Ferramentas de suporte ao estudo da Disciplina de Sistemas Digitais – Mapas de Karnaugh

Sérgio Barbosa, Manuel Bernardo Cunha, Augusto Silva

***Resumo* – Este artigo pretende dar a conhecer uma ferramenta para simplificação de funções booleanas, desenvolvida no âmbito do programa de Sistemas Digitais, parte integrante do Projecto Matemática Ensino (PmatE).**

O documento começa por apresentar exemplos de aplicabilidade, descrevee sucintamente a tecnologia utilizada, enuncia e descreve (com alguns exemplos) os diferentes passos do algoritmo e finaliza, ilustrando graficamente os resultados obtidos em laboratório.

Destaca-se a forma peculiarmente relevante como foi implementada, tendo como principais objectivos criar uma ferramenta essencialmente útil, fácil na usabilidade e de suporte comum, embora especialmente concebida para plataformas Windows.

Para além de reunir este conjunto de interesses, de referir ainda a capacidade deste algoritmo por a nu algumas fragilidades de ferramentas como o *Espresso*, que embora se superiorize no seu desempenho mostra, em raros casos, sinais de menor eficiência.

***Abstract* – This article intends to reveal a tool for Boolean functions simplification, developed in the scope of the Digital Systems program, part of the Mathematical Education Project.**

The document begins by showing some work examples, describes the technology used in the project, it enunciates and describes (with examples) all the steps of the algorithm and finishes by illustrating graphically the results gathered in laboratory.

We highlight the relevant way how it was implemented, having as immediate goals the utility of the tool, the ease of usability, in a well disseminated platform (MS Excel), although specially created for Windows.

Beyond all these interests, it still refers the capacity of the algorithm bares some fragilities of tools as the *Espresso* that, although having better performance, it shows, in rare cases, signs of smaller efficiency.

I. INTRODUÇÃO

A integração da Disciplina de Sistemas Digitais na plataforma do Projecto Matemática Ensino conduziu a equipa de trabalho ao estudo pormenorizado das matérias que fazem parte do programa da disciplina. Este estudo serviu de suporte à elaboração de conteúdos, obrigando, em vários momentos, à equação de algoritmos, autênticos motores de resolução de problemas e que, naturalmente, foram reaproveitados para a construção de ferramentas que podem ser utilizadas fora do contexto da plataforma. Estas ferramentas possuem características semelhantes a outras que mais vulgarmente utilizamos, como a máquina de calcular, e que podem ser úteis quando pretendemos confirmar um resultado, ou simplesmente usufruir da velocidade de processamento quando a aprendizagem das regras não está em causa.

Pode-se dividir este artigo em 4 partes. Uma que contextualiza a necessidade de utilização deste tipo de ferramentas, abordando o aspecto da minimização de funções lógicas. Uma outra que descreve a tecnologia utilizada na construção da ferramenta, argumentando a opção feita, outra ainda que descreve os diferentes passos do algoritmo, ilustrando com imagens sempre que se julgar conveniente. Por fim, uma avaliação final de desempenho da ferramenta.

II. APLICAÇÃO DA FERRAMENTA

A. Área científica

A Álgebra de Boole tem aplicações práticas nomeadamente em algumas áreas da electrónica, mais concretamente nas áreas do domínio digital, como por exemplo na Disciplina de Sistemas Digitais, leccionada a alguns cursos da Universidade de Aveiro. Esta disciplina dedica-se ao estudo de componentes digitais, *edifícios* lógicos mais complexos obtidos à custa de componentes simples e circuitos que integram todos estes componentes. A álgebra permite modelar o comportamento destes circuitos, e operações de simplificação sobre funções lógicas que podem resultar em circuitos mais simples, logo, circuitos mais eficientes na relação custo desempenho.

Quando temos que lidar com funções lógicas a forma mais imediata é utilizar uma simplificação com base no conjunto de postulados da Álgebra de Boole e todos os teoremas que deles derivam. Mas sempre que as funções booleanas tomam proporções de alguma complexidade, pode ser vantajoso recorrer a ferramentas de minimização, processo mais rápido e sujeito a menos erros.

No universo de ferramentas existentes podemos destacar os mapas de Karnaugh (MK) e o método de Quine McCluskey como ferramentas de uso manual. Contudo, existem ferramentas automáticas que resultam da implementação informatizada das heurísticas que são utilizadas nos processos manuais. Foi com base nestes

dois algoritmos que foi desenvolvida a ferramenta descrita neste artigo. A primeira utilizada para encontrar o número de implicantes primos total, identificando os que são essenciais, e a segunda como referência para extracção das heurísticas responsáveis por rejeitar e incluir os restantes implicantes que asseguram a cobertura da função. O conjunto de implicantes apurado representa uma das soluções possíveis. A ferramenta descrita no artigo apresenta a solução final sob a forma de soma de produtos, ou por outro lado, disjunção de conjunções.

O estudo de ferramentas para apresentar o resultado sob a forma de produto de somas também foi levado acabo, sendo apenas necessário pequenas alterações no software ao nível da representação, uma vez que a estrutura de processamento se mantém.

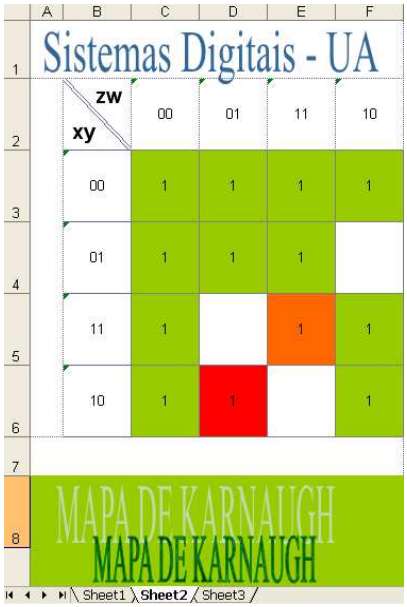

Figura 1 - Mapa de Karnaugh

B. Usabilidade

Para utilizar a ferramenta basta apenas ter conhecimento dos índices dos produtos de variáveis independentes que estão associados à saída igual a "1" da função (ou *verdade*). Para isso basta apenas colocar sobre sob a forma de tabela de verdade a função lógica (relembro) de 4 variáveis e dela obter os referidos índices.

A utilização da ferramenta é feita através da interface da Figura 2, onde é possível definir as variáveis que participam na função, os termos mínimos através dos índices e os desconhecidos, utilizando o mesmo processo.

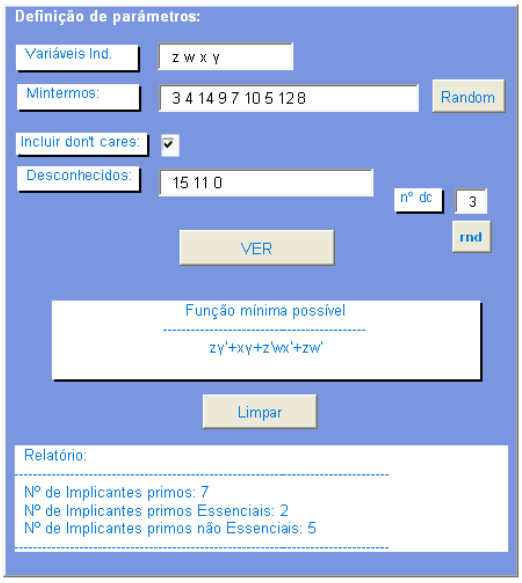

Figura 2 - Interface para inserção de dados

A ferramenta processa os dados, e vai identificando os implicantes obtidos (demarcação por cor e referência aos índices dos termos mínimos). Para cada um o utilizador é obrigado a confirmar o implicante, ao mesmo tempo que vão sendo identificados graficamente no mapa.

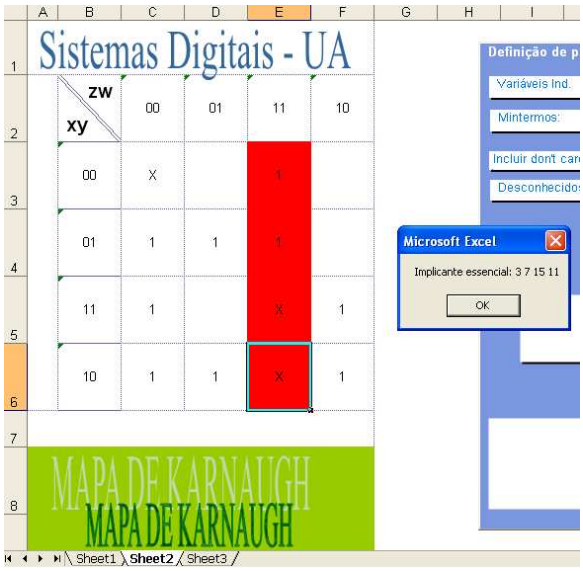

Figura 3 - Identificação de um implicante primo

A resposta do algoritmo é feita de forma faseada, para que seja perceptível ao utilizador identificar a evolução do *execução* do algoritmo.

No final é afixado um pequeno relatório com os dados sobre o número de implicantes, descriminando em essenciais e não essenciais, e uma das soluções possíveis relativamente à função mínima na forma soma de produtos. É importante sublinhar que a solução nem sempre é única e aquela encontrada pela ferramenta é apenas uma das possíveis.

O ambiente de trabalho global encontra-se na Figura 4.

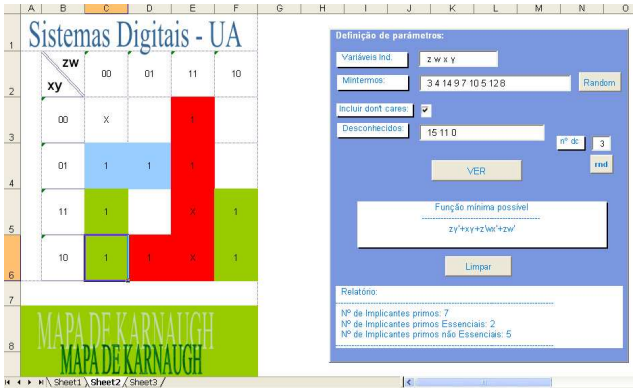

Figura 4 - Interface completa

De referir ainda a possibilidade de incluir desconhecidos. Ou seja, combinações de variáveis independentes que tanto podem assumir os valores lógicos "1" ou "0". Neste exercício a ferramenta deve apresentar a opção feita, com os desconhecidos que incluiu (colorindo-os) e com os que desprezou (desprezando-os).

Falta apenas referir que o algoritmo permite simplificar funções que possam ir até 16 variáveis independentes. Ou seja, o código fonte foi retirado do ficheiro Excel (laboratório de desenvolvimento) integrado na plataforma sob a forma de função, servindo diferentes modelos com funções de *n* variáveis.

III. TECNOLOGIA UTILIZADA

O algoritmo que resolve o mapa de karnaugh foi originalmente desenvolvido para funcionar como motor de processamento para modelos de MK. Uma vez que a plataforma de desenvolvimento dos modelos é suportada por Visual Basic, e o MK sendo uma ferramenta com um formato matricial, o Excel acabou por se apresentar como uma boa solução, integrando o VBA (Visual Basic for Applications) e uma folha de cálculo numa só aplicação.

Terminado o período de desenvolvimento e testes, foi só colocar o conjunto de funções (código fonte) na plataforma do PmatE, definindo a interface da função principal a ser invocada.

A arquitectura do código desenvolvido é essencialmente procedimental, recorrendo a várias estruturas complexas de dados. Estas estruturas foram fundamentais no suporte das diferentes propriedades que caracterizam os diferentes implicantes primos.

Além das vantagens já enunciadas, rapidamente revelou-se ainda mais útil por constituir um bom suporte de aprendizagem. Apesar do seu aspecto pouco formal da ferramenta, ela permite que os alunos possam gerar mapas de Karnaugh, e armazenar no mesmo ficheiro os exemplos mais relevantes.

IV. ALGORITMO

A. Heurísticas do algoritmo

O algoritmo obtido foi desenvolvido segundo um conjunto de heurísticas que se passa a enunciar:

1. Encontrar todos os implicantes primos;
2. Ordenar cada implicante primo por ordem decrescente segundo o número de termos mínimos que contêm;
3. Determinar os implicantes primos essenciais;
4. Eliminar todos os implicantes primos que estão cobertos por implicantes primos essenciais;
5. Determinar os implicantes que não intersectam os essenciais;
6. Para todos os implicantes encontrados no ponto 5 (não essenciais e que não intersectam implicantes primos essenciais) determinar a sua prioridade (a prioridade do implicante é tanto maior quanto menor for o número de termos mínimos que contêm, que intersectam outros implicantes na mesma condição).
7. Ordenar por ordem decrescente de prioridade.
8. Incluir implicantes pela ordem formada, desde que os implicantes seleccionados não intersectem outros que já pertençam à soma mínima ou que tenham um peso (nº de termos mínimos por incluir) igual ao valor máximo;
9. Incluir implicantes primos cujos termos mínimos não pertençam ainda nenhum implicante mas eventualmente possam cruzar com outros implicantes de peso máximo.
10. Pesar os implicantes restantes (nº de termos mínimos que contém e que não estão cobertos por nenhum implicante já pertence à soma mínima);
11. Ordenar por ordem decrescente de peso os implicantes acima;
12. Incluir os últimos implicantes
    a. Incluir na soma mínima o primeiro da lista obtida no passo anterior (se existir);
    b. Pesar os implicantes restantes;
    c. Ordenar novamente por ordem decrescente e voltar a a).

*B. Algoritmo **step-by-step***

Este sector da secção IV dedica-se à demonstração, com pequenos exemplos, dos diferentes passos do algoritmo.

Para identificar implicantes primos, será usada a seguinte notação, por exemplo [Ka+Kb] para representar o implicante primo constituído pelos termos mínimos das células de coordenadas Ka e Kb.

1. Implicantes Primos

Definidos os índices correspondentes aos termos mínimos de uma função o primeiro passo é identificar todos os implicantes primos (conjunto de células contíguas a 1, em número correspondente a uma potência de 2 e com formato rectangular). O código fonte que identifica os implicantes foi retirado do livro *Digital Design Principles and Practices* **[2]**, capítulo 4 (*Combinational Logic Design Principles*) e representa a primeira heurística referida no ponto ***A*** desta secção. No exemplo da Figura 5 os implicantes primos [C3+D3], [C5+D5] e [E4+F4] são essenciais, os restantes ([C3+C4]; [C4+C5] e [C4+F4]) são não essenciais.

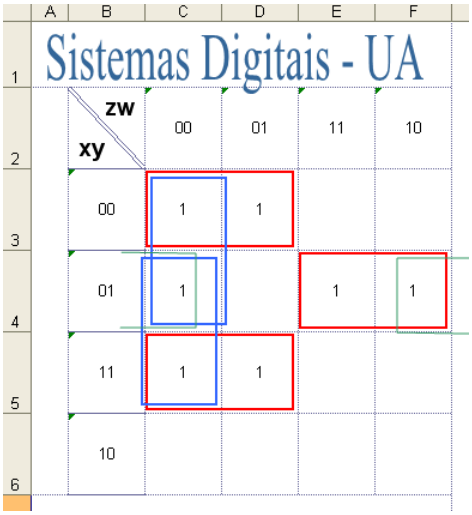

Figura 5 - Implicantes primos

2. Ordenar por ordem crescente de peso

Esta tarefa é levada acabo com o objectivo de memorizar o implicante com maior número de termos mínimos (peso máximo). Informação necessária à *heurística 8*.

3. Determinar os implicantes essenciais

Implicantes primos essenciais são todos aqueles que possuem, pelo menos um, termo mínimo que não é coberto por mais nenhum implicante. Na Figura 5 é possível identificar o termo mínimo na célula 3D, o termo na célula 4E e na 5D. Cada um destes mintermos fazem dos implicantes onde estão integrados essenciais, definidos na por [C3+D3], [C5+D5] e [E4+F4].

4. Eliminar os implicantes cobertos por essenciais

No exemplo da Figura 6 o implicante [D4+E4] é totalmente coberto por implicantes primos essenciais. Este segmento do algoritmo é responsável pelo eliminar desta redundância.

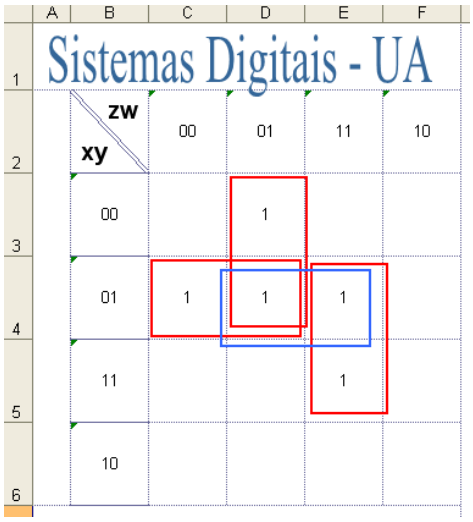

Figura 6 - Exemplo de mapa de Karnaugh

5. Determinar implicantes que não intersectam essenciais

Ao identificar os implicantes primos que não intersectam essenciais permite dar prioridade, por exemplo, ao implicante definido por [4D+ 4E] na Figura 7, relegando para uma segunda análise o implicante de células [3D+4D] ou o implicante definido por [4E+5E]. Uma vez assegurada a cobertura da totalidade dos termos mínimos, estes últimos implicantes acabam por ser desprezados, resultando numa soma mínima de produtos.

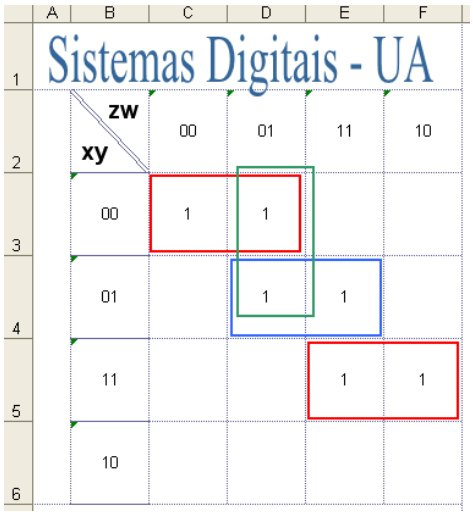

Figura 7 - Exemplo associado à heurística 5

6. Cálculo de prioridade

A heurística nº6 evita que ocorra o que se verifica no exemplo da Figura 8. Num cenário em que a totalidade dos implicantes são não essenciais, estes devem ser assinalados por uma determinada ordem. Esta ordem deve respeitar um alinhamento em que o implicante seguinte a seleccionar deve ser contíguo a qualquer um dos

anteriormente afixados. Isto só é possível, estabelecendo um ranking que é obtido através da atribuição de uma prioridade com posterior ordenação.

7. Ordenar por ordem crescente de prioridade

Processo que se segue à execução da heurística anterior.
O implicante mais prioritário será aquele com menor número de intersecções.

8. Incluir implicantes pela ordem formada...

Figura 8 - Exemplo para heurística 6

De acordo com a Figura 8, se o implicante constituído pelos termos mínimos [C3+C4] foi o primeiro a ser seleccionado, o seguinte só poderá ser [D3+E3], ou [E4+F4]. Esta ordem resulta do facto de cada um destes implicantes apenas intersectar um termo mínimo dos implicantes ainda não considerados, que não intersectam implicantes que fazem já parte da soma mínima.

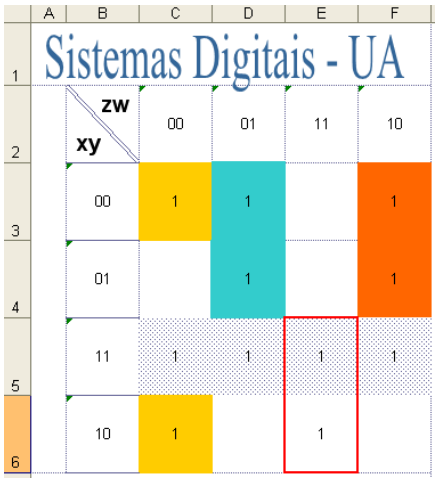

Figura 9 - 2º exemplo associado à heurística 8

Ainda relativamente à Figura 8, o implicante primo constituído por [E3+E4] intersecta dois termos mínimos, logo o menos prioritário.

Esta heurística também tem em linha de conta o peso dos implicantes, não deixa que nesta fase os implicantes a afixar intersectem outros de peso superior (implicantes com mais termos mínimos por cobrir).
Não considerando os implicantes essenciais [E5+E6], implicantes primos como [D3+D4] ou [C3+C6] têm prioridade em relação a implicantes como [D4+D5] ou [C5+C6].

9. Incluir implicantes primos cujos termos...

Figura 10 - Exemplo associado à heurística 9

Na Figura 10 existem 3 implicantes nestas condições: [D4+E4], [C4+D4] e [D4+D5]. Aleatoriamente o sistema fixa um dos 3 e a partir desse momento deixam de existir implicantes nessas condições: implicantes que, simultaneamente, não intersectem outros com peso máximo (peso igual a 4, neste caso) ou que já pertençam à função mínima.

10. Pesa os implicantes restantes...

A colocação dos últimos implicantes exige uma nova pesagem (ganha prioridade os implicantes primos com maior nº de termos mínimos a descoberto).

11. Ordenar por ordem decrescente os últimos implicantes...

Aqui a política de selecção dá maior importância aos implicantes finais que cubram um maior número de termos mínimos ainda por cobrir.

12. Incluir os últimos implicantes...

É possível observar na Figura 11 que, dos últimos implicantes ainda por assinalar ([C4+C5+D4+D5]; [D4+D5+E4+E5] e [E3+E4+E5+E6]), é o implicante formado pelos termos mínimos [D4+D5+E4+E5] que se apresenta com maior prioridade, e por isso, o eleito para ser assinalado no mapa.

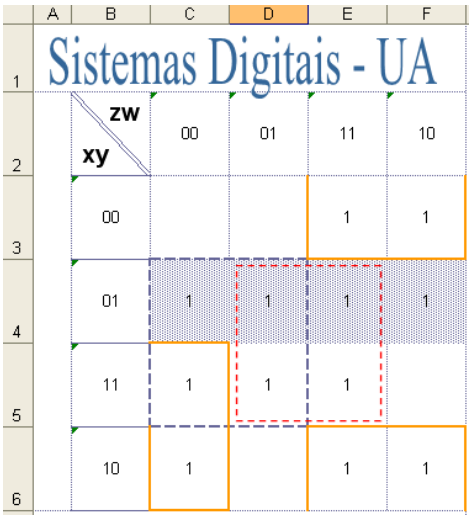

Figura 11 - Exemplo associado à heurística

A heurística nº 12 traduz-se num processo iterativo que, após afixar o implicante com maior prioridade ([C4+C5+D4+D5]), volta à heurística nº 10 e repete o processo.

Assim, após ser afixado o implicante de termos mínimos [C4+C5+D4+D5] na Figura 12, realiza-se nova pesagem que coloca como mais prioritários os de termos mínimos [D3+E3+D6+E6] e [C6+D6+E6+F6].

No processo de ordenação cai o implicante [C5+D5+C6+E6] com apenas 1 termo mínimo por cobrir.

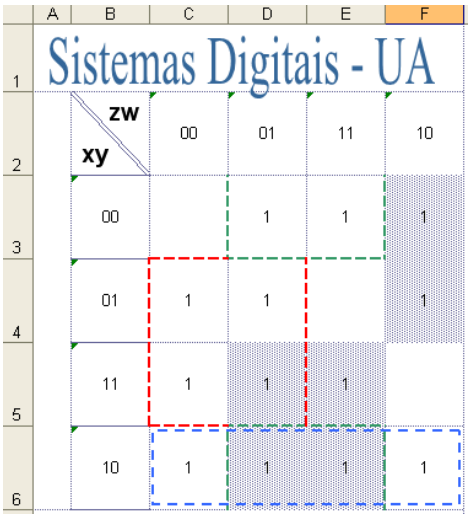

Figura 12 - Exemplo associado à heurística 12

IV. AVALIAÇÃO DO ALGORITMO POR COMPARAÇÃO COM A FERRAMENTA DE REFERÊNCIA (*ESPRESSO*)

A avaliação final da ferramenta exigia uma referência para poder confrontar resultados.

Testar um algoritmo que poderá, como entrada, ter mais de 38000 inputs distintos, num cenário de apenas 4 variáveis independentes, e um número de termos mínimos que pode variar entre os 8 e os 12, obriga a testar a ferramenta em larga escala e a comparar resultados com ferramentas já existentes.

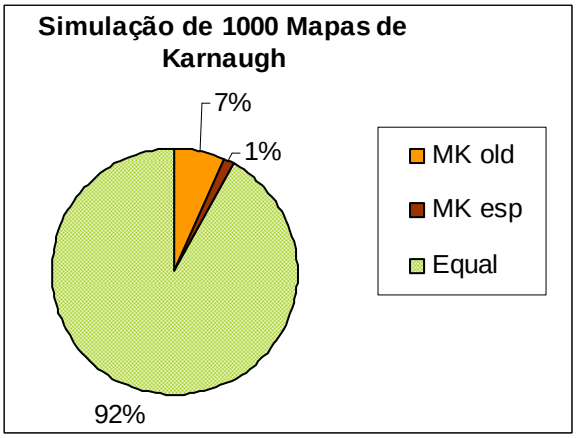

Gráfico 1 - Simulação (1000x; 4 variáveis; nº de mintermos 8-12)

O Gráfico 1 apresenta o resultado da simulação de 1000 mapas de Karnaugh, para uma função aleatória de 4 variáveis independentes e um número variável, entre 8 e 12, de termos mínimos.

Dos resultados obtidos conclui-se que 92% foram eficientemente equivalentes.

A percentagem em que se diferenciaram divide-se em 7% de menor eficiência para a ferramenta em causa e 1% em que foi o *Espresso* a ter pior desempenho.

As baterias de testes efectuados para funções de 4 variáveis independentes permitiram analisar graficamente os resultados, o que se revelou extremamente útil na detecção gráfica dos mapas mal resolvidos. E por isso, mais fácil reflectir esse feedback em futuras alterações.

Embora o algoritmo tenha sido estruturado para permitir simplificar funções até 16 variáveis, a percepção visual das anomalias passa para dimensões difíceis de conceber.

Ainda assim o gráfico 2 ilustra a simulação de 500 mapas de Karnaugh com o aumento de apenas uma variável.

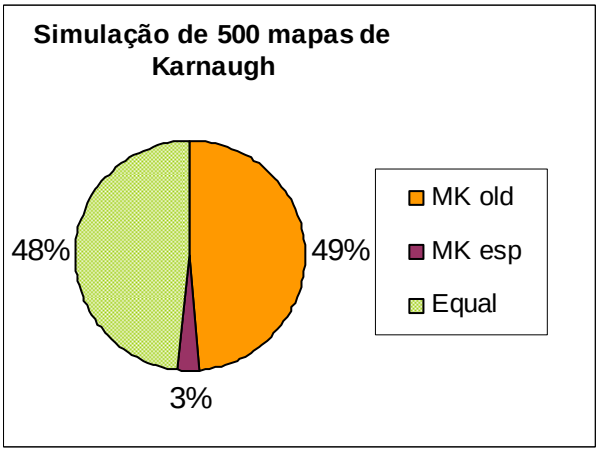

Gráfico 2 - Simulação (500x; 4 variáveis; nº de mintermos: 8-12)

A partir do Gráfico 2 é possível perceber que com o aumento do nº de variáveis independentes os dois algoritmos se degradam, no entanto é notório que na confrontação de desempenho a ferramenta desenvolvida apresenta-se menos robusta.

REFERÊNCIAS BIBLIOGRÁFICAS

[1] Giovanni De Micheli, *Synthesis And Optimization of Digital Circuits*, McGRAW-HILL

[2] John F. Wakerly, *Digital Design Principles and Practices* J. Wiley, 3 Ed. Prentice-Hall.