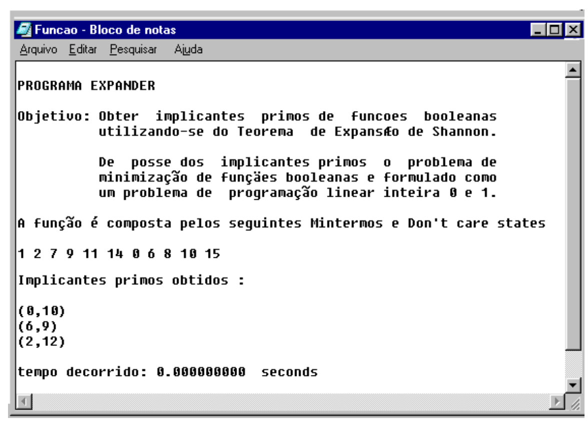

FIGURA. 2
TELA DO ARQUIVO DE SAÍDA DO PROGRAMA EXPANDER.

TABELA I
TABELA DE COBERTURA DE MCCLUSKLEY CUJOS IMPLICANTES PRIMOS FORAM GERADOS PELO ALGORITMO EXPANDER

	Mintermos				
Implicantes	0	6	8	10	15
XX10		x		x	
1X1X				x	x
X11X		x			x
X00X	x		x		

Optou-se por tratar o problema de cobertura como um problema matemático. Dessa forma, empregou-se uma notação matemática mais simples.

Fazendo-se: $x_1 = XX10$, $x_2 = 1X1X$, $x_3 = X11X$ e $x_4 = X00X$ o problema apresentado na Tabela I é formulado como um problema de programação linear inteira 0 e 1 como descrito a seguir [12]:

$$
\begin{array}{l}
\min x_0 = 3x_1 + 3x_2 + 3x_3 + 3x_4 \\
\text{s.a.} \\
\begin{array}{llllll}
x_1 & + x_2 & & & \geq 1 \\
x_1 & & + x_3 & & \geq 1 \\
 & + x_2 & + x_3 & & \geq 1 \\
 & & & + x_4 & \geq 1
\end{array} \\
x_1, x_2, x_3, x_4 \in \{0, 1\}
\end{array}
$$

Na resolução deste problema de programação linear usou-se uma linguagem de programação [13] MATLAB 5.3, que foi implementado em duas etapas:

a) Fez-se um programa para resolver o problema pelo método Simplex e obteve-se uma solução otimista, apresentado na Tabela II.

TABELA II
SOLUÇÃO OTIMISTA PELO MÉTODO SIMPLEX

7.5000	-1.5000	-1.5000	-1.5000	-3.0000
0.5000	-0.5000	-0.5000	0.5000	0
0.5000	-0.5000	0.5000	-0.5000	0
0.5000	0.5000	-0.5000	-0.5000	0
1.0000	0	0	0	-1.0000

O valor da função objetivo é igual a 7.5000, os valores da coluna abaixo do valor da função objetivo são os valores das variáveis, ou seja, {(0.5000, 0.5000, 0.5000, 1.0000)};

b) A solução da primeira etapa é otimista, então, implementou-se o programa (Branch2) empregando o dual simplex canalizado, assim, dando seqüência na resolução do método Branch and Bound.

No desenvolvimento do problema tem-se dois subproblemas P_1 e P_2.

Resolução do subproblema P_1

Para resolver o subproblema P_1 acrescentou-se uma restrição representado na Tabela III.

TABELA III
SOLUÇÃO OTIMISTA COM ACRÉSCIMO DA RESTRIÇÃO

7.5000	-1.5000	-1.5000	-1.5000	-3.0000
0.5000	-0.5000	-0.5000	0.5000	0
0.5000	-0.5000	0.5000	-0.5000	0
0.5000	0.5000	-0.5000	-0.5000	0
1.0000	0	0	0	-1.0000
-0.5000	-0.5000	0.5000	0.5000	0

Implementou-se o programa Dual Simplex Canalizado e encontrou-se como solução ótima global da função booleana apresentada na Tabela IV.

O valor da função objetivo é igual a 9 e os valores da coluna abaixo do valor da função objetivo são os valores das variáveis, ou seja, o conjunto verdade {(1, 0, 1, 1, 1)}.

TABELA IV
SOLUÇÃO ÓTIMA DO SUBPROBLEMA P_1

9	0	-3	-3	-3
1	0	-1	0	0
0	-1	1	0	0
1	1	-1	-1	0
1	0	0	0	-1
1	1	-2	-1	0

Resolução do subproblema P_2

Para resolver o subproblema P_2 adicionou-se uma nova restrição e implementou-se o programa Dual Simplex Canalizado e obteve-se como solução ótima do problema de cobertura mínima da função booleana f(A,B,C,D) = Σm(0, 6, 8, 10, 15) + d(1, 2, 7, 9, 11, 14) representado na Tabela V.

TABELA V
SOLUÇÃO ÓTIMA DO SUBPROBLEMA P_2

9	-3	-3	-3	-3
0	0	0	1	0
1	-1	0	-1	0
1	0	-1	-1	0
1	0	0	0	-1
1	-1	-1	2	0

A solução ótima do subproblema P_2, é o valor da F.O = 9 (função objetivo) e os valores das variáveis é o conjunto verdade {(0, 1, 1, 1, 1)}.

Observa-se que em ambos os subproblemas P_1 e P_2 o custo da função booleana f(A,B,C,D) = Σm(0, 6, 8, 10,15) + d(1, 2, 7, 9, 11, 14) é F.O. = 9.

Outros algoritmos foram implementados neste Exemplo na linguagem de programação MATLAB 5.3 e o valor obtido em ambos os casos foi o mesmo para função booleana f(A,B,C,D) = Σm(0, 6, 8, 10, 15) + d(1, 2, 7, 9, 11, 14).

Implementação do Algoritmo Dual do Simplex

Partindo-se do quadro otimista implementa-se o algoritmo Dual Simplex na linguagem de programação (MATLAB 5.3), e obtém-se como resultado:

FO = 9, X2 = 1, X3 = 1, X7 = 1 e X4 = 1

Assim, a solução ótima global do problema de cobertura da função booleana f(A,B,C,D) = Σm(0, 6, 8, 10,15) + d(1, 2, 7, 9, 11, 14) é dada por: custo mínimo igual a 9, e o conjunto verdade {(0, 1, 1, 1, 0, 0, 1)}, embora em ambos casos os conjuntos verdades sejam diferentes, porém, o custo é o mesmo.

Implementação do Algoritmo Plano de Corte ou Algoritmo de Gomory.

Partindo-se do quadro otimista da Tabela II encontrado pelo algoritmo Simplex, e acrescenta-se uma restrição de corte tem-se a Tabela VI, para depois implementar o algoritmo de Gomory (Gomo2) na linguagem de programação MATLAB 5.3.

TABELA VI
ACRÉSCIMO DE UMA NOVA RESTRIÇÃO NO QUADRO OTIMISTA

7.5000	-1.5000	-1.5000	-1.5000	-3.0000
0.5000	-0.5000	-0.5000	0.5000	0
0.5000	-0.5000	0.5000	-0.5000	0
0.5000	0.5000	-0.5000	-0.5000	0
1.0000	0	0	0	-1.0000
-0.5000	0.5000	-0.5000	-0.5000	0

Implementando o algoritmo de Gomory (Gomo2) tem-se a apresentação da Tabela VII a qual apresenta a solução ótima do problema.

TABELA VII
SOLUÇÃO ÓTIMA PELO MÉTODO DE GOMORY

9	-3	0	0	-3
1	-1	0	1	0
1	-1	1	0	0
0	1	-1	-1	0
1	0	0	0	-1
1	-2	1	1	0

A função objetivo FO = 9 e os números que aparecem na coluna abaixo do número 9 são os valores das variáveis, ou seja, o conjunto verdade {(1, 1, 0, 1, 1)}.

Comparando-se as três implementações obteve-se o conjunto de soluções diferentes, porém com o mesmo custo, ou seja, em todos os casos a solução mínima global da função booleana $f(A,B,C,D) = \Sigma m(0, 6, 8, 10,15) + d(1, 2, 7, 9, 11, 14)$ tem o custo igual a 9. Assim, obteve-se parâmetros para os três algoritmos implementados para relacionar custos, números de iterações, tempo computacional e quantidades de memórias, apresentados na Tabela VIII. Os resultados obtidos são:

TABELA VIII
APRESENTAÇÃO DA ANÁLISE DOS ALGORITMOS BRANCH2, GOMO2 E DUAL

Program	Custo	Iteração	Tempo(seg)	Mem.(bytes)
Branch2	9	6	2,56	2.772
Gomo2	9	5	2,09	2.448
Dual	9	3	0,77	1.780

As Figuras 3, 4 e 5, respectivamente, apresentam a relação gráfica das iterações, tempo gasto em segundos e memória em bytes dos três algoritmos implementados.

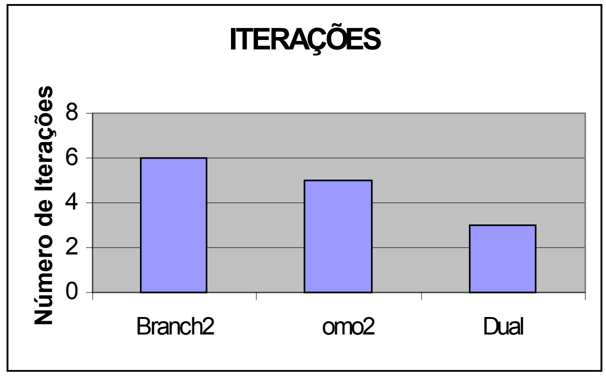

FIGURA. 3
APRESENTA AS ITERAÇÕES GASTAS PELOS PROGRAMAS BRANC2, GOMO2 E DUAL.

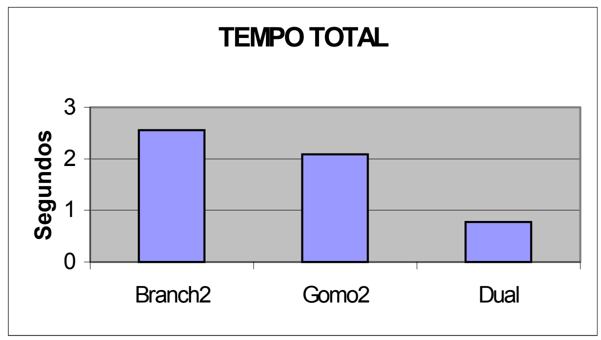

FIGURA. 4
APRESENTA O TEMPO GASTO PELOS PROGRAMAS BRANC2, GOMO2 E DUAL.

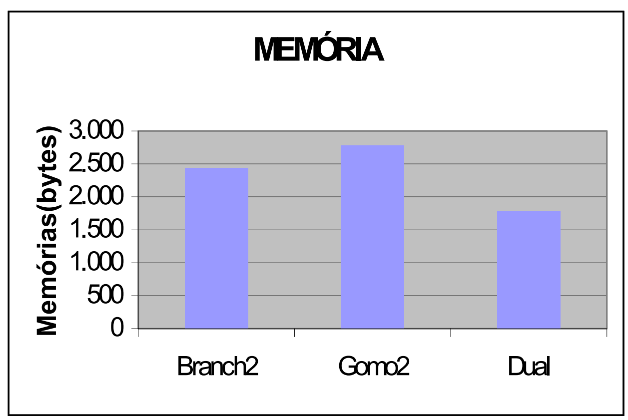

FIGURA. 5
APRESENTA A QUANTIDA DE MEMÓRIAS GASTA PELOS PROGRAMAS BRANC2, GOMO2 E DUAL.

CONCLUSÃO

Os circuitos elétricos são representados matematicamente pelas funções booleanas e neste trabalho revolveu-se a função booleana f(A,B,C,D) = Σm(0, 6, 8, 10,15) + d(1, 2, 7, 9, 11, 14) como exemplo. Através do algoritmo Expander, que utiliza-se do método de expansão de Shannon, gerou-se os implicantes primos as funções e o problema de cobertura foi modelado como um problema de programação matemática.

Visto que todo problema de cobertura mínima de uma função booleana é solução de um problema de programação linear inteira 0 e 1, implementou-se o algoritmo de Branch and Bound e encontrou-se a solução ótima global do problema de cobertura mínima.

Comparando-se com a implementação de outros algoritmos como Plano de Corte de Gomory e Dual Simplex observou-se que os três algoritmos implementados encontram o mesmo custo. A análise da Tabela II mostra os parâmetros entre os programas Branch2, Gomo2 e Dual quanto ao número de iterações, quantidade de memórias e o tempo em segundos gastos na execução de cada programa. Chega-se a conclusão que entre os três programas implementados o Dual leva uma ligeira vantagem, pois tem um número menor de iterações, o tempo gasto para executar o programa é menor comparando com os outros dois e a quantidade de memórias é bem menor.

Muito outros casos de cobertura estão sendo avaliados com o objetivo de validar os algoritmos implementados.

AGRADECIMENTOS

Os autores agradecem a Direção da Fundação do Ensino Superior de Rio Verde "FESURV" e a CAPES, pelo apoio irrestrito dado as pesquisas realizadas no desenvolvimento deste trabalho.

REFERÊNCIAS

[1] Hill, F. e G. Peterson, Switcihng *Theory and Logical Design, Wiley*, New York, 1981.

[2] McCluskey, E.J. *"Minimization of Boolean functions, The Bell System Technical Journal,* 35, No 5, November 1956. pp. 1417-1444.

[3] Kohavi, Z. *Switching and Finite Automata Theory,* McGraw-Hill, 1970.

[4] Quine, W. *The Problem of Simplifying Truth Functions*, American Mathematical Monthly, Vol. 59, 1952, pp. 521-531.

[5] Rudell and A. Sangiovanni-Vincentelli, Multiple - *valued Minimization for PLA Optimization,* IEEE Transactions on CAD/ ICAS, Vol. CAD-6, No. 5, September 1987, pp 727-750.

[6] Dagenais, M - Agarwal V. and Rumin, N. McBOOLE: *A New Procedure for Exact Logic Minimization,* IEEE Transactions on CAD/ICAS, Vol. CAD-5, No. 1, January 1986, pp. 229-223.

[7] Coudert, O. - Madre, J. C.: *"Implicit and incremental computation of primes and essential primes of Boolean funcitions,* In Proc. of the Desing Automation Conf. (Anaheim,CA, June 1992), pp. 36-39.

[8] Hlavička, J. and Fišer, *BOOM- a Heuristic Boolean Minimizer.* Proc. ICCAD-2001, San Jose. Cal. (USA).

[9] Bazaraa, M. S., Jarvis, J. J. and SheraliI, H. D.- *"Linear Programming and Network Flows"* - Second Edition - John Wiley & Sons, Inc. - New York - 1993.

[10] .Garfinkel, R. S. and Nehauser, G. L. "*Ĭnteger Programming*", Library of Congress Cataloging in Publicacation Data - Unite States - 1976.

[11] Emer, F.R.P, *"Formulação Matemática do Problema de Otimização de Funções Booleanas Através do Método de Expansão de Shannon"*, Dissertação de Mestrado - FEE - UNESP - Ilha Solteira- Outubro de 2002.

[12] Silva, A.C.R., *"Contribuição à Síntese de Circuitos Digitais Utilizando Programação Linear Inteira 0 e 1",* Tese de Doutorado – FEE – UNICAMP, Novembro, 1993.

[13] Santos, T.A. *"Aplicação do Método de Branch and Bound Aplicado na Cobertura de Funções Booleanas",* Qualificação de Mestrado - FEE - UNESP - Ilha Solteira- Novembro de 2002.