UNIVERSIDADE ESTADUAL DA PARAÍBA
CAMPUS VII – GOVERNADOR ANTÔNIO MARIZ
CENTRO DE CIÊNCIAS EXATAS E SOCIAIS APLICADAS
CURSO DE LICENCIATURA PLENA EM COMPUTAÇÃO

ARLICLEITON AILTON DA SILVA

AVALIAÇÃO DA APLICAÇÃO DA HEURÍSTICA DOS PESOS NA ORDENAÇÃO E REDUÇÃO DE DIAGRAMAS DE DECISÃO BINÁRIA DE ESPECIFICAÇÕES BOOLEANAS DE SISTEMAS INDÚSTRIAIS

PATOS – PB
2014

ARLICLEITON AILTON DA SILVA

AVALIAÇÃO DA APLICAÇÃO DA HEURÍSTICA DOS PESOS NA ORDENAÇÃO E REDUÇÃO DE DIAGRAMAS DE DECISÃO BINÁRIA DE ESPECIFICAÇÕES BOOLEANAS DE SISTEMAS INDÚSTRIAIS

Monografia apresentada ao Curso de Licenciatura plena em Computação da Universidade Estadual da Paraíba em cumprimento à exigência para obtenção do grau de licenciado em computação.

Orientadora: Dra Kézia De Vasconcelos Oliveira

PATOS – PB

2014

UEPB - SIB - Setorial - Campus VII

S586a Silva, Arlicleiton Ailton da
Avaliação da aplicação da Heurística dos Pesos na ordenação e redução de diagramas de decisão binária de especificações booleanas de sistemas indústriais [manuscrito] / Arlicleiton Ailton da Silva. - 2014.
46 p. : il.

Digitado.
Trabalho de Conclusão de Curso (Graduação em Computação) - Centro de Ciências Exatas e Sociais Aplicadas, Universidade Estadual da Paraíba, 2014.
"Orientação: Profa. Dra. Kézia de Vasconcelos de Oliveira, CCEA".

1. Funções Booleanas. 2. ROBDDs. 3. Heurística dos Pesos. 4. Sistemas Industriais. I. Título.

21. ed. CDD 005.3

Arlicleiton Ailton da Silva

AVALIAÇÃO DA APLICAÇÃO DA HEURÍSTICA DOS PESOS NA ORDENAÇÃO E REDUÇÃO DE DIAGRAMAS DE DECISÃO BINÁRIA DE ESPECIFICAÇÕES BOOLEANAS DE SISTEMAS INDUSTRIAIS

Trabalho de Conclusão de Curso apresentado ao Curso de Licenciatura em Computação da Universidade Estadual da Paraíba, em cumprimento à exigência para obtenção do grau de Licenciado em Computação

Aprovado em 04 de dezembro de 2014

BANCA EXAMINADORA

Kézia de Vasconcelos Oliveira Dantas
(Orientadora)

Leonardo da Costa Santos
(Examinador)

Eugênio de Carvalho Saraiva
(Examinador)

DEDICATÓRIA

Dedico a Deus, a meus pais, a minha esposa, a meu filho, a meus irmãos e a minha sobrinha, que me deram força e coragem para superar obstáculos. A minha orientadora que sempre teve presente na orientação desse trabalho e aos meus colegas da minha turma que nunca me esquecerei dos bons momentos que passamos.

AGRADECIMENTOS

A Deus, que me dá sabedoria, saúde, motivação e me fortalece quando preciso vencer obstáculos na vida.

A minha família que me dar apoio e motivação para seguir em frente com os meus objetivos.

Agradeço, também, aos meus professores de um modo geral, que contribuem para o meu processo de aprendizagem, para que eu possa crescer enquanto acadêmico e futuro profissional de licenciado em computação.

A orientadora, Dra Kézia De Vasconcelos Oliveira, que dedica seu tempo para me orientar na construção desse trabalho.

E, a todos que fazem parte do Campus VII da UEPB.

RESUMO

Muitos problemas em aplicações como desenho de sistemas digitais, testes de lógica e inteligência artificial, entre outras áreas, podem ser expressos com a representação e manipulação de funções booleanas. Dentre os vários métodos desenvolvidos para representar e manipular funções booleanas, existem os Diagramas de Decisão Binária Ordenado e Reduzido (*Reduced Ordered Binary Decision Diagrams - ROBDDs)*, que são estruturas de dados canônicas, cuja ideia geral é diminuir o tamanho do Diagrama de Decisão Binária *(Binary Decision Diagram - BDD)*. A ordenação das variáveis pode influenciar no tamanho do ROBDD, logo é fundamental uma ordenação eficiente, para se obter bons resultados. Um dos tipos de ordenação de variáveis é a estática, que estabelece a ordem destas, antes da construção do ROBDD. A heurística dos pesos é um algoritmo de ordenação estática baseado na dependência das funções de transição do sistema e, conforme esta dependência, são atribuídos pesos as variáveis que determinam a sua posição na ordenação. Neste trabalho será avaliada a utilização da heurística dos pesos na geração dos ROBDDS. Para isto, foi feito um estudo de caso composto por 3 tipos de especificações booleanas de sistemas industriais reais, utilizou - se a ferramenta visBDD para gerá-los, com o objetivo de obter ROBDDs menores.

Palavras chave: Funções Booleanas, ROBDDs, Heurística dos pesos, Sistemas Industriais.

ABSTRACT

Many problems in applications such as design of digital systems, logic tests and artificial intelligence, among other areas, can be expressed with the representation and manipulation of Boolean functions. Among the many methods developed to represent and manipulate Boolean functions, there are Binary Decision Diagrams Sort and Reduced (Reduced Ordered Binary Decision Diagrams - ROBDDs), which are canonical data structures, the general idea is to decrease the Decision Diagram size Binary (Binary Decision Diagram - BDD). The ordering of the variables can influence the ROBDD size, sorting is an efficient logo key to obtain good results. One type of variable ordering is static establishing the order of these, before the construction of ROBDD. The heuristic weights is a static sorting algorithm based on the dependence of the system transition functions and, as this dependence, weights are assigned the variables that determine its position in the ranking. This work will be evaluated using the heuristic of the weights in the generation of ROBDDS. For this was done a case study consists of 3 types of Boolean specification actual manufacturing systems used - the visBDD tool to generate them in order to obtain smaller ROBDDs.

Keywords: Boolean functions, ROBDDs, Heuristic weights, Industrial Systems.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ALGORITMOS

LISTA DE SIGLAS

BDD	Diagrama de Decisão Binária
ITE	If-Then-Else
OBDD	Diagrama de Decisão Binária Ordenado
ROBDD	Diagrama de Decisão Binária Ordenado e Reduzido

SUMÁRIO

1. INTRODUÇÃO

As funções booleanas são fundamentais para a Ciência da Computação, pois muitos problemas em aplicações como desenho de sistemas digitais, testes de lógica e inteligência artificial, entre outras áreas, podem ser expressos com a representação e manipulação de funções booleanas (BRYANT, 1986).

Existem vários métodos desenvolvidos para representação e manipulação de funções booleanas, como por exemplo, tabelas verdade, mapas de Karnaugh, soma de produtos Bryant (1986, apud Hill e Peterson, 1974) e Diagramas de Decisão Binária (*Binary Decision Diagrams* - BDDs) que são grafos acíclicos dirigidos com vértices não terminais, arcos e dois vértices terminais que permite representar e manipular funções booleanas. No entanto, nenhuma dessas representações são formas canônicas, isto é, uma determinada função pode ter várias representações diferentes. Consequentemente, a avaliação de equivalência e satisfatibilidade pode se tornar bastante difícil. (BRYANT, 1986).

Os BDDs são bastantes populares na utilização em algoritmos para síntese lógica de circuitos digitais, e de acordo com as propriedades de cada BDD diferentes algoritmos podem ser aplicados para os mais diversos fins (MARQUES, 2003). O seu uso tem expandido nas áreas projeto de sistemas concorrentes, lógica matemática e inteligência artificial. Eles apresentam características próprias que podem ser aproveitadas para reduzir o tamanho de representação e/ou seu tempo de manipulação em muitos domínios de aplicação (FLORES, 1996).

Os BDDs possuem diversas variações, as mais populares são os Diagramas de Decisão Binária Ordenado *(Ordered Binary Decision Diagrams - OBDDs)* aos quais são BDDs em que as variáveis de entrada aparecem em ordem fixa e, apenas uma vez em cada caminho do grafo. E os Diagramas de Decisão Binária Ordenado e Reduzido (*Reduced Ordered Binary Decision Diagrams - ROBDDs)* que são estruturas de dados canônicas, cuja ideia geral é diminuir o tamanho do BDD, eliminando vértices redundantes sem alterar a função representada.

Dependendo da ordenação do ROBDD pode influenciar em seu tamanho, logo é fundamental uma ordenação eficiente, para se obter bons resultados. Para isto, existe uma variedade de técnicas para ordenar um ROBDD, que são

classificadas em dois tipos: as ordenações estáticas e as dinâmicas. ordenação estática estabelece a ordem das variáveis antes da construção do BDD, exemplos: Fine Heuristic (MALIK et. al, 1988), Meta Heurística (DRECHSLER, 2002), Heurística dos Pesos (VIDAL, 2002). Ordenação dinâmica tenta ajustar a ordem durante a construção, exemplos: *Sifting* (RUDELL, 1993), *Window* (FUJITA et. al, 1991).

Os algoritmos da ordenação estática têm sido desenvolvidos analisando as funções booleanas, e encontra boas ordens, além disso, a qualidade da ordenação inicial é diretamente proporcional à reordenação (VIDAL, 2002). Entre a variedade desses algoritmos está a heurística dos pesos, que é baseado na dependência das funções de transição do sistema e, conforme esta dependência, são atribuídos pesos as variáveis, aos quais, estes pesos determinam a posição da variável na ordenação (as variáveis de maior peso são colocadas no inicio da ordenação). A heurística dos Pesos é boa para BDDs que representam máquinas de estados finitos, como circuitos sequenciais. (VIDAL, 2002).

Este trabalho tem foco na avaliação dos ROBDDs, gerados a partir dos estudos de casos sobre especificações booleanas de sistemas industriais reais, ao qual para geração desses ROBDDs na ferramenta VisBDD desenvolvida por (MEINEL et al., 2002), será utilizada a ordenação estática denominada de heurística dos pesos. Através dessa avaliação, será verificado se a aplicação da ordenação com esta heurística gerou resultados melhores, piores ou iguais em relação ao uso da ordenação default da ferramenta visBDD que é a ordenação quando inicializa a ferramenta dada em ordem cresceste das variáveis encontradas em uma determinada expressão logica.

Objetivos

Objetivo Geral

Avaliar os ROBDDS gerados a partir de especificações booleanas de sistemas indústrias reais utilizando a heurística dos pesos.

Objetivos Específicos

- Fazer uma busca na literatura acerca de técnicas de ordenação estática em ROBDDs.

- Investigar técnicas de ordenação estática, com foco na técnica heurística dos pesos.

- Aplicar a técnica de ordenação estática heurística dos pesos a estudos de casos reais.

- Comparar os resultados obtidos com outros trabalhos descritos na literatura.

2. FUNDAMENTAÇÃO TEÓRICA

As seções deste capítulo são fundamentais para o embasamento teórico deste trabalho. Este capítulo está relacionado aos seguintes conceitos: Funções Booleanas; Diagramas de Decisão Binária; Diagramas de Decisão Binária Ordenado; Diagramas de Decisão Binária Ordenado e Reduzido; Heurística dos Pesos; e Operador ITE.

1.1 FUNÇÕES BOOLEANAS

A manipulação de funções booleanas é de fundamental importância para a ciência da computação, relata Bryant (1986) que muitos problemas em aplicações como desenho de sistemas digitais, testes de lógica e inteligência artificial, entre outras áreas, podem ser expressos como uma sequência de operações em funções booleanas. Tais aplicações se beneficiariam de algoritmos eficientes para representar e manipular funções booleanas de forma simbólica.

O cálculo clássico para lidar com valores booleanos que assumem as constantes 1 para verdadeiro e 0 para falso, utiliza - se os operadores lógicos: conjunção, disjunção, negação , implicação e bi-implicação, representados respectivamente pelos símbolos $\wedge, \vee, \neg, \rightarrow, \leftrightarrow$, que, formam as expressões booleanas. Por vezes, as variáveis são chamadas de variáveis proposicionais ou letras proposicionais e as expressões booleanas conhecidas como lógica proposicional. (ANDERSEN, 1997).

Segundo Andersen (1997) a sintaxe de uma expressão booleana é definida pela gramática:

$$t ::= xi|0|1|\neg t|t \wedge t|t \vee t|t \rightarrow t|t \leftrightarrow t$$

onde xi denota a variável proposicional que varia ao longo de um conjunto de variáveis booleanas. A sintaxe inclui parênteses para resolver ambiguidades e modificar a ordem de prioridade relativa entre os operadores que seguem na seguinte ordem decrescente: $\neg$, $\wedge$, $\vee$, $\rightarrow$, $\leftrightarrow$, por exemplo:

$$\neg x_1 \wedge x_2 \vee x_3 \rightarrow x_4 = \Big(\big((\neg x_1) \wedge x_2\big) \vee x_3\Big) \rightarrow x_4,$$

Uma expressão booleana com variáveis x1, ..., xn denota para cada atribuição de valores verdade para as próprias variáveis um valor verdade, conforme a tabela verdade padrão, as atribuições verdade são escritas como sequências de atribuições de valores para as variáveis, como por exemplo, [0/x_1 , 1/ x_2, 0/x_3, 1/x_4] que atribui 0 a x_1 e a x_3, e 1 a x_2 e x_4. Com esta atribuição acima a expressão tem valor 1 (verdadeiro), enquanto [0/x_1 , 1/ x_2, 0/x_3, 0/x_4] a expressão tem valor 0 (falso). (ANDERSEN, 1997).

Duas expressões booleanas são iguais se produzirem os mesmos valores para todas as atribuições. Uma expressão booleana é uma tautologia se produz verdadeira para todas as atribuições. É satisfatória se produz verdadeira para pelo menos uma atribuição. (ANDERSEN, 1997).

Vários métodos têm sido desenvolvidos para representar e manipular funções booleanas. Bryant (1986, apud Hill e Peterson, 1974) destaca alguns métodos clássicos, tais como tabelas verdade e soma de produtos.

Tabelas verdade são adequadas para a manipulação em computadores, especialmente nos recentes processadores vetoriais de alta velocidade [IYY87] ou máquinas paralelas. No entanto, elas precisam 2^n bits de memória para representar uma função de n - entrada, mesmo para funções muito simples. Soma de produtos são considerados como uma forma especial de expressões booleanas formadas pelo AND – OR este tipo de implementação é também chamada de lógica de dois níveis. Eles têm sido extensivamente estudados por muitos anos e empregada para representar a função booleana em computadores. Soma de produtos, por vezes, resulta uma representação mais compacta do que as tabelas verdade; no entanto, os cubos redundantes podem aparecer em operações lógicas, por isso têm de ser reduzidos para verificar tautologia ou equivalência. (MINATO, 1996).

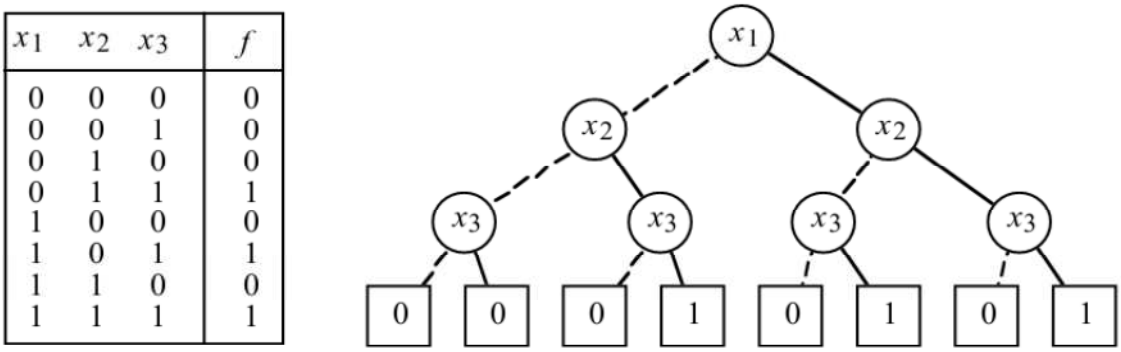
Como é visto os métodos citados acima possuem suas deficiências em praticá-los. Bryant (1986) explica que devido a estas características, a maioria dos programas que processam uma sequência de operações em funções booleanas possui comportamento bastante errático. Eles prosseguem a um ritmo razoável, mas, de repente, podem ficar sem armazenamento ou deixar de concluir uma operação em um período razoável de tempo.

A busca por um método eficiente para a representação de funções tem sido desejado. Então surgiu o Diagrama de Decisão Binária traduzido do inglês Binary Decision Diagrams (BDDs) que são representações gráficas de funções booleanas. O conceito básico foi introduzido por Akers em 1978, e métodos de manipulação eficientes foi desenvolvido por Bryant em 1986. Desde então, BDDs têm atraído a atenção de muitos pesquisadores devido as suas boas propriedades para representar funções booleanas. Embora um BDD pode torna-se de tamanho exponencial para o número de entradas no pior dos casos, o tamanho varia de acordo com o tipo de funções, ao contrário de tabela verdade sempre exigem 2^n bit de memória. Sabe-se que muitas funções práticas pode ser representado por um tamanho viável de BDDs. Esta é uma característica atraente (MINATO, 1996). Na seção 2 veremos conceitos básicos sobre os BDDs.

1.2 DIAGRAMAS DE DECISÃO BINÁRIA

Como foi mencionado na INTRODUÇÃO, um diagrama de decisão binária é um grafo acíclico dirigido composto por nós intermediários, nós terminais e arcos, que permite representar e manipular funções booleanas (BRYANT, 1986). No exemplo da Figura 1, ilustra a representação da função booleana $f(x1, x2, x3)$ definida pela tabela verdade dada à esquerda, para o caso especial ao qual o grafo é uma árvore. Cada vértice não terminal denominado v é marcado por uma variável

Figura 1: Representações de uma função booleana: tabela de verdade e de árvore de decisão.



x_1	x_2	x_3	f
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	0
1	1	1	1

Fonte: (BRYANT, 1992).

denominada $var(v)$, ou seja, os vértices não terminais deste grafo são nomeados pelas variáveis $x1,\ x2\ e\ x3$ e tem arcos direcionados para dois filhos: os arcos denominados arco-0 quando a variável de controle vale 0 representado por linha

tracejada e o arco-1 quando a variável de controle vale 1 representado por linha não tracejada. Cada vértice terminal é rotulado com 0 ou 1. Para uma dada atribuição das variáveis o valor gerado pela função é determinado traçando um caminho desde a raiz até o vértice terminal, seguindo os ramos indicados por valores atribuídos as variáveis. Então o valor da função é dado pelo vértice do terminal rotulado. Devido à forma como os ramos são ordenados na Figura 1, os valores dos vértices terminais, lidos da esquerda para a direita, correspondem aos da tabela verdade, lidos de cima para baixo BRYANT (1992).

Cada vértice de um BDD representa uma decomposição de Shannon (1948) para a função booleana f expressa por:

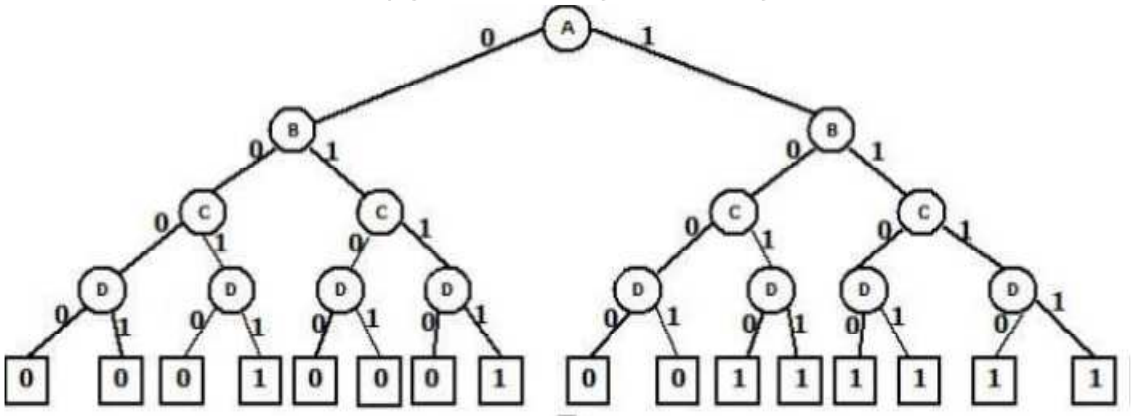
$$f = (\neg x \wedge f[k=0/x]) \vee (x \wedge f[k=1/x]) \text{ para } x \in f,$$

onde f[k/x] (denominada de cofator de f) é a função booleana obtida a partir de f, substituindo-se toda ocorrência da variável x pela constante k. Então as expressões booleanas $f[0/x]$ denominado cofator negativo e $f[1/x]$ denominado cofator positivo, referem-se, respectivamente, à atribuição de 0 e 1 para todas as ocorrências da variável booleana x.

2.1.1 Diagramas de Decisão Binária Ordenado

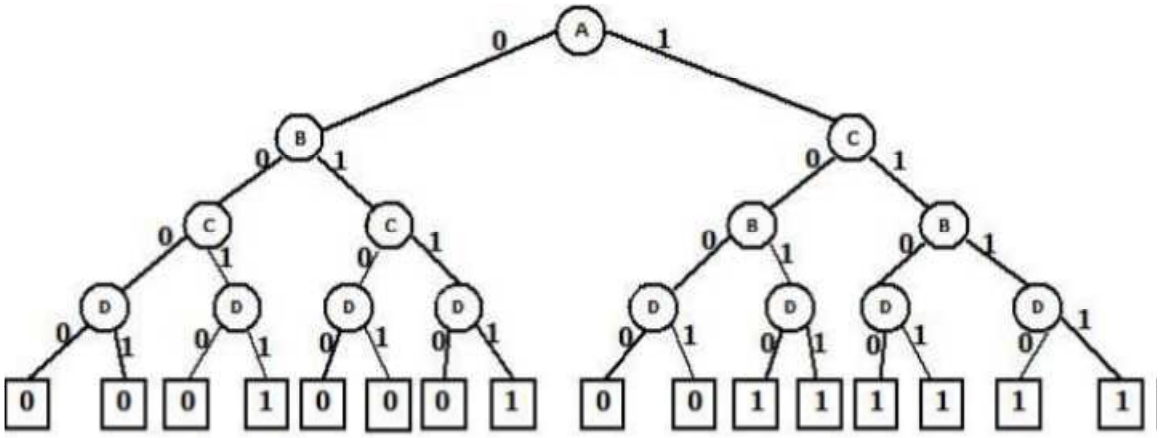
Diagrama de Decisão Binária Ordenado *(Ordered Binary Decision Diagram – OBDD)* é um BDD em que as variáveis de entrada aparecem em ordem fixa e, aparecem apenas uma vez em cada caminho do grafo (MINATO, 1992), o que pode ser observado na Figura 2 em que a função $f = (A \Lambda (B \vee C) \vee (C \Lambda D))$ representa um OBDD, enquanto que na Figura 3 não é um OBDD, pois as variáveis não estão em ordem fixa, logo perde a características de um OBDD. Bryant (1992) afirma que a ideia chave de OBBDs, é que restringindo a representação, a manipulação booleana torne-se computacionalmente mais simples, proporcionando uma estrutura de dados adequada.

FIGURA 2: BDD ORDENADO.



Fonte: (MARQUES, 2013).

FIGURA 3: BDD NÃO ORDENADO



Fonte: (MARQUES, 2013).

> Marques (2003, p. 28) Estes BDDS não se apresentam em uma forma reduzida e, desta forma, seu uso é comprometido na representatividade de funções com muitas variáveis. O consumo de memória neste caso seria equivalente à representação de uma tabela verdade para a função. A vantagem que existe em relação a uma tabela seria o tempo de pesquisa, que é reduzido por se tratar de uma árvore binária.

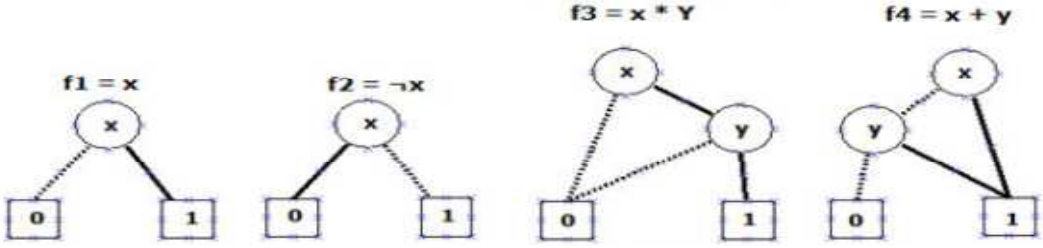
O problema de encontrar uma ordem ideal da variável para a construção de um BDD de tamanho mínimo (em termos de número de vértices necessários para codificar a função) é conhecido por ser NP-completo (RICE E KULHARI 2008, apud BOLLIG, et al., 1996). A complexidade do problema de ordenação de variável para a construção de um BDD necessitou de pesquisadores para estabelecer tipos de ordenação mais eficientes. Geralmente são consideradas como duas classes de técnicas de ordenação de variáveis: a ordenação de variáveis estáticas e ordenação de variáveis dinâmicas (RICE E KULHARI, 2008).

Técnicas de ordenação da variável estática tenta estabelecer a ordem da variável antes da construção real do BDD, enquanto a técnica de ordenação de variável dinâmica tenta ajustar a ordenação durante a real construção do diagrama de decisão. Uma vez que a heurística estática gera a ordem da variável final antes da construção do diagrama de decisão, não existe garantia de soluções de boa qualidade a partir da ordem resultante. Alternativamente, uma vez que a heurística dinâmica permite ajustar a ordem das variáveis durante a própria construção do diagrama de decisão, elas são geralmente consideradas mais eficazes no fornecimento de ordenações eficientes; no entanto, elas também são tipicamente muito mais demoradas (RICE E KULHARI, 2008). Dentre as ordenações estáticas e dinâmicas existem entre outras, por exemplo, *Fine Heuristc* (MALIK et. al, 1988), Meta Heurística (DRECHSLER, 2002), Heurística dos Pesos (VIDAL, 2002), *Sifting* (RUDELL, 1993), *Window* (FUJITA et. al, 1991). Na seção 2.3 é abordado conceitos em relação à heurística dos Pesos.

2.2.2 Diagramas de Decisão Binária Ordenado e Reduzido

Diagramas de Decisão Binária Ordenado e Reduzido (*Reduced Ordered Binary Decision Diagrams - ROBDDs*) foram introduzidos por Bryant (1986), como representações canônicas de funções booleanas, isto é, dado uma ordem fixa das variáveis, existe apenas um ROBDD para representa-la. Marques (2003, p. 28) ressalta: “esta propriedade é muito importante para aplicações práticas, como por exemplo, a verificação de equivalência entre duas funções booleanas, que pode ser facilmente verificada considerando o isomorfismo de seus ROBDDs”. Cardoso (2007) exemplifica na Figura 4 ROBDDs representando algumas das funções booleanas mais simples.

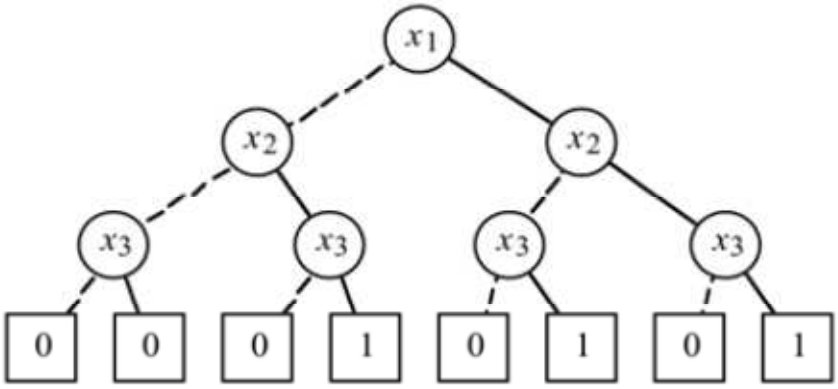
Figura 4: Exemplo de ROBDDs representando funções booleanas



Fonte: (CARDOSO, 2007).

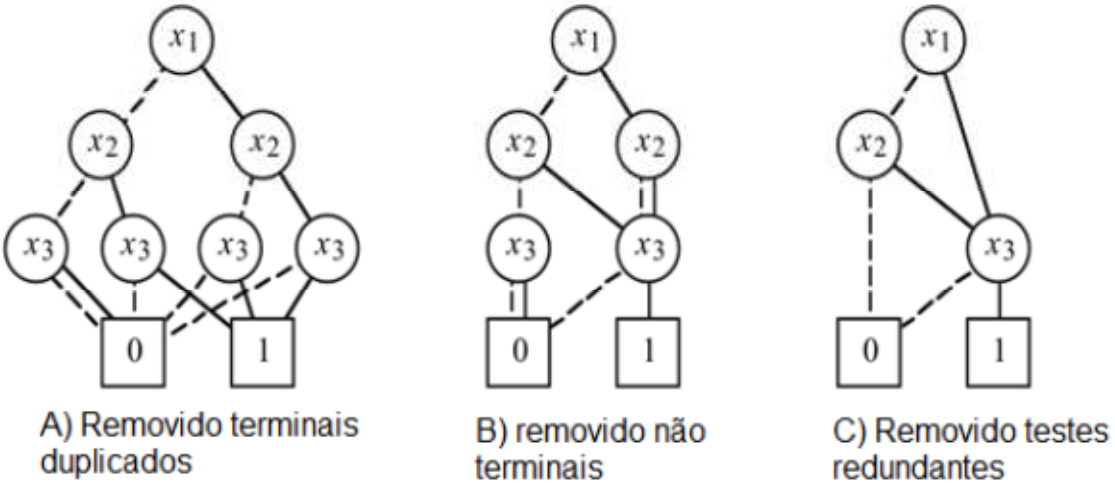
A ideia de ROBDDs é reduzir o tamanho do BDD eliminando vértices redundantes sem alterar a função representada, e para isto existem três regras definidas por Bryant (1992). Estas regras podem ser visualizadas na Figura 6 que é derivada a partir da aplicação dessas regras sobre a Figura 5.

Figura 5: Representações de uma função booleana: árvore de decisão.



Fonte: (BRYANT, 1992).

Figura 6: Redução da Árvore de Decisão para OBDD Aplicando as regras de redução de três para a árvore. Da Figura 5 produz a representação canônica da função como um OBDD.



Fonte: (BRYANT, 1992).

- **Remover terminais duplicados**, isto é eliminar todos os vértices terminais duplicados e redirecionar todos os arcos dos vértices que foram eliminados para os vértices não terminais que restaram;

- **Remover não terminais duplicados**, isto é se os vértices não terminais u e v tem variáveis iguais $(var(u) = var(v))$, o arco representado por linha tracejada do vértice u e o arco representado por linha tracejada do vértice v são iguais, o arco de linha sólida do vértice u e o arco de linha sólida do vértice v são iguais, neste caso se os vértices atenderem estas condições, em seguida elimina um dos dois vértices e redireciona todas as chamadas recebidas até o outro vértice;
- **Remover testes redundantes** se os dois arcos que saem de um vértice não terminal v apontam para um mesmo vértice u, então elimina o vértice v e redireciona todos os arcos que chegam em v para o vértice u.

  Satisfazendo a propriedade de ordenação, o tamanho do BDD pode ser reduzido por várias vezes, aplicando as regras de transformação, por exemplo, na Figura 6, aplicando a primeira regra de transformação (A), reduz de oito vértices terminais para dois; aplicando a segunda regra de transformação (B), elimina dois dos vértices terminais tendo variável $x3$, elimina os arcos para vértices terminal com rótulos 0 e 1; aplicando a terceira regra de transformação (C), elimina dois vértices: um com variável $x3$ e outro com variável $x2$. (BRYANT, 1992).

2.2.3 Heurística dos Pesos

O tamanho do BDD tem relação com sua ordenação, ou seja, tem dependências conforme a ordem que as variáveis estão dispostas, então se torna necessário o estudo e desenvolvimento de algoritmos para encontrar boas ordens de variáveis. Para isso, várias técnicas de ordenação estática e reordenação de BDDs têm sido desenvolvidas (VIDAL, 2002).

Segundo Vidal (2002) pesquisas têm mostrado que os métodos de reordenação são eficientes, no entanto quando se utiliza uma heurística estática que fornece uma boa ordenação, logo os métodos de reordenação se tornam mais rápidos e encontram melhores ordens. Porém quando a ordenação inicial não é boa, os métodos de reordenação não conseguem reverter para uma boa ordem.

Como visto na seção 2.1 existem vários tipos de ordenação tanto estáticas como dinâmicas, no entanto esta seção aborda um algoritmo de ordenação estática denominada de heurística dos pesos, que foi desenvolvida no trabalho de Vidal (2002), cujo, objetivo foi apresentar um algoritmo de ordenação estática eficiente, para BDDs que representam máquinas de estado finito, como circuitos sequenciais. Este algoritmo permite identificar a dependência das funções de transição com relação às variáveis, ou seja, o conjunto de variáveis da qual esta função depende. No conjunto de dependências de muitas funções que estão as variáveis, entende – se de forma intuitiva que é esse conjunto que define o comportamento do sistema e, quando colocado essas variáveis no inicio da ordenação, pode otimizar o tamanho do BDD e tornar os algoritmos que percorrem o grafo, mais rápidos.

A ideia principal do algoritmo que define a heurística dos pesos, é a de atribuir pesos as variáveis, aos quais as variáveis que tiverem o maior peso ficam no topo da ordenação dos BDDs. Para vigorar esta ideia é necessária avaliar as funções de transição de cada variável de estado e identificar de que outras variáveis ela depende (VIDAL, 2002).

O algoritmo 01 consiste em, para cada variável $x \in X$, determinar de quais variáveis sua função de transição depende. Para isto considera - se a denominação Si o conjunto de todas as variáveis que são argumentos da função de transição de $xi,$ chamado de suporte de xi. Quanto mais vezes uma variável aparece suporte de outra mais esta variável influenciará o comportamento do sistema, ou seja, maior é o peso desta variável, então ela é colocada no topo da ordenação, O peso dessa variável é determinado pelo numero de funções que dependem dela (VIDAL, 2002).

O algoritmo a seguir, serve para calcular os pesos das variáveis seguindo os passos de execução: inicialmente cada variável recebe peso 0, (**linha 2 a linha 4**) então o suporte Si é calculado para cada função de transição f_i (**linha 5 a linha 6**) e os pesos de cada variável xj em Si é incrementado (**linha 7 a linha 10**) (VIDAL, 2002).

Depois da atribuição dos pesos das variáveis, e através destes, conforme o valor do peso de cada variável é definido uma ordenação inicial para os BDDs que representam estas variáveis. Deve ser observado que não há dependência nas funções das variáveis que representam o próximo estado são denominadas de x',

logo as variáveis x' ficam no fim da ordem. Mas, como os algoritmos de manipulação de BDDs faz o produto relacional das variáveis de estado x com x', então elas devem sempre estarem próximas umas das outras. Isto acontece da seguinte forma: o algoritmo mantém na ordenação cada variável x_i antes de x_i' agrupando – ás. (VIDAL, 2002).

Algoritmo 01 ordenação inicial de variáveis

Begin ..... **linha 1**
foreach $wi \in W$ **do** ..... **linha 2**
$peso(wi) \leftarrow 0$ ..... **linha 3**
Od ..... **linha 4**
foreach $x_i \in X$ **do** ..... **linha 5**
$S_I \leftarrow suporte(f_i)$ ..... **linha 6**
foreach $w_i \in S_i$ **do** ..... **linha 7**
$peso(wj) \leftarrow peso(wj) + 1$ ..... **linha 8**
Od ..... **linha 9**
Od ..... **linha 10**
$ordena(W, peso)$ ..... **linha 11**
End ..... **linha 12**

Fonte: (VIDAL, 2002).

A seguir Vidal (2002) exemplifica o funcionamento do algoritmo: Em um circuito, temos o conjunto das variáveis de estado deste circuito $X = \{x_1, x_2, x_3\}$, o conjunto de entradas do circuito $E = \{p\}$, e um conjunto de cópias das variáveis de estado $X' = \{x_1', x_2', x_3'\}$.Dessa forma temos sete variáveis BDDs para representar este circuito e as seguintes relações de transição:

$$f_1(W) = (x_3 \wedge x_2) \oplus x_1$$

$$f_2(W) = x_3 \oplus x_2$$

$$f_3(W) = \neg x_3 \oplus p$$

Através dessas relações podemos identificar os pesos das variáveis na Tabela 1. Nesta tabela pode ser observado que as variáveis de próximo estado não têm pesos calculados, pois não há dependência de funções destas variáveis. As variáveis de próximo estado como já visto, elas são ordenadas após a variável que lhe corresponde no estado atual. Observe como fica a ordenação inicial das variáveis BDDs do circuito:

$$\{x_3, x_3', x_2, x_2', x_1, x_1', p\}$$

Tabela 1: Cálculo dos pesos das variáveis.

Variáveis	Pesos
x_1	1
x_2	2
x_3	3
p	1

Fonte: (VIDAL, 2002).

2.2.4 Operador ITE (If – Then - Else)

Segundo Brace et all (1990), por questões de eficiência na manipulação de funções booleanas de BDDs na simulação simbólica de ROBDDs, é conhecido que a operação ITE abreviado do inglês If-The-Else, pode ser utilizado para implementar todas as operações booleanas de duas variáveis, como mostrado na Tabela 2. O operador ITE é uma função booleana definida por três entradas F, G, H que calcula: se F então G senão H, equivalente a:

$$ITE(F,G,H) = F \cdot G + \bar{F} \cdot H \text{ (æ } F \text{ então } G, \text{ænão } H).$$

A Tabela 2 impõe uma forma canônica forte em vértices no ROBDD, de modo que cada vértice representa uma função lógica exclusiva. Esta tabela mapeia uma tripla (v, G, H) para um vértice do ROBDD $F = (v, G, H)$. Cada vértice no ROBDD tem uma entrada na tabela única. Antes de um vértice ser adicionado no ROBDD é feito uma pesquisa na tabela que determina se o vértice para esta função já existe, caso exista, então este vértice é utilizado, caso contrário, um novo vértice é adicionado ao ROBDD e é feita uma nova entrada na tabela original. Então é suposto que quando é criado um novo vértice F, os vértices G e H já obedece à forma canônica forte (BRACE et. al., 1990).

A formulação recursiva para a computação *ITE(F,G,H)* para as funções de *F,G,H* representada em forma ROBDD. Seja *F, G e H* os vértices de um ROBDD e v a variável topo destes vértices, então *ITE(F,G,H)* pode ser calculado através da seguinte formulação recursiva:

$$ITE(F, G, H) = ITE(v, ITE(F\ \mathbf{v=1}, G\ \mathbf{v=1}, H\ \mathbf{v=1}), ITE(F\ \mathbf{v=0}, G\ \mathbf{v=0}, H\ \mathbf{v=0})).$$

Os casos terminais para a recursão são:

$$ITE(1, F, G) = ITE(0, G, F) = ITE(F, 1, O) = F.$$ (BRACE et. al., 1990).

Tabela 2: Funções de duas variáveis descritas pela operação ITE

Tabela	Name	Expression	Equivalent form
0000	0	0	0
0001	$AND\ (F,G)$	$F \cdot G$	$ite(F,G,0)$
0010	$F > G$	$F \cdot \overline{G}$	$ite(F,\overline{G},0)$
0011	F	F	F
0100	$F <$ G	$\overline{F} \cdot G$	$ite(F,0,G)$
0101	G	G	G
0110	$XOR(F,G)$	$F \oplus G$	$ite(F,\overline{G},G)$
0111	$OR(F,G)$	$F + G$	$ite(F,1,G)$
1000	$NOR(F,G)$	$\overline{F \oplus G}$	$ite(F,0,\overline{G})$
1001	$XNOR(F,G)$	$\overline{F \oplus G}$	$ite(F,G,\overline{G})$
1010	$NOT(G)$	$\overline{G}$	$ite(G,0,1)$
1011	F >= G	$F \cdot \overline{G}$	$ite(F,1,\overline{G})$
1100	$NOT(F)$	$\overline{F}$	$ite(F,0,1)$
1101	F <= G	$\overline{F} \cdot G$	$ite(F,G,1)$
1110	$NAND\ (F,G)$	$\overline{F \cdot G}$	$ite(F,\overline{G},1)$
1111	*1*	*1*	*1*

Fonte: (BRACE et all, 1990).

3. METODOLOGIA

Primeiramente foi feito a pesquisa bibliográfica do conteúdo a ser trabalhado nesta temática, que abrange uma variedade de métodos para representar expressões booleanas (tabela verdade, BDDs, etc.), conceito de BDDs, tipos de BDDs (OBDDs, ROBDDs), tipos de ordenação (dinâmicas e estáticas), a operação ITE (IF – THEN - ELSE), ao qual foram utilizados artigos, dissertações e um livro de (MINATO, 1996), encontrados através de buscas no google acadêmico.

Foram utilizados os estudos de casos de Oliveira (2014) como base de dados para avaliar os ROBDDs gerados a partir de especificações booleanas de sistemas indústrias reais, ao qual foi utilizado para a ordenação destes, a ordenação estática com a heurística dos pesos de Vidal (2002), e também foi utilizado a ferramenta visBDD para a geração desses ROBDDs e a ordenação citada.

3. ESTUDOS DE CASOS

Nestes Estudos de Casos, o objetivo é avaliar o método de ordenação utilizando a heurística de peso nos ROBDDS, gerados a partir de especificações booleanas de sistemas industriais, dos seguintes estudos de casos: Sistema de Prevenção de incêndio; Sistema de Segurança de Detecção de Fogo ou Gás e; Sistema de Controle de Nível de um Tanque.

3.1 SISTEMA DE PREVENÇÃO DE INCÊNDIO

Na descrição desse sistema extraído do trabalho de Oliveira (2014) temos as seguintes especificações: o sistema é composto por: três sensores (F1, F2, F3) utilizados para detectar fogo; um alarme; um led; um botão; uma chave manual e um atuador, utilizado para ativar dois extintores de incêndio de forma automática. As seguintes operações são realizadas:

- O alarme só é disparado se no mínimo dois sensores acusarem fogo;
- O botão pode ser utilizado para ativar o alarme de incêndio;
- A chave manual pode ser usada para desligar o alarme depois que os sensores forem desligados;
- Caso haja uma tentativa de desligar o sistema de alarme com os sensores ativados o alarme não desliga;
- O led acionado indica que no mínimo um dos sensores está ativado;
- Caso o alarme ou o led sejam acionados então, o atuador é programado para colocar em ação os dois extintores de incêndio.

Para este sistema são especificadas: um conjunto das Entradas E e um conjunto das Saídas S.

$E = \{F1, F2, F3, manual, chave\};$

$S = \{s1, s2, s3, s4\}$. Cada saída é uma tupla da forma (Y, f) em que Y é o conjunto das variáveis de entrada e f é a função booleana que define a saída.

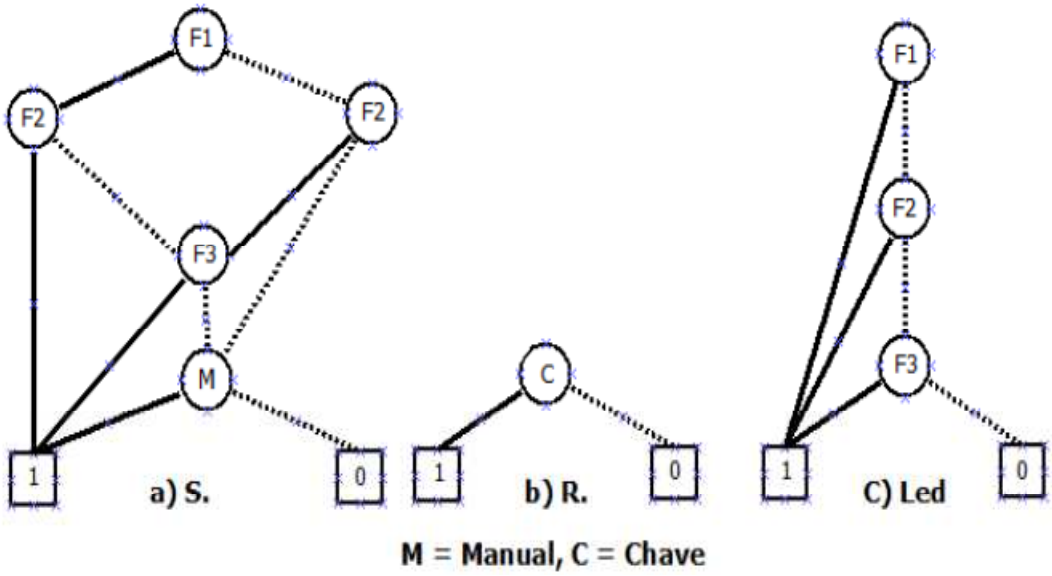
- $s1 = S = (\{F1, F2, F3\}, ((F1 \wedge F2) \vee (F1 \wedge F3) \vee (F2 \wedge F3) \vee Manual))$
- $s2 = R = (\{chave\}, chave)$
- $s3 = Alarme = (\{S, R\}\ ((\mathrm{R} \wedge Alarme) \vee (R \wedge S)))$

– $s4 = Led = (\{F1,\ F2,\ F3\},\ (F1 \vee F2 \vee F3)\ ).$

A partir destas saídas são gerados os ROBDDs utilizando a ordenação com a heurística dos pesos, isto pode ser observado na Figura 7. E os ROBDDS que não utilizam a heurística dos pesos na Figura 8.

Observa-se que para as 4 saídas existem apenas 3 ROBDDS, isto acontece devido à função booleana para a saída $Alarme$ ser representada por uma célula de memória, ou seja, neste caso o resultado para a função é dado pelos valores das variáveis de saída $S\ e\ R$.

Figura 7: ROBDDs para as saídas do sistema de prevenção de incêndio utilizando a ordenação com a heurística dos pesos.

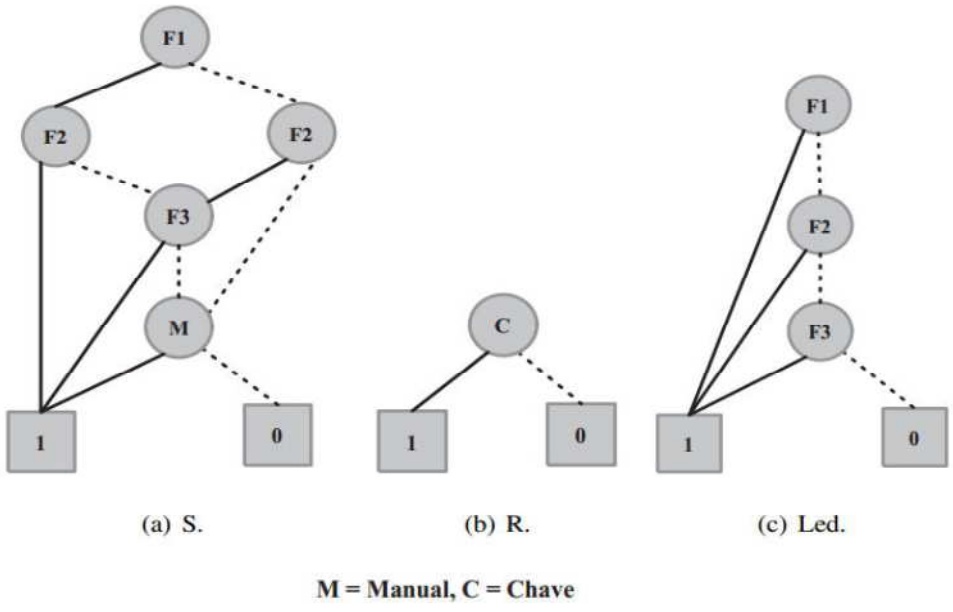


Fonte: Autoria própria.

Comparando a geração dos ROBDDS da Figura 7 com a Figura 8 extraída do trabalho de Oliveira (2014). Pode – se deduzir que para estas funções booleanas não há redução do ROBDD utilizando a heurística dos pesos em relação à Figura 8, por exemplo, na Figura 8 na saída *(a)* aplicando a heurística dos pesos obtêm-se em cada variável, os seguintes pesos: $F1 = 2, F2 = 2, F3 = 2\ e\ M = 1$, com a atribuição destes pesos tem-se a seguinte ordenação: $F1 < F2 < F3 < M$; logo a ordenação utilizando a heurística dos pesos não influencia o tamanho do RBDD, pois $F1,\ F2\ e\ F3$ possuem pesos iguais. Na saída *(b)* o tamanho será o mesmo, pois temos apenas a variável $C = 1$, logo há apenas a possibilidade de uma ordenação independente da heurística que se utilize. Na saída *(C)* as variáveis possuem os

seguintes pesos: $F1 = 1,\ F2 = 1\ e\ F3 = 1$ e através destes pesos tem-se a seguinte ordenação: $F1 < F2 < F3$; logo a mudança de ordenação não altera o tamanho do ROBDD, pois todas as variáveis possuem pesos iguais.

FIGURA 8: ROBDDs para as saídas do sistema de prevenção de incêndio



Fonte: (OLIVEIRA, 2014).

3.2 SISTEMA DE SEGURANÇA PARA DETECÇÃO DE FOGO OU GÁS

Na descrição desse sistema extraído do trabalho de Oliveira (2014) temos as seguintes especificações: O sistema é composto por cinco sensores, ao qual dois são para detectar fumaça $(SF1, SF2)$ e três para detectar gás $(SG1, SG2, SG3)$, um tanque, um dispositivo que libera gás CO_2 no momento que há confirmação na zona denominada $(DispCO_2)$, um alarme sonoro que indica a presença de gás $(AlaGDZ)$ e duas válvulas utilizadas para controlar o nível de gás no tanque, ao qual, uma dessas válvulas é denominada auxiliar, ela é acionada quando ocorre vazamento de gás na válvula principal e, é utilizada para manter o sistema estável evitando eventuais danos às instalações da zona. As seguintes operações são realizadas no sistema:

- se algum dos sensores for acionado de fumaça então, foi detectado fogo na zona;
- se os sensores de gás 1 e 2 ou 3 forem acionados então, foi detectado presença de gás na zona;
- se os sensores 2 e 3 forem acionados então, foi detectado presença de gás na zona;
- se for detectado fogo na zona então o alarme sonoro que indica presença de fogo é acionado e após 2 segundos o dispositivo que libera gás CO_2 e a válvula principal é desligada. O gás CO_2 liberado é utilizado para sanar o fogo na zona. Quando não houver mais presença de fogo na zona o alarme sonoro é desligado;
- se for detectado fogo na zona, então o alarme sonoro que indica presença de gás é acionado e após 4 segundos a válvula que controla o nível de gás no tanque é desligada e a válvula que mantém o sistema estável é aberta. Esta válvula é utilizada para controlar a quantidade de gás no tanque de forma que não haja danificações nas instalações e nem desperdício de material. Quando não houver mais presença de gás na zona então o alarme sonoro é desligado;

Para este sistema são especificadas: um conjunto das Entradas E e um conjunto das Saídas S.

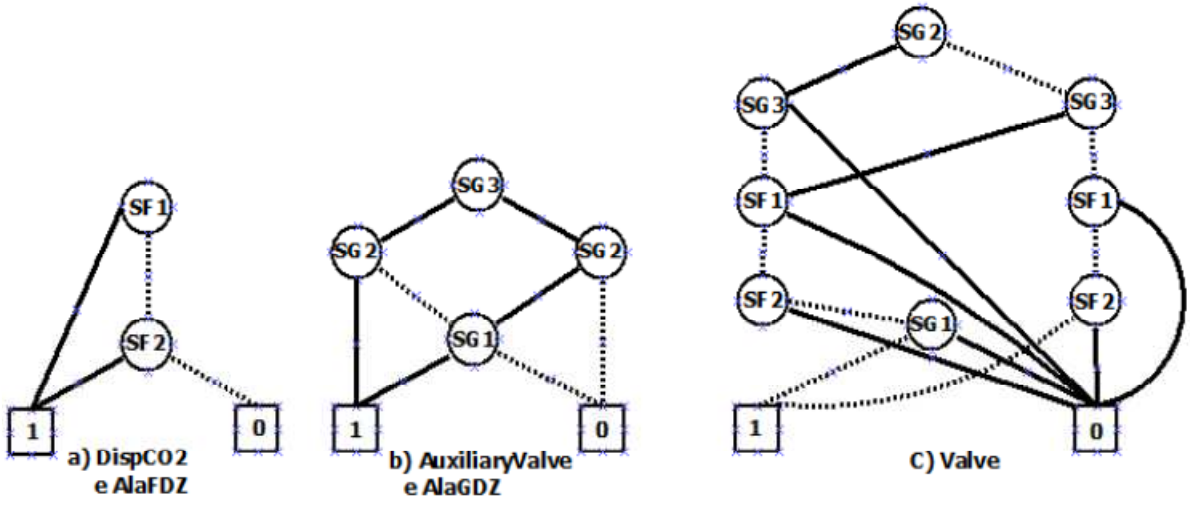
- $E = \{SF1, SF2, SG1, SG2, SG3\}$
- $S = \{s1, s2, s3, s4, s5\}$. Cada saída é uma tupla da forma (Y, f), em que Y é o conjunto de variáveis de entrada e f é a função booleana que define a saída.

  - $s1 = DispCO2 = \big(\{SF1, SF2\}, (SF1 \vee SF2)\big)$;
  - $s2 = AlaFDZ = \big(\{SF1, SF2\}, (SF1 \vee SF2)\big)$;
  - $s3 = AuxiliaryValve = (\{SG1, SG2, SG3\}, ((SG1 \wedge (SG2 \vee SG3)) \vee (SG2 \wedge SG3)))$;
  - $s4 = AlaGDZ = (\{SG1, SG2, SG3\}, ((SG1 \wedge (SG2 \vee SG3)) \vee (SG2 \wedge SG3)))$;
  - $s5 = Valve = (\{SF1, SF2, SG1, SG2, SG3\}, (\neg (SF1 \vee SF2) \wedge \neg ((SG1 \wedge (SG2 \vee SG3)) \vee (SG2 \wedge SG3))))$.

Observa-se que para as 5 saídas foram gerados somente 3 ROBDDS. Isto acontece porque a função Booleana é a mesma para as saídas $DispCO2$ e $AlaFDZ$ e também, as saídas $AuxiliaryValve$ e $AlaGDZ$ possuem a mesma função booleana.

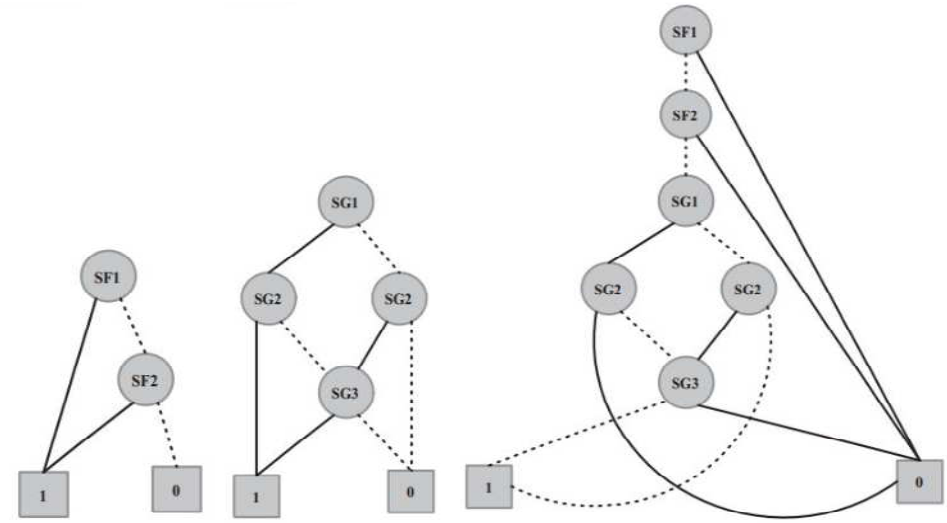
Comparando a geração dos ROBDDS da Figura 9 com a Figura 10 extraída do trabalho de Oliveira (2014). Pode – se deduzir que as saídas $(DispCO_2$ e $AlaGDZ)$ e também $(AuxiliaryValve\ e\ AlaGDZ)$ utilizando a ordenação de heurística dos peso, não alterou o tamanho do ROBDD, por exemplo, na Figura 9 nas saídas $DispCO_2$ e $AlaGDZ$ utilizando a heurística dos pesos obtém-se em cada variável os seguintes pesos: $SF1 = 1\ e\ SF2 = 1$, com a atribuição destes pesos tem a seguinte ordenação: $SF1 < SF2$, logo para estas saídas aplicando a heurística dos pesos não altera o tamanho do ROBDD, pois as variáveis possuem pesos iguais. Em relação às saídas $AuxiliaryValve$ e $AlaGDZ$ obtém-se de cada variável os seguintes pesos: $SG1 = 1$, $SG2 = 2\ e\ SG3 = 2$, e as seguintes ordenações possíveis $SG2 < SG3 < SG1$, logo estas ordenações não alteram o tamanho do ROBDD.

Figura 9: ROBDDs para as saídas do sistema de segurança para detecção de fogo ou gás, utilizando a ordenação com a heurística de peso.



Fonte: Autoria própria.

Figura 10: ROBDDs para as saídas do sistema de detecção de fogo ou gás



Fonte: (OLIVEIRA, 2014).

Porém utilizando esta mesma heurística na saída $Valve$ houve diferenças de tamanho, tornando o ROBDD maior em relação ao da Figura 10. Por exemplo, os pesos obtidos de cada variável foram: $SF1 = 1,\ SF2 = 1,\ SG1 = 1,\ SG2 = 2,\ SG3 = 2$, e através desses pesos é obtida a seguinte ordenação: $SG2 < SG3 < SF1 < SF2 < SG1$.

Justificando o aumento da saída $Valve$ com a operação ITE, utilizando a ordenação com a heurística dos pesos, obtém-se as seguintes operações: respeitando a ordem de prioridade:

- Para a função: $\neg\,(SF1 \vee SF2\,)$, então faz o $ITE(F,G,H)$ obtendo para esta função as seguintes operações: $ITE(SF1,0,SF2) \rightarrow ITE(1,0,SF2) \rightarrow ITE(0,0,SF2)$ a partir dessas operações, obtém o seguinte resultado: o $SF1$ no topo aponta o $arco\_1$ para o $terminal\ 0$ e o $arco\_0$ para vértice $SF2$, que tem seu $arco\_1$ apontando para o $terminal\ 0$ e o $arco\_0$ apontando para o $terminal\ 1$.
- Para a função $(SG2 \vee SG3)$ obtém-se: $ITE(SG2,1,x5) \rightarrow ITE(1,1,SG3) \rightarrow ITE(0,1,SG3)$, a partir dessas operações ITE, tem o seguinte resultado: o $SG2$ é raiz que tem o $arco\_1$ apontando para o $terminal\ 1$ e o $arco\_0$

apontando para o $SG3$, que tem seu $arco\_1$ apontando para o $terminal\ 1$ e o $arco\_0$ apontando para o $terminal\ 0$.

- Para a função $(SG1 \wedge (SG2 \vee SG3))$ obtém-se: $ITE(SG1, SG2,0) \rightarrow ITE(SG1,1,0) \rightarrow ITE(SG1, SG3,0) \rightarrow ITE(SG1,1,0) \rightarrow ITE(SG, 0,0)$, a partir dessas operações ITE, tem o seguinte resultado: o vértice $SG2$ como raiz, ao qual seu $arco\_1$ aponta para o $SG1$, que tem o $arco\_0$ apontando para o $terminal\ 0$ e o $arco\_1$ apontando para o $terminal\ 1$, o $arco\_0$ do $SG2$ apontando para o $SG3$, que tem o seu $arco\_1$ apontando para o $SG1$ e o $arco\_0$ apontando para o $terminal\ 0$.
- Para a função $(SG2 \wedge SG3)$ tem: $ITE(SG2, SG3,0) \rightarrow ITE(1, SG3,0) \rightarrow ITE(0, SG3,0)$, a partir dessas operações ITE obtém-se o seguinte resultado: o $SG2$ como raiz, ao qual seu $arco\_1$ aponta para o $SG3$ que tem o $arco\_1$ apontando para o $terminal\ 1$ e o $arco\_0$ apontando para o $terminal\ 0$, o $arco\_0$ do $SG3$ aponta para o $terminal\ 0$.
- Para esta função $\neg ((SG1 \wedge (SG2 \vee SG3)) \vee (SG2 \wedge SG3))$ obtém – se: $ITE(SG2,0, SG2) \rightarrow ITE(SG1,0, SG3) \rightarrow ITE(SG1,0,0) \rightarrow ITE(1,0,1) \rightarrow$ ITE(0,0,1), obtendo o seguinte resultado: o vértice $SG2$ como raiz, que tem o $arco\_1$ apontando para o $SG3$, que tem o seu $arco\_1$ apontando para o $terminal\ 0$ e o seu $arco\_0$ apontando para o $SG1$, que tem o seu $arco\_1$ apontando para o $terminal\ 0$ e o $arco\_0$ apontando para o $terminal\ 1$, o $arco\_0$ do $SG1$ aponta para outro vértice $SG3$ que tem o $arco\_1$ apontando para o $SG1$, e o $arco\_0$ apontando para o $terminal\ 1$.

E por ultimo tem a operação ITE da função completa da saída $Valve$: $(\neg (SF1 \vee SF2) \wedge \neg ((SG1 \wedge (SG2 \vee SG3)) \vee (SG2 \wedge SG3)))$,então faz o $ITE(SF1, SG2,0) \rightarrow ITE(SF1, x5,0) \rightarrow ITE(SF1,0,0) \rightarrow ITE(SF1, SG1,0) \rightarrow$ ITE(0,SG1,0) → ITE(SF2,SG1,0)→ ITE(0,SG1,0) → $ITE(1, SG1,0) \rightarrow ITE(SF1, x5,0) \rightarrow ITE(SF1, x3,0) \rightarrow ITE(SF1,1,0)$, através desta operação obtém o resultado da saída $Valve$ na Figura 9.

3.3 SISTEMA DE CONTROLE DE NÍVEL DE UM TANQUE

Na descrição desse estudo de caso extraído do trabalho de Oliveira (2014) temos as seguintes especificações: o controle do nível de um tanque é realizado

através do acionamento de um sistema de drenagem composto por uma válvula de escape e uma bomba de sucção. Esse sistema pode funcionar de 4 modos a seguir:

- modo 1: completamente manual;
- modo 2: completamente automático;
- modo 3: bomba manual e válvula automática;
- modo 4: bomba automática e válvula manual;

O modo de funcionamento desejado é escolhido pela combinação dos estados de duas chaves, $Automatico,\ Automatico2$, que são utilizadas para selecionar, respectivamente, o modo automático ou manual para o acionamento da bomba e da válvula. Com a planta funcionando no modo 1, tanto a válvula quanto a bomba são comandadas de forma independente pelo operador. Para a válvula essa ação é feita através das chaves $Abre\ e\ Fecha$, e para a bomba através do $Liga\ e\ Desliga$. No modo 2, o estado dos sensores de nível determinam as ações que serão realizadas. Os sinais dos sensores são $Nivel\_Muito\_Baixo,\ Nivel\_Baixo,$ $e\ Nivel\_Muito\_Alto$. Nos modos em que a bomba ou a válvula estão no modo manual as ações são função da combinação do estado dos sensores e dos comandos do operador. Existem também ações de emergência que são tomadas independentemente do modo selecionado, e que são combinadas com as ações de qualquer modo escolhido para determinar o estado final do sistema. As possíveis combinações para os sinais de entrada e as correspondentes ações que devem ser efetuadas pelo sistema são apresentadas, a seguir uma descrição sobre estas:

Independente do modo de funcionamento selecionado segue:

- se $Nivel\_Muito\_Baixo$ apresentar o valor verdadeiro por $5s$ consecutivos a ação tomada pelo sistema deve ser desligar a bomba e fechar a válvula.
- se $Nivel\_Muito\_Alto$ apresentar o valor verdadeiro por $5s$ consecutivos, abrir a válvula .

A seguir, na descrição dos modos, os estados apresentados serão combinações entre as ações dos respectivos modos e das ações de emergência (elas serão apresentadas em negrito).

No modo 1, o comportamento dos dois dispositivos é determinado pelo operador. Mas se for verificado alguma das combinações de sinais consideradas críticas, então o sistema executará as ações de emergência programadas.

No modo 2, se $Nivel\_Baixo$ apresentar valor verdadeiro por $5s$ consecutivos, então deve – se ligar a bomba e **abrir a válvula.**

No modo 3, se $Nivel\_Muito\_Baixo$ apresentar valor verdadeiro por 5s consecutivos, então é necessário desligar a bomba e **fechar a válvula.** Se $Nivel\_Baixo$ for verdadeiro por $5s$ consecutivos, então o sistema realizará a ação de fechar a válvula e o estado da bomba será determinado pelo operador. Se $Nível\_Muito\_Alto$ for verdadeiro por $5s$ consecutivos, o estado da bomba será determinado pelo operador e a ação do sistema será **abrir a válvula**.

No modo 4, se $Nivel\_Muito\_Baixo$ apresentar valor verdadeiro por $5s$ consecutivos, então deve - se **desligar a bomba e fechar a válvula.** Se $Nivel\_Baixo$ apresentar valor verdadeiro por 5s, então o sistema realizará a ação de desligar a bomba, e o estado da válvula será determinado pelo operador. Se $Nivel\_Muito\_Baixo$ for verdadeiro por $5s$ consecutivos, então deve – se ligar a bomba e abrir a válvula.

Para este sistema são especificadas: um conjunto das Entradas E e um conjunto das Saídas S.

- $E = \{Nivel\_Muito\_Baixo,\ Desliga,\ Automatico,\ Nivel\_Baixo,\ Nivel\_Muito\_Alto,\ Liga, Abre,\ Automatico2,\ Fecha\}$;
- Conjunto das saídas: $S = \{s1,\ s2,\ s3, s4,\ s5,\ s6,\ s7,\ s8,\ s9,\ s10\}$. Cada saída é uma tupla da forma $\{Y, f\}$, em que Y é o conjunto de variável de entradas e f é a função booleana que define a saída.

  - $s1 = In\_Timer1 = (\{Nivel\_Muito\_Baixo\}, (Nível\_Muito\_Baixo))$;
  - $s2 = In\_Timer2 = (\{Desliga\}, (Desligar))$;
  - $s3 = In\_Timer3 = (\{Nivel\_Baixo\}, (Nivel\_Baixo))$;
  - $s4 = Desliga = (\{Nivel\_Muito\_Baixo, Desliga, Automatico, Nível\_Baixo\}, ((out\_Timer1) \vee (out\_Timer2 \vee \neg Automatico) \vee (out\_Timer3 \wedge Automatico)))$;

- $s5 = In\_Timer4 = (\{Nivel\_Muito\_Alto\}, (Nivel\_Muito\_Alto))$;
- $s6 = In\_Timer5 = (\{Nivel\_Muito\_Alto, Automatico, Liga\}, ((out\_Timer4 \wedge Automatic) \vee (\sim Automatic \wedge Liga)))$;
- $s7 = In\_Timer6 = (\{Abre\}, (Abre))$;
- $s8 = In\_Timer7 = (\{Fecha\}, (Fecha))$;
- $s9 = S = (\{Nivel\_Muito\_Alto, Abre, Automatico2\}, ((Nivel\_Muito\_Alto)\ or\ (out\_Timer6\ and\ !\,Automatico2)))$;
- $s10 = R = (\{Nivel\_Baixo, Automatico2, Nivel\_Muito\_Baixo, Fecha\}, ((Nivel\_Baixo \wedge Automatico2) \vee (out\_Timer7 \wedge \neg Automatico2) \vee (Nivel\_Muito\_Baixo)))$;

Comparando a geração dos ROBDDs da Figura 11 com a Figura 12, pode – se deduzir que: as saídas In_Timer1, In_Timer2, In_Timer3, In_Timer4, In_Timer6 e In_Timer7, utilizado a heurística dos peso, não alterou o tamanho do ROBDD, pois cada uma dessas saídas possuem apenas uma variável de entrada, por exemplo, respectivamente as saídas possuem as seguintes variáveis com seus pesos: $NMB = 1, D = 1, NB = 1, NMA = 1, Ab = 1\ e\ F = 1$, logo estas saídas geram o mesmo ROBDD.

Na saída In_Timer5 houve uma redução no tamanho do ROBDD quando foi aplicado a heurística dos pesos em sua ordenação, por exemplo, nesta saída obtém-se os seguintes pesos das variáveis de entrada: $A = 2, 04 = 1, L = 1$, ao qual através da atribuição desses pesos fornece a seguinte ordenação: $A < 04 < L$, logo a redução ocorreu devido a variável A que possui peso 2 passou a ser raiz do ROBDD, sendo assim reduziu um vértice A.

Justificando a diminuição da saída In_Timer5 com a operação ITE utilizando a ordenação com a heurística dos pesos, obtém-se as seguintes operações: respeitando a ordem de prioridade:

- Primeiro foi feita a operação com a função $(out\_Timer4 \wedge Automatic)$, que denominaremos $out\_Timer4\ e\ Automatic$, respectivamente de 04 e A, então faz o $ITE(F, G, H)$ obtendo para esta função: $ITE(F, G, 0) \rightarrow ITE(F, G, 0) \rightarrow ITE(04, A, 0) \rightarrow ITE(04,1,0) \rightarrow ITE(04,0,0)$, a partir dessas operações ITE obtém-se o seguinte resultado: o vértice A como raiz, ao qual seu $arco\_1$

aponta para o 04 que tem o $arco\_1$ apontando para o $terminal\ 1$ e o $arco\_0$ apontando para o $terminal\ 0$, e o $arco\_0$ do A apontando para o $terminal\ 0$.

- Para a função $\sim Automatic$ obtém-se: $ITE(A,0,1) \rightarrow ITE(1,0,1) \rightarrow ITE(0,0,1)$, tem o seguinte resultado: o vértice A como raiz, ao qual o $arco\_1$ aponta para o $terminal\ 0$ e o $arco\_0$ aponta para o $terminal\ 1$.
- Para a função $(\sim Automatic \wedge Liga)$ que chamamos $Liga$ de L, então faz o $ITE(x2,x3,0) \rightarrow ITE(0,x3,0) \rightarrow ITE(1,x3,0)$, obtém-se o seguinte resultado: A como raiz, ao qual o $arco\_1$ aponta para o $terminal\ 0$ e o $arco\_0$ aponta para o L, que tem o $arco\_0$ apontando para o $terminal\ 0$ e $arco\_1$ apontando para o $terminal\ 1$.

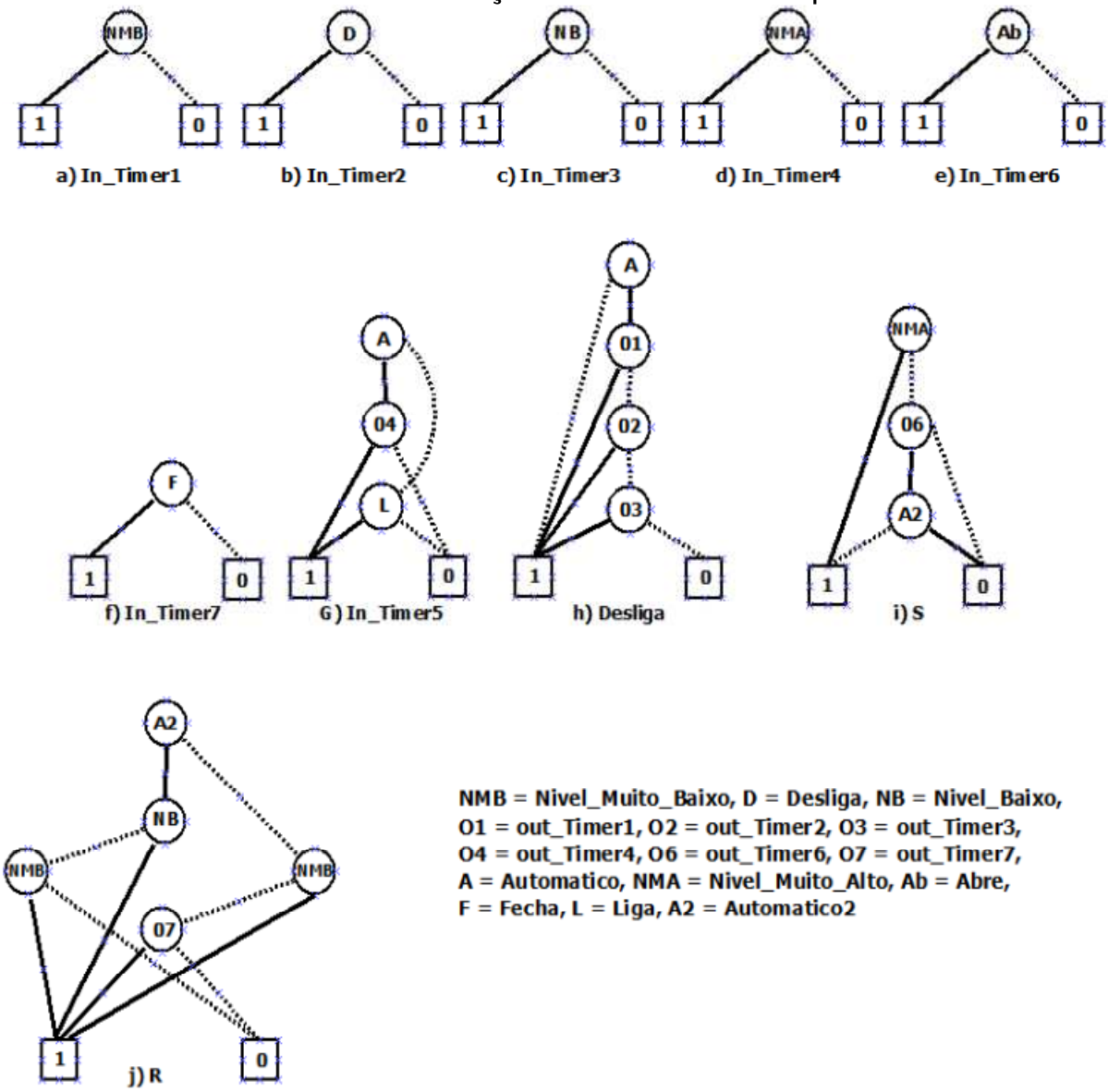
E por ultimo obtêm-se a operação ITE da função completa da saída $In\_Timer5 : ((out\_Timer4 \wedge Automatic) \vee (\sim Automatic \wedge Liga))$, então faz o $ITE(A,1,A) \rightarrow ITE(04,1,0) \rightarrow ITE(0,1,L)$, obtendo o resultado da saída In_Timer5 da Figura 11.

Na saída $Desliga$ não houve mudança no tamanho do ROBDD, por exemplo, utilizando a heurística dos pesos obtém-se os seguintes pesos de cada variável de sua função booleana: $A = 2,\ 01 = 1,\ 02 = 1\ e\ 03 = 1$, e através desses pesos chegamos a seguinte ordenação: $A < 01 < 02 < 03$, logo o tamanho do ROBDD permaneceu o mesmo.

Na saída S o tamanho do ROBDD não sofreu mudança, pois todas as variáveis possuem o mesmo peso, sendo assim a ordenação permanece default, por exemplo, utilizando a heurística dos pesos são obtidos os seguintes pesos: $06 = 1$, $NMA = 1\ e\ A2 = 1$, e através destes foi obtido a ordenação $NMA < 06 < A2$, logo o ROBDD permaneceu o mesmo em relação a Figura 12.

Na saída R ocorreu uma redução no ROBDD quando aplicou – se a heurística dos pesos. Na ordenação default, o ROBDD tem 6 vértices já na ordenação dos pesos ele passou a ter 5 vértices, por exemplo, com aplicação da heurística obtém-se os seguintes pesos de cada variável: $A2 = 2, NB = 1, NMB = 1\ e\ 07 = 1$, e através destes pesos obtém - se a ordenação $A2 < NB < NMB < 07$.

Figura 11: ROBDDs para as saídas do sistema de controle de nível de um tanque, utilizando a ordenação com a heurística de peso.



Fonte: Autoria própria.

Justificando a diminuição da saída R com a operação ITE utilizando a ordenação com a heurística dos pesos, obtemos as seguintes operações: respeitando a ordem de prioridade:

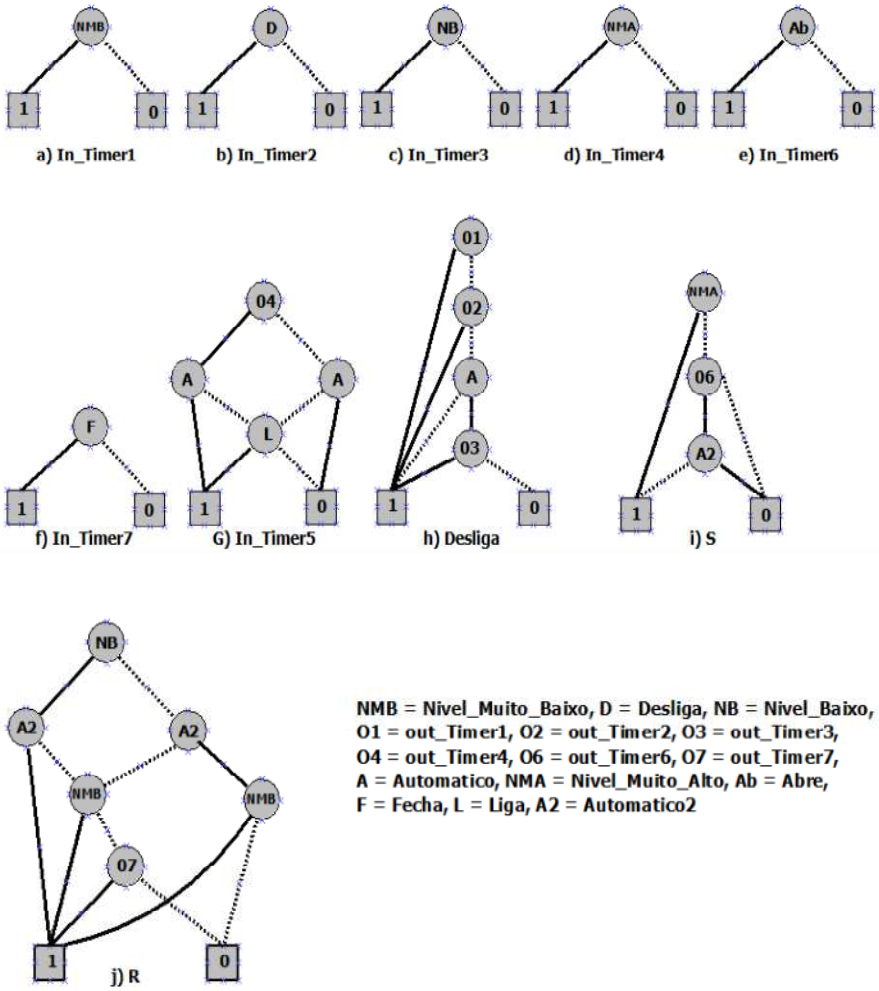
- Primeiro foi feita a operação com a função $(Nivel\_Baixo \wedge Automatico2)$, ao qual chamamos as variáveis $Nivel\_Baixo$ e $Automatico2$, respectivamente de NB e $A2$, então faz o $ITE(F,G,H)$ obtendo: $ITE(F,G,0) \rightarrow ITE(NB,A2,0) \rightarrow ITE(NB,1,0) \rightarrow ITE(NB,0,0) \rightarrow ITE(NB,0,0)$, a partir dessas operações ITE

tem o seguinte resultado: o vértice $A2$ como raiz, cujo seu $arco\_0$ aponta para o $terminal\ 0$ e o $arco\_1$ aponta para o não terminal NB, que tem o $arco\_0$ apontando para o $terminal\ 0$ e o seu $arco\_1$ aponta para o $terminal\ 1$.

- Para a função $\neg Automatico2$ obtém-se: $ITE(A2,0,1) \rightarrow ITE(1,0,1) \rightarrow ITE(0,0,1)$, a partir dessas operações ITE obtém-se o seguinte resultado: o $arco\_1$ aponta para o $terminal\ 0$ e o $arco\_0$ aponta para o $terminal\ 1$.
- Para a função $(out\_Timer7 \wedge \neg Automatico2)$ chamamos as variáveis out_Timer7 de 07, tem-se: $ITE(07, A2,0) \rightarrow ITE(07,0,0) \rightarrow ITE(07,1,0)$, a partir dessas operações ITE obtém-se o seguinte resultado: o $arco\_1$ do $A2$ aponta para o $terminal\ 0$ e o $arco\_0$ aponta para o 07, que tem o $arco\_1$ apontado para o $terminal\ 1$ e o $arco\_0$ apontado para o $terminal\ 0$.
- Na função $(Nivel\_Baixo \wedge Automatico2\ ) \vee (out\_Timer7 \wedge \neg Automatico2)$ tem-se as seguintes operações: $ITE(A2, 1, A2) \rightarrow ITE(NB, 1, 0) \rightarrow ITE(0, 1, 07)$, obtém-se o seguinte resultado: o $arco\_1$ do vértice $A2$ aponta para o NB que tem o $arco\_1$ apontado para o $terminal\ 1$ e tem o $arco\_0$ apontando para o $terminal\ 0$ e, o $arco\_0$ do $A2$ aponta para o vértice 07, que tem o $arco\_1$ apontado para o $terminal\ 1$ e tem o $arco\_0$ apontando para o $terminal\ 0$.

E por ultimo obtém-se a operação ITE da função completa da saída R: $((Nivel\_Baixo \wedge Automatico2\ ) \vee (out\_Timer7 \wedge \neg Automatico2) \vee (Nivel\_Muito\_Baixo)\ ))$ ao qual chamamos a variável $Nivel\_Muito\_Baixo$ de NMB, então faz o $ITE(A2,1, NMB) \rightarrow ITE(NB, 1, x3) \rightarrow ITE(1,1, NMB) \rightarrow ITE(0,1, NMB) \rightarrow ITE(07,1, x3) \rightarrow ITE(07,1,1) \rightarrow ITE(07,1,0)$, obtendo o resultado da saída R da Figura 11.

Figura 12: ROBDDs para as saídas do sistema de controle de nível de um tanque.



Fonte: (OLIVEIRA, 2014).

CONSIDERAÇÕES FINAIS

Este trabalho se propôs a avaliação dos ROBDDS gerados a partir de especificações booleanas de sistemas indústrias reais utilizando a heurística dos pesos. E para isto foi necessário conceitos sobre funções booleanas, BDDs, OBDDs, ROBDDs, tipos de ordenações de ROBDDs. Um enfoque maior foi dado aos ROBDDs, e ao tipo de ordenação estática utilizando a heurística dos pesos, para gera-los.

O objetivo deste trabalho foi atingido, pois foi feita a aplicação do algoritmo da ordenação estática heurística dos pesos, nos estudos de casos, utilizando a ferramenta visBDD desenvolvida por (MEINEL et al., 2002) que utiliza a operação ITE e a ordenação manual e, obtiveram os seguintes resultados nos sistemas de especificações booleanas dos sistemas industriais:

- No primeiro estudo de caso, referente ao **Sistema de Prevenção de Incêndio**, a aplicação desta heurística não surtiu efeito no tamanho dos ROBDDs em relação a ordenação default da ferramenta visBDD que é a ordenação quando inicializa a ferramenta dada em ordem cresceste das variáveis encontradas em uma determinada expressão logica;
- No segundo estudo de caso, que é o **Sistema de Segurança para Detecção de Fogo ou Gás,** apenas na saída $valve$ houve diferenças de tamanho, tornando o ROBDD maior, então não é recomendável utilizar esta heurística para esta saída;
- No terceiro estudo de caso, que é o **Sistema de Controle de Nível de Um Tanque, apenas nas** saídas In_Timer5 e R houve mudanças no tamanho ocorrendo uma redução, então é recomendável utilizar a heurística dos pesos para ordenação, ao invés de usar a ordenação default da ferramenta citada.

Este trabalho contribuiu para geração de ROBDDs menores, demonstrando o potencial de se utilizar a heurística dos pesos para ordená-los, e estes ROBDDs podem ser utilizados na geração de casos de teste, para os tipos de especificações booleanas de sistemas industriais.

Sugestões para trabalhos futuros:

- extrair casos de teste a partir dos ROBDDs gerados nos estudos de casos;
- realizar mais estudos de casos para avaliar a viabilidade da heurística dos pesos;
- Desenvolver uma ferramenta que gere ROBDDs de forma automática utilizando esta heurística;

REFERÊNCIAS

ALVES, L. V. **Verificação de equivalência combinacional utilizando hiper-relução binária.** Belo Horizonte, 2010, 86 p.

ANDERSEN, H. R. **Introduction to Binary Decision Diagrams**, 1997.

BRACE, K.S., RUDELL, R.L., & BRYANT, R.E. 1990. **Efficient Implementation of a BDD Package**. Pages 40–45 of: Proceedings of the 27th ACM/IEEE Conference on Design Automation.

BRYANT, R. E. **Graph-based algorithms for boolean function manipulation**. IEEE Transactions on Computers, 35(8):677691, 1986.

_____. **Symbolic Boolean Manipulation with Ordered Binary Decision Diagrams.** 1992.

CARDOSO, T. M. G. **Exploração de Reordenamento de ROBDDs no Mapeamento Tecnológico de Circuitos Integrados.** Porto Alegre, 2007, 92 p.

DRECHSLER, Rolf. **Evaluation of static variable ordering heuristics for MDD construction [multi-valued decision diagrams]**. In: Multiple-Valued Logic, 2002. ISMVL 2002. Proceedings 32nd IEEE International Symposium on. IEEE, 2002. p. 254-260.

FERREIRA, N. F. G. **Verificação formal de sistemas modelados em estados finitos**. São Paulo, 2005.

FLORES, P. **Utilização de Diagramas de Decisão Binária (BDDs) para Representar Máquinas de Estado (FSMs) e Verificar Fórmulas de Lógica Temporal (CTL)**, 1996.

FUJITA, Masahiro; MATSUNAGA, Yusuke; KAKUDA, Taeko. **On variable ordering of binary decision diagrams for the application of multi-level logic synthesis**. In: Proceedings of the conference on European design automation. IEEE Computer Society Press, 1991. p. 50-54.

MARQUES, F. S.. **Um Algoritmo Formal para Remoção de Redundâncias.** Porto Alegre, 2003.

MALIK, S. et. al. **Logic Verification Using Binary Decision Diagrams in a Logic Synthesis Environment**. International Conference on CAD: pp.6-9, 1988.

Meinel, C., Sack, H., & Schillings, V. **VisBDD - A Web-based Visualization Framework for OBDD Algorithms**. IWLS, 2002, 385–390.

MINATO, Shin-Ichi. **Binary decision diagrams and applications for VLSI CAD**. Kluwer International series in Engineering and Computer Science. Boston, Dordrecht, London: Kluwer Academic Publishers, 1996.

OLIVEIRA, Kézia Vasconcelos. **Geração e execução automática de testes para p programas de controladores lógicos programáveis para sistemas instrumentados de s segurança.** Cidade**:** Campina Grande – PB, Universidade Federal de Campina Grande Centro de Engenharia Elétrica, 2014.

PEREIRA E BARROS. **Diagramas de Decisão Binária.** RT-MAC-2007-05, IME-USP, 2007.

RICE M. E KULHARI S. **A Survey of Static Variable Ordering Heuristics for Efficient BDD/MDD Construction**.Technical report, University of California, Riverside, 2008.

RUDELL, Richard. **Dynamic variable ordering for ordered binary decision diagrams**. In: Proceedings of the 1993 IEEE/ACM international conference on Computer-aided design. IEEE Computer Society Press, 1993. p. 42-47.

SOUSA, T. C. **Revisão de modelos formais de sistemas de estados finitos.** São Paulo, 2010, 91 p.

SHANNON, C. E. 1948. **A Mathematical Theory of Communication**. The Bell SystemTechnical Journal, 27(July, October), 379–423, 623–656.

VIDAL, J. M. B. **Ordenação inicial de BDDs para a verificação automática de sistemas de transição finita**. 2002. 70. Dissertação (Mestrado em Sistemas e Computação) - Departamento de Informática e Matemática Aplicada do Centro de Ciências Exatas e da Terra da Universidade Federal do Rio Grande do Norte, Natal, 2002.