DVenn – Um *Software* de Auxílio ao Aprendizado de Lógica nos Cursos de Graduação em Computação

Abstract.

This paper presents the DVenn, a software that convert logical expressions in Venn diagrams. The application, implemented in JAVA technology, enables a better understanding during the study for the simplification of logical expressions and helps the student in understanding and correcting its activities. Unlike other tools to simplify, it receives a logical expression of any size, containing up to four different variables, and presents the resulting diagram the student graphically.

Resumo. *Este artigo apresenta o DVenn, um software que mapeia expressões lógicas em diagramas de Venn. A aplicação, implementada na tecnologia JAVA, possibilita uma melhor compreensão durante o estudo de simplificação de expressões lógicas e auxilia o educando na compreensão e correção de suas atividades. Diferente de outras ferramentas de simplificação, esta recebe uma expressão lógica de qualquer tamanho, contendo até quatro variáveis distintas, e apresenta o diagrama resultante ao educando de forma gráfica.*

1. Introdução

Em 1880, o lógico inglês John Venn publicou um artigo com o título "Sobre representação diagramática e mecânica de proposições e raciocínios". Trabalhando na recém-criada área de *Álgebra de Boole* e associando-a com a nova visão da *Teoria dos Conjuntos* desenvolvida por G. Cantor, Venn propôs a ideia de representar as relações entre conjuntos através de configurações de figuras no plano. O objetivo dele, claramente formulado naquele artigo [1], foi:

> *"...antes de mais nada os diagramas servem para auxiliar o olho e a mente graças a natureza intuitiva do seu testemunho..."*

Nos dias atuais, pode-se dizer que o objetivo postulado por Venn foi plenamente alcançado, já que 129 anos mais tarde todos os livros elementares de matemática usam este caminho para introduzir alunos em *Teoria de Conjuntos.* Já nos cursos superiores, os diagramas de Venn representam um método a mais no ensino das funções lógicas.

O aprendizado da simplificação de funções lógicas sempre representou uma dificuldade na disciplina de lógica nos cursos da área da computação. Há três métodos fundamentais para a simplificação: Algébrico (teoremas), Diagramas de Venn e Mapas de Karnaugh. O primeiro, baseado nos teoremas de *Boole*, sempre considerado o mais complexo pelos alunos; o segundo facilita o aprendizado, já que o aluno faz associações

com a teoria dos conjuntos aprendida no ensino básico. O terceiro é considerado de fácil representação, porém em muitos casos as relações de adjacência são de difícil percepção pelos alunos.

Embora os diagramas de Venn sejam considerados fáceis pelos alunos, à medida que a complexidade da expressão lógica aumenta, a representação através dos diagramas de Venn torna-se complexa e mais suscetível a erros, visto que a representação escrita demanda tempo e concentração. Pensando-se nisso, foi projetado e desenvolvido um *software* que recebe a expressão a ser simplificada e, de forma instantânea, retorna ao usuário o diagrama resultante do processo de simplificação.

Este artigo está organizado em cinco seções: a primeira contextualiza o problema; a segunda apresenta três conceitos de raciocínio, fundamentais no ensino da lógica, e como estes conceitos são trabalhados com os diagramas de Venn. A terceira seção demonstra situações do dia a dia da disciplina de lógica, apresentando exemplos de funções lógicas representadas através dos diagramas de Venn. A quarta seção descreve com detalhes a solução em *software* desenvolvida para aprimorar o ensino-aprendizagem da simplificação de funções lógicas através dos diagramas; ao final as conclusões obtidas até o presente momento.

2. Raciocínios

O raciocínio é uma atividade mental intencional, consiste no encadeamento de juizos extraindo deles uma conclusão. Servimo-nos do raciocínio para demonstrar, descobrir, reconstruir e interpretar fatos, convencer, justificar uma tese/teoria, entre outros. No domínio da lógica e da matemática, o encadeamento proposicional é linear, já que se elabora obedecendo estritamente a regras formais [2]. Nesta seção, são apresentados três tipos de raciocínio, classificados como Raciocínio Direto, Indireto e Transitivo. Não é objetivo deste artigo encerrar as possibilidades de classificação dos diversos tipos de raciocínio e sim demonstrar a utilização do diagrama de Venn na representação do raciocínio.

2.1. Raciocínio Direto

No raciocínio direto a ideia subjacente é que Se $A \subset B$ e $x \in A$, então é possível concluir que $x \in B$. A tabela 1 mostra um exemplo de raciocínio direto.

Tabela 1: Exemplo de Raciocínio Direto

Premissa: Todos elementos em A estão também em B .	$p \rightarrow q$: Se um elemento está em A, então ele está em B.
Premissa: x é um elemento em A.	p: x é um elemento em A.
Conclusão: x é um elemento em B.	q: x está em B.

O diagrama que representa este raciocínio é:

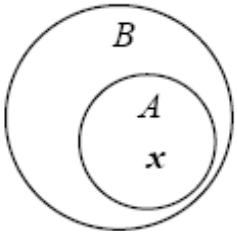

Figura 1: Diagrama de Venn utilizado para demonstrar o Raciocínio Direto

O argumento é válido porque Se A está em B e x está em A, então é possível afirmar com 100% de certeza que x está em B. Mas deve-se prestar atenção: Se $A \subset B$ e $x \in B$, não é possível garantir com 100% de certeza que $x \in A$. O argumento seria inválido. O diagrama da figura 2 mostra que $A \subset B$ e $x \in B$, porém $x \notin A$.

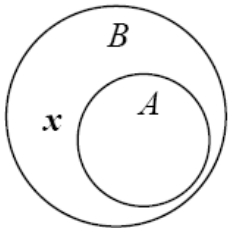

Figura 2: Diagrama de Venn utilizado para demonstrar o Raciocínio Direto

2.2. Raciocínio Indireto

No raciocínio indireto a ideia subjacente é que Se $A \subset B$ e $x \notin A$, então é possível concluir que $x \notin A$. A tabela 2 mostra um exemplo de raciocínio indireto.

Tabela 2: Exemplo de Raciocínio Indireto

Premissa: Todos elementos em *A* estão também em *B* .	$p \rightarrow q$: Se um elemento não está em *A*, então ele está em *B*.
Premissa: *x* não é um elemento em *B*.	*p*: *x* não é um elemento em *B*.
Conclusão: *x* não é um elemento em *A*.	*q*: *x* não está em *A*.

O diagrama que representa este raciocínio é:

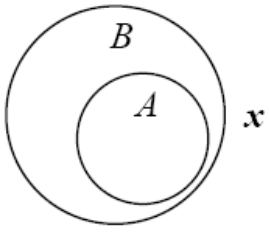

Figura 3: Diagrama de Venn utilizado para demonstrar o Raciocínio Indireto

O argumento é válido porque Se A está em B e x não está em A, então pode-se

ter 100% de certeza que x não está em A. Entretanto Se $A \subset B$ e $x \notin A$, não é possível afirmar com 100% de certeza que $x \notin B$. O argumento seria inválido. O diagrama da figura 4 mostra que $A \subset B$ e $x \in B$, mas $x \notin A$.

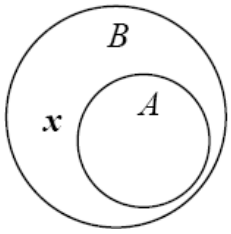

Figura 4: Diagrama de Venn utilizado para demonstrar uma possível indução ao erro no Raciocínio Indireto

2.3. Raciocínio Transitivo

No raciocínio transitivo a ideia subjacente é que Se $A \subset B$ e $B \subset C$, então é possível concluir que $A \subset C$. A tabela 3 mostra um exemplo de raciocínio transitivo.

Tabela 3: Exemplo de Raciocínio Transitivo

Premissa: Todos elementos em *A* estão também em *B* .	$p \rightarrow q$: Se um elemento não está em *B*, então ele não está em *A*.
Premissa: *x* não é um elemento em *B*.	*p*: *x* não é um elemento em *B*.
Conclusão: *x* não é um elemento em *A*.	*q*: *x* não está em *A*.

O diagrama que representa este raciocínio é:

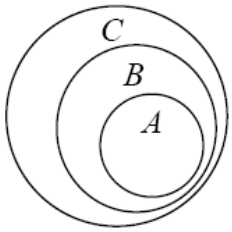

Figura 5: Diagrama de Venn utilizado para demonstrar o Raciocínio Transitivo

O argumento é válido porque Se A está em B e B está em C, então tem-se 100% de certeza que A está em C. Entretanto, mais uma vez pode-se demonstrar que Se $A \cap B$ e $B \cap C$, não será necessariamente verdade que $A \cap C$, como representado na figura 6.

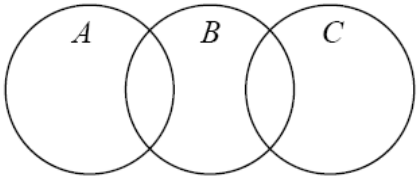

Figura 6: Diagrama de Venn utilizado para demonstrar uma exceção no Raciocínio Transitivo

3. Aplicando o Diagrama de Venn no aprendizado de lógica

Na disciplina de lógica, os diagramas de Venn são comumente utilizados na representação de funções lógicas. Os diagramas de Venn são úteis para reforçar a noção intuitiva sobre conjuntos, principalmente para analisar relações entre os conjuntos e também seus membros. No diagrama de Venn o conjunto universo é representado por um retângulo, isto é, pelos pontos interiores ao retângulo. Qualquer conjunto é desenhado como sendo uma curva fechada (círculo ou elipse), inteiramente contida no retângulo. Os pontos interiores à curva correspondem aos elementos do conjunto. Na figura 7, pode-se visualizar um *slide* de uma aula na disciplina de lógica onde são apresentadas as diversas funções lógicas com duas variáveis possíveis de serem representadas em um diagrama de Venn.

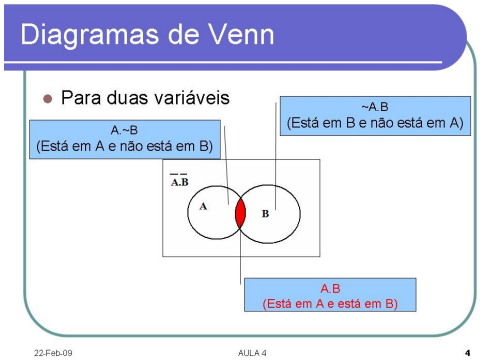

Figura 7: Aplicando Diagramas de Venn na representação de funções lógicas

Outra aplicação dos diagramas está na representação da igualdade de funções lógicas, este fato também é enfatizado nas aulas da disciplina de lógica, conforme pode-se constatar na figura 8:

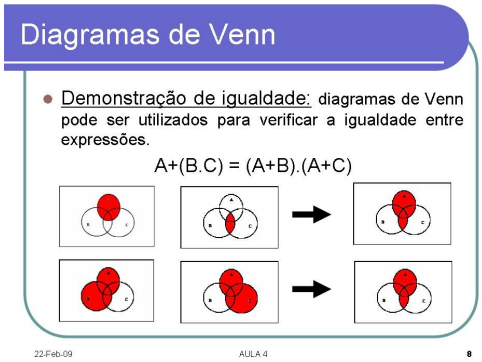

Figura 8: Aplicando Diagramas de Venn na representação de igualdade entre funções lógicas

Os dois exemplos demonstram aplicabilidade dos diagramas de Venn no ensino de funções lógicas, simplificando por vezes a compreensão do educando, muitas vezes não afeito a utilizar as tabelas-verdade. Entretanto, à medida que as funções lógicas tornam-se mais complexas, a representação por meio dos diagramas de Venn produz no educando uma aversão a este tipo de representação, fazendo-o abandonar a sua utilização e impedindo-o de aproveitar os benefícios propiciados pelo diagrama. Pode-se exemplificar esta afirmação apresentando a seguinte função lógica:

((A . B . C') + (A'. B' . C)) . A

A representação através do diagrama de Venn resulta na figura 9.

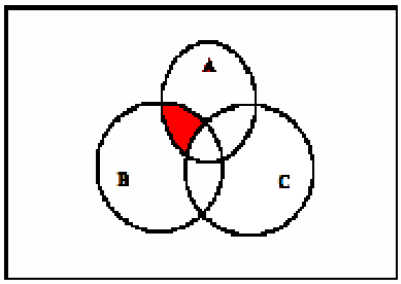

Figura 9: Diagrama de Venn representando a função lógica ((A . B . C') + (A'. B' . C)) . A

Verificou-se, pela experiência em sala de aula, que os alunos têm dificuldades de compreender e aceitar esta técnica quando as expressões lógicas são mais complexas. Desta forma, foi proposto o desenvolvimento de um software que propiciasse um melhor aprendizado dos diagramas de Venn. A próxima seção descreve a ferramenta desenvolvida.

4. O protótipo do *Software DVenn*

O DVenn originou-se da falta, precariedade e alto custo dos *softwares* de auxílio ao estudo de álgebra booleana. É conhecida a dificuldade que estudantes possuem para representar uma expressão booleana em diagramas de Venn, a fim de simplificá-la. Muitas das aplicações construídas com o fim de ajudar a representar dados apenas oferecem uma interface para que o usuário "monte" o seu diagrama, sem cálculos ou resolução de expressões.

As ferramentas de simplificação de expressões lógicas, normalmente não exibem seus resultados em diagramas, o que as torna impróprias no aprendizado deste tipo de representação, pois não fornecem ao estudante o diagrama, para fins de correção de um exercício. Além disso, é também conhecido que o desenho de um diagrama de Venn qualquer demanda tempo e certa concentração, exigindo energia e disposição do professor que ministra este conteúdo, além de, em caso de erro, praticamente todo o trabalho do desenho deverá ser refeito.

Apesar de existirem ferramentas que simplificam expressões, como o *Binary Function Simplification* [3], e que auxiliam no desenho de um diagrama de Venn [4], como o *SmartDraw*, não existe uma ferramenta com ambas as funcionalidades, permitindo a otimização do tempo de aprendizado. Assim, durante uma aula de lógica booleana surgiu a ideia de desenvolver um *software* que simplificasse expressões booleanas e representasse seus dados em forma de diagrama de Venn, rápido, portável e acessível. A figura 10 mostra uma tela do *Software* DVenn.

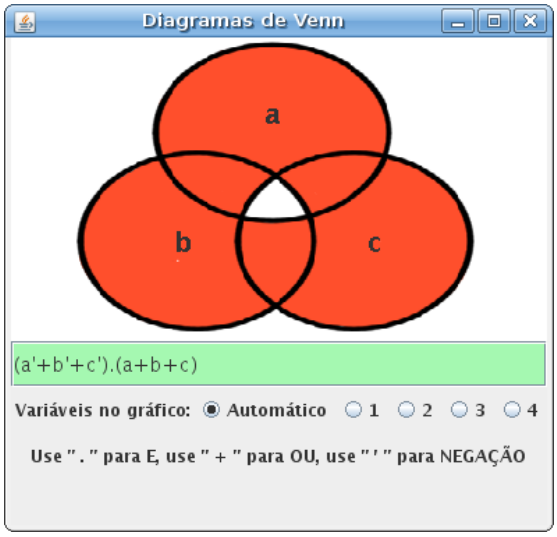

Figura 10: Interface do Dvenn

4.1. A funcionalidade do DVenn

No início do projeto, tinha-se em mente apenas a representação de diagramas através de expressões booleanas já simplificadas e que contivessem três variáveis. Porém, por demanda do público teste – professores e alunos de lógica – foi inserido à aplicação um módulo de simplificação de expressões, que recebe polinômios booleanos e os exibe em forma de diagrama. Para isto, a convenção para representação de operações booleanas foi inserida de forma dinâmica na aplicação, já que existem várias notações para a representação de operações. Optou-se pela notação onde o sinal da operação OU é o sinal de adição (+), o sinal da operação E é o sinal de multiplicação (.) e as negações de sinal são expressadas por uma aspa simples (').

No campo de texto, existe a expressão analisada pela aplicação, e acima, o diagrama obtido após sua simplificação (figura 10). A aplicação detecta o número de variáveis e produz um diagrama de acordo, caso a opção “Variáveis no gráfico:” esteja com a opção “Automático” selecionado, conforme pode-se visualizar na figura 11.

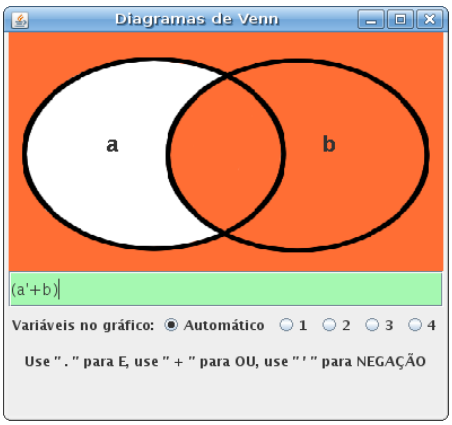

Figura 11: Detecção automática de variáveis

É também possível “forçar” o número de variáveis do diagrama, porém as variáveis excedentes terão seu valor substituído por 1, conforme pode-se visualizar na figura 12.

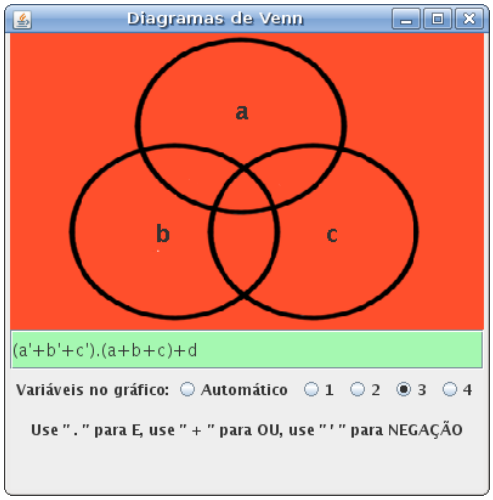

Figura 12 – Especificando o número de variáveis no diagrama

Somente são aceitas expressões válidas, ou seja, expressões booleanas que possam ser resolvidas. Caso a aplicação detecte erro na expressão, o campo de texto torna-se vermelho e um aviso é emitido na tela, conforme pode-se visualizar na figura 13.

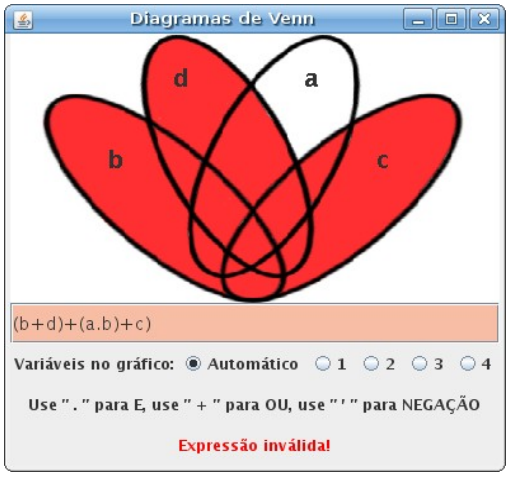

Figura 13: Exemplo de erro

4.2. A arquitetura da aplicação

A fim de atingir o maior número de estudantes possível, uma preocupação foi a escolha da linguagem de programação da aplicação. Otimizando o desenvolvimento, aumentando o alcance e a inclusão através da *web* e sem a necessidade de compilação de código – pouco dominada entre os estudantes da disciplina de lógica (que consta na grade curricular no primeiro semestre do curso) – foi então escolhida a linguagem Java, da *Sun Microsystems*, que assegura portabilidade através de sua máquina virtual, além de um sistema de *Applet* que permite a inclusão direta da aplicação em páginas *web*, sem necessidade de *download* e instalação de um cliente local.

Para implementação da interface gráfica, foi utilizada a biblioteca *Swing* e para a representação no diagrama, foi utilizado Java2D, obtendo melhor performance na renderização das imagens. A aplicação também possui implementação de *Applet*, o que possibilita inclusão direta em uma página *web*. Todas as bibliotecas utilizadas são

padrões e distribuídas junto à máquina virtual Java, garantindo compatibilidade total dos recursos utilizados, independente de sistema operacional e *browser*.

A fase seguinte do desenvolvimento foi o desenho dos diagramas representando expressões lógicas com até quatro variáveis. Foram criados desenhos dos diagramas através do *software* de desenho Krita [5], preeenchidas as lacunas do diagrama com cores adstringentes, e separadas por camadas. Cada camada é então uma única imagem, e a formação do diagrama na tela é montada sobrepondo as imagens de acordo com a expressão no método de pintura do painel Java2D. Este painel é então mostrado pela janela criada pela biblioteca *Swing*, como mostra a figura 14.

Figura 14: Arquitetura do DVenn

4.3. Componentes do DVenn

Baseando-se em um sistema de palavras binárias, com o número de *bits* igual ao número de possibilidades representadas pelo diagrama, cada *bit* representa uma seção do diagrama, como mostra a imagem apresentada na figura 15.

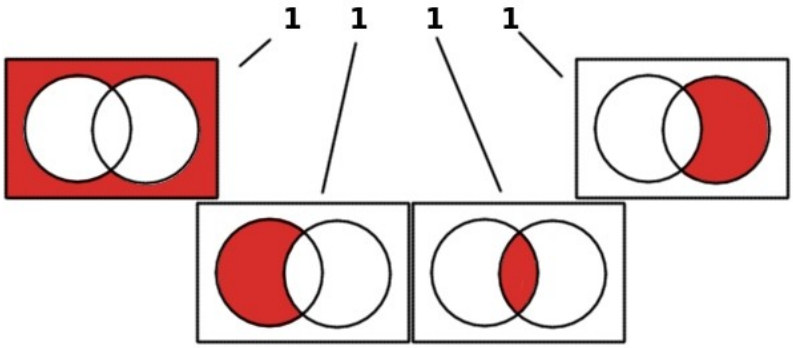

Figura 15: Representação da associação de bits a seções do diagrama

Como cada variável do diagrama pode representar uma ou mais seções do diagrama, cada variável é uma palavra binária por si só, contendo as seções de diagrama pintadas quando aparecem sozinhas. Por exemplo, em um diagrama para a expressão de duas variáveis A+B, a variável A é mapeada com o valor 0110 e a variável B com o valor 0011.

Quando o usuário insere uma expressão lógica, o programa encarrega-se de substituir as variáveis por seus devidos valores binários. Após, esta nova expressão binária é repassada a um método responsável por sua resolução. Este método percorre toda a expressão e a resolve a partir da hierarquia mais alta, nos parênteses mais internos, para a mais baixa, parênteses mais externos.

5. Conclusões

Neste trabalho, o *software* DVenn foi implementado com o objetivo de ser agente

catalisador no processo de ensino-aprendizagem da simplificação e representação de funções lógicas. A utilização do *software* DVenn no dia a dia da disciplina de Lógica nos cursos de Computação representa uma alternativa viável e acessível. Adicionalmente, neste artigo pôde-se demonstrar o nível de abstração propiciado por este tipo de *software*. Sem dúvida, este nível de informação favorecerá a aprendizagem e representará um passo a mais em direção à transformação do processo educacional.

Apesar dos resultados alcançados ainda estarem sendo tabulados, tendo em vista sua recente criação, o fato da sua prévia utilização não impediu uma pré-avaliação da sua eficácia e já permitiu que o professor realizasse intervenções no processo para aprimoramento da disciplina. Certamente, se as conclusões já tivessem sido geradas com maior antecedência, poderiam ter favorecido resultados imediatos, propiciando melhorias ao processo e minimizando o tempo de aprendizado da simplificação de funções lógicas contribuindo para o aprimoramento da disciplina.

Referências

[1] Solecky, A. (2009). Diagramas de Venn. [online]: http://www.andsol.org/portugues/mat/venn.html

[2] Teodósio, J. (2007). Tipos de Raciocínio Indutivo. [online]: http://br.geocities.com/grupo10b/JQ.htm

[3] University of Northern Colorado (1998). Boolean Function Simplification. [online]: http://hopper.unco.edu/KARNAUGH1.1/Function.html

[4] SmartDraw.com (2009). Easy Venn Diagram Software. [online]: http://www.smartdraw.com/specials/venn.htm?id=2087

[5] The KOffice Projct: Krita (2009). KOffice image editing program. [online]: http://www.koffice.org/krita