**UNIVERSIDADE FEDERAL DO ESPÍRITO SANTO**
**CENTRO DE CIÊNCIAS EXATAS, NATURAIS E DA SAÚDE**
**DEPARTAMENTO DE COMPUTAÇÃO**

**JULIANA DALLAPICULLA BASILIO DA SILVA**

# JDEDUTOR: UMA FERRAMENTA AUXILIAR DE DEDUÇÃO DE SENTENÇAS LÓGICAS PARA AULAS DE LÓGICA MATEMÁTICA

ALEGRE – ES
2016

JULIANA DALLAPICULLA BASILIO DA SILVA

# JDEDUTOR: UMA FERRAMENTA AUXILIAR DE DEDUÇÃO DE SENTENÇAS LÓGICAS PARA AULAS DE LÓGICA MATEMÁTICA

Trabalho de conclusão de curso apresentado ao Departamento de Computação do Centro de Ciências Exatas, Naturais e da Saúde da Universidade Federal do Espírito Santo, como requisito parcial para obtenção do grau de Bacharel em Ciência da Computação.

Orientador: Prof. M.Sc. Jacson Rodrigues Correia da Silva.

ALEGRE - ES

2016

JULIANA DALLAPICULLA BASILIO DA SILVA

# JDEDUTOR: UMA FERRAMENTA AUXILIAR DE DEDUÇÃO DE SENTENÇAS LÓGICAS PARA AULAS DE LÓGICA MATEMÁTICA

Trabalho de conclusão de curso apresentado ao Departamento de Computação do Centro de Ciências Exatas, Naturais e da Saúde da Universidade Federal do Espírito Santo, como requisito parcial para obtenção do grau de Bacharel em Ciência da Computação.

Aprovado em 28 de Novembro de 2016.

**Banca Examinadora**

Jacson R. C. Silva

**Prof. M.Sc. Jacson Rodrigues Correia da Silva.**
**Universidade Federal do Espírito Santo**
**Orientador**

Rodrigo F. S.

**Prof. M.Sc. Rodrigo Freitas Silva.**
**Universidade Federal do Espírito Santo**

J. Pirovani

**Profª. M.Sc. Juliana Pinheiro Campos Pirovani**
**Universidade Federal do Espírito Santo**

Alegre – ES
2016

## Agradecimentos

À Deus, que acima de qualquer crença, foi o responsável por toda inteligência, paciência, saúde e garra concebida para que esta etapa fosse concluída, apesar de todas barreiras que apareceram no caminho.

À minha família, e em especial meus pais, Maria Antônia e Marceano, que sempre se mostraram fortes apesar da distância e das dificuldades enfrentadas. Eles foram minha força quando o desejo pela desistência bateu à porta. À minha irmã, Jaqueline, que sempre se mostrou presente, mesmo de longe. Ao meu tio César (Didim), que foi o responsável por descobrir da minha aprovação na universidade, e o primeiro à me estimular correr atrás dos meus sonhos. Ao meu tio Alexandre que, junto com meus pais, acreditou desde o primeiro momento que esta trajetória seria realizada na primeira oportunidade apresentada.

Uma imensa gratidão por Bhetynna, Priscila, e Marília que foram minha família e souberam ter a paciência necessária para me guiar desde de os primeiros momentos em Alegre, e que até hoje seguem comigo nesta caminhada. Gratidão por Yan e Barbara, amigos que reencontrei, e primeiros a me acolherem na cidade de Alegre, no auge do desespero de mudar para uma nova rotina. Ao meu amigo Caíque, que também foi muito importante para que eu me reintegrasse à rotina de Alegre após o intercambio.

Aos meus amigos do intercambio, principalmente César, que me mostraram que não existe distância, saudade, ou medo, quando se tem amigos que te passam confiança e estão ao seu lado para qualquer desafio. À Richelle, companheira que na reta final desta faculdade me ajudou a persistir no meu sonho.

Aos amigos de Colatina, em especial Jessica, Dani, Tonho, e Camu, que nunca deixaram de me receber de braços abertos quando a saudade de casa falava mais alto que a distância.

Ao Jeiks, por ter me aceitado para fazer parte do seu time tão disputado de orientandos. Obrigada pela paciência e confiança depositada ao longo desses meses.

À Universidade Federal do Espírito Santo, seu corpo docente e demais funcionários por terem proporcionado o conhecimento e espaço necessários.

Aos demais amigos, servidores, e familiares que estiverem comigo nesta caminhada

# Resumo

Ferramentas online tem ganhado espaço com a facilidade de acesso à Internet nos dias de hoje. Aplicativos e softwares são cada dia mais indispensáveis não só para a comunicação, mas também para atividades acadêmicas. Programas desenvolvidos especificamente para disciplinas que possuem um alto grau de dificuldade, ou alto índices de reprovações, ganham mais espaço com o objetivo de quebrar barreiras no aprendizado. A Lógica Computacional é uma disciplina básica para o cursos de Computação. Porém, apesar de sua importância para o ensino superior, esta disciplina não é apresentadas, atualmente, na educação de base, o que colabora para um aumento nas dificuldades para entender seus conceitos.

A dedução de premissas lógicas faz parte da Lógica Computacional. Este trabalho visa a criação de um software para tais deduções com o objetivo de ampliar as formas de aprendizagem da disciplina e diminuir as barreiras para o aprendizado das mesmas. A ferramenta deste trabalho, chamada JDedutor, exibe não só as premissas fornecidas pelo usuário, mas também as regras reconhecidas, e a possível conclusão para a dedução. Foram escolhidas para desenvolvimento neste trabalho as regras de Dupla Negação, Modus Tollens, Modus Ponens, e a Simplificação por serem constantemente utilizadas no meio computacional dentro do desenvolvimento de programas. Uma análise qualitativa com os alunos dos cursos de Ciência da Computação e Sistemas de Informação, que estão matriculados nas disciplinas de Lógica Computacional I e II do Centro de Ciências Exatas, Naturais e da Saúde (CCENS) da Universidade Federal do Espírito Santo. Esta análise foi responsável para a avaliar a repercussão e a utilidade da ferramenta após sua apresentação.



**Palavras chaves:** Aplicativos educacionais, desenvolvimento de software, Lógica Computacional, dedução lógica

## Abstract

Online tools have gained space with an ease of Internet access these days. Applications and software are increasingly indispensable not only for communication, but also for academic activities. Programs developed specifically for courses that have a high degree of difficulty, or high rates of disapproval, have gained more space in order to break down barriers in learning. Computational logic is a basic course for computing courses. However, despite its importance to higher education, this course is not currently presented in basic education, which contributes to an increase in the difficulties to understand its concepts.

The deduction of logical premises is part of Computational Logic. This paper aims at the creation of software for deduction of logical premises with the objective of broadening the learning forms of the course and reducing the barriers to learning them. The tool in this paper displays not only the code provided, but also the rules recognized, and the possible conclusion for the deduction. The rules of Double Negation, Modus Tollens, Modus Ponens, and Simplification were chosen for development in this paper because they are constantly used to the development of software. A qualitative analysis with the students of the courses of Computer Science and Information Systems, which are enrolled in the courses of Computational Logic I and II of the Centro de Ciências Exatas, Naturais e da Saúde (CCENS) da Universidade Federal do Espírito Santo. This analysis was responsible for evaluating the repercussion and usefulness of the tool after its presentation.

**Sumário**

1. Introdução ..... 9
1.1. O problema e a sua importância ..... 10
1.2. Objetivos ..... 12
1.2.1. Objetivo geral ..... 12
1.2.2. Objetivos específicos ..... 12
2. Revisão de Literatura ..... 13
3. Metodologia ..... 23
4. Análise dos Resultados ..... 29
5. Conclusão ..... 41
6. Referências ..... 43
7. Apêndices ..... 46

## LISTA DE FIGURAS

Figura 1 - Aprovações e reprovações em Lógica Computacional I ..... 11
Figura 2 - Modus Ponens ..... 20
Figura 3 – Modus Tollens ..... 20
Figura 4 - Dupla Negação ..... 21
Figura 5 - Simplificação 1 ..... 21
Figura 6 - Simplificação 2 ..... 21
Figura 7 - Demonstração das produções para leitura da gramática ..... 25
Figura 8 - Tela inicial do JDedutor ..... 26
Figura 9 - Tela após arquivo selecionado ..... 27
Figura 10 - Tela após executar o arquivo ..... 27
Figura 11 - Porcentagem para resposta 1 de Lógica Computacional I ..... 30
Figura 12 - Porcentagem para resposta 1 de Lógica Computacional II ..... 30
Figura 13 - Porcentagem para resposta 2 de Lógica Computacional I ..... 31
Figura 14 - Porcentagem para resposta 2 de Lógica Computacional II ..... 31
Figura 15 - Porcentagem para resposta 3 de Lógica Computacional I ..... 32
Figura 16 - Porcentagem para resposta 3 de Lógica Computacional II ..... 32
Figura 17 - Porcentagem para resposta 4 de Lógica Computacional I ..... 33
Figura 18 - Porcentagem para resposta 4 de Lógica Computacional II ..... 34
Figura 19 - Porcentagem para resposta 5 de Lógica Computacional I ..... 34
Figura 20 - Porcentagem para resposta 5 de Lógica Computacional II ..... 35
Figura 21 - Porcentagem para resposta 6 de Lógica Computacional I ..... 35
Figura 22 - Porcentagem para resposta 6 de Lógica Computacional II ..... 36
Figura 23 - Porcentagem para resposta 7 de Lógica Computacional I ..... 36
Figura 24 - Porcentagem para resposta 7 de Lógica Computacional II ..... 37
Figura 25 - Porcentagem para resposta 8 de Lógica Computacional I ..... 37
Figura 26 - Porcentagem para resposta 8 de Lógica Computacional II ..... 38
Figura 27 - Porcentagem para resposta 9 de Lógica Computacional I ..... 38
Figura 28 - Porcentagem para resposta 9 de Lógica Computacional II ..... 38
Figura 29 - Porcentagem para resposta 10 de Lógica Computacional I ..... 39
Figura 30 - Porcentagem para resposta 10 de Lógica Computacional II ..... 39
Figura 31 - Porcentagem para resposta 11 de Lógica Computacional I ..... 40
Figura 32 - Porcentagem para resposta 11 de Lógica Computacional II ..... 40

# 1. Introdução

O avanço tecnológico e a popularização de dispositivos eletrônicos são fatores que tem causado impactos e mudanças em diversos aspectos do nosso cotidiano. De acordo com Dim e Rocha (2011), a área da educação tem usado esses dois fatores para diversificar e dinamizar as formas de aprendizado, visando assim, uma otimização das metodologias de ensino em todos os níveis de educação. Santos e Costa (2006) informam que o desenvolvimento de softwares e de aplicações utilizadas como ferramentas educacionais tem sido de suma importância na minimização dos problemas de aprendizado enfrentados por muitos cursos.

Como exemplo de cursos que enfrentam problemas particulares de metodologias de ensino, podemos citar os cursos da área tecnológica. Segundo SILVA et al. (2014), a apresentação de disciplinas do curso de computação limita-se apenas a cursos superiores ou técnicos. Disciplinas como programação, por exemplo, são importantes para o desenvolvimento de raciocínio lógico-matemático e computacional. Uma vez que essas disciplinas passam a ser ministradas somente no ensino superior, o primeiro contato torna-se algo muito mais complicado e de difícil aprendizado do que outras já cursadas (SILVA et al., 2014). Como estas disciplinas são base para o restante do curso, seus aproveitamentos são essenciais para que o aluno possa continuar seu aprendizado durante o restante do curso (DIM e ROCHA, 2011).

Adicionalmente, Santos, Franco e Corsini (2015) expõem que desenvolver softwares torna-se mais abstrato quando é ensinado com a criação de pseudocódigos feitos no papel. Este fator torna a prática de programação algo difícil de ser visualizado ou entendido, já que o funcionamento de um programa é difícil de ser estimado ou previsto sem que se tenha o programa propriamente dito em funcionamento. Tomando como foco os problemas anteriormente mencionados, essas novas ferramentas tem sido utilizadas para auxiliar no aprendizado de determinadas disciplinas que são apenas apresentadas no ensino superior dos cursos destas áreas.

Outra disciplina base dos cursos tecnológicos é a disciplina de Lógica Matemática, também chamada de Lógica Computacional, sendo seu conteúdo

importante para o aluno aprender os conceitos da arquitetura de um computador e dos programas desenvolvidos para ele. A Lógica Computacional é a responsável, dentro da área Computacional, por abranger os conceitos de Lógica Proposicional (SILVA, FINGER e MELO, 2006, p.8). Por sua vez, a Lógica Proposicional relaciona eventos sentenças declarativas com conectivos lógicos que fazem parte da Linguagem Proposicional (SOUZA, 2008, p.4).

Souza, 2008, explica que a Linguagem Proposicional é formada por uma alfabeto lógico contendo símbolos de verdade e proposicionais, e conectivos lógicos. As sentenças são conhecidas como proposições dentro da Lógica Proposicional e são representadas pelos símbolos proposicionais (SOUZA, 2008, p.6). A combinação de símbolos com os conectivos formam as premissas que são responsáveis por evidenciar ou garantirem uma conclusão (MORTARI, 2001). Estas premissas são a base para que as deduções lógicas, sejam possíveis. A dedução lógica tem seus conceitos ministrados durante o aprendizado da disciplina de Lógica Computacional.

Pela análise apresentada na seção 1.1, pode-se perceber a dificuldade que diversos alunos têm com a disciplina.

### 1.1. O problema e a sua importância

Como explicam Santos e Costa (2006), os conhecimentos sobre os fundamentos lógicos e matemáticos são de extrema importância para os cursos da área computacional. Santos, Franco e Corsini (2015) apresentam que o ensino da Lógica Computacional ainda é visto como algo muito abstrato.

Para verificar o aproveitamento das disciplinas pelos alunos, foi analisado o índice de reprovação de três turmas do Departamento de Computação do Centro de Ciências Exatas, Naturais e da Saúde (CCENS) da Universidade Federal do Espírito Santo (UFES).

A primeira turma avaliada possuía 65 alunos dos cursos de Ciência da Computação e de Sistemas de Informação do Campus. A disciplina, lecionada no período de 2013/2, era Lógica Computacional I. Como é apresentado na Figura 1, do total de alunos, apenas 17 foram aprovados. Dos demais, 26 reprovaram por não atingirem a nota necessária para aprovação, e outros 22 reprovaram por falta. Houve então, um índice de 40% de reprovações por notas neste período, totalizando

aproximadamente 70% de reprovados por critério de notas ou faltas. Estes dados correspondem a uma estatística baixa de entendimento da disciplina.

Já no período de 2014/2, 41 alunos cursaram Lógica Computacional I (Figura 1). Nesta turma, 19 foram aprovados, oito reprovaram por não conseguiram nota, e os outros 14 também reprovaram, porém, por desistência. Apesar do índice de aprovações ter sido de aproximadamente 46%, o que se mostrou maior que da turma anteriormente analisada, este índice ainda representa menos da metade da turma, o que também não é um bom sinal de aproveitamento por parte dos alunos da disciplina.

A terceira turma analisada foi a de Lógica Computacional I lecionada no período de 2015/1. Como mostra a Figura 1, 27 alunos foram matriculados na disciplina. 16 alunos foram aprovados, o que mostra um melhor aproveitamento, seis reprovaram por notas insuficientes e cinco por falta. Houve um índice muito mais alto se comparado às turmas anteriores. Aproximadamente 59% dos alunos conseguiram notas suficientes..

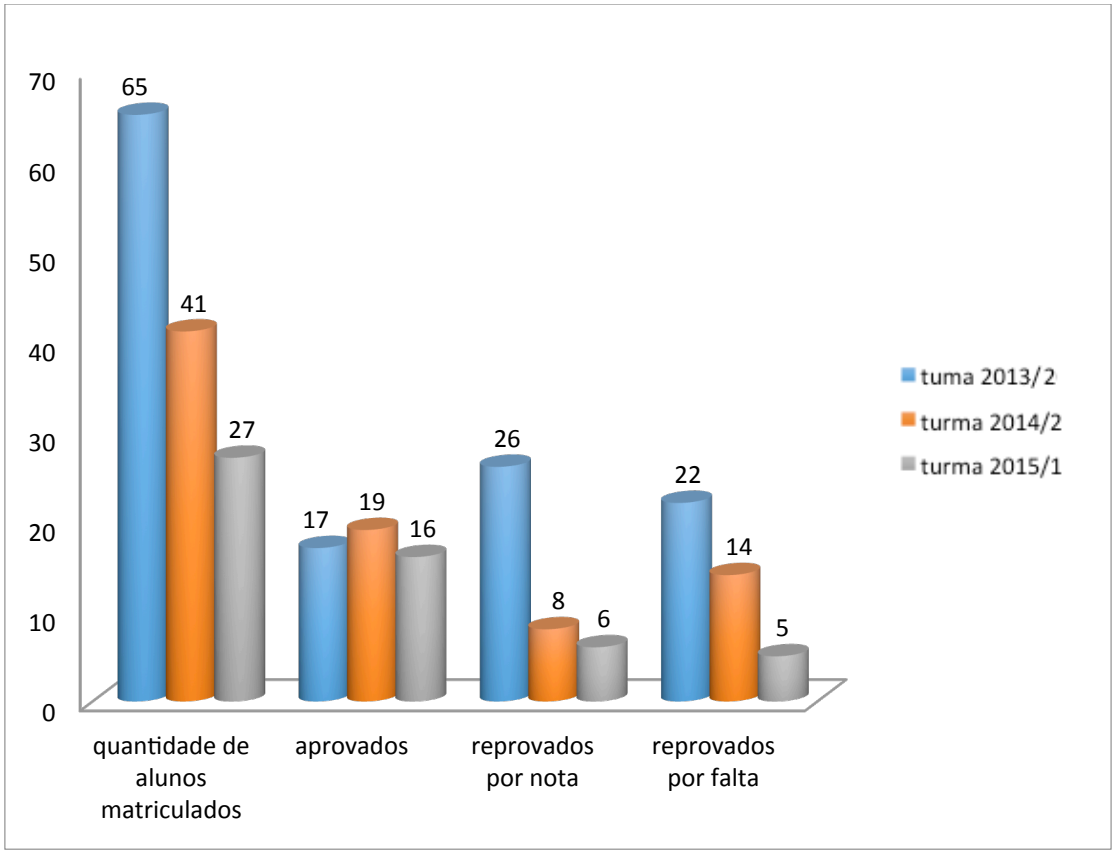


Figura 1 - Aprovações e reprovações em Lógica Computacional I

Outro fato a ser considerado é que alguns dos alunos presentes nesta pesquisa cursaram esta disciplina mais que uma vez para conseguir a aprovação. Por exemplo, como esta disciplina é obrigatória dos primeiros períodos de Ciência da Computação do Campus estudado, se os alunos possuírem matrículas com os quatro primeiros números diferentes do ano de oferta da disciplina significa que não estão na sua turma original. Assim, concluiu-se que estes prestaram a disciplina pelo menos uma vez e ainda assim não conseguiram pontuação suficiente para a aprovação.

Assim, esse trabalho propôs criar uma ferramenta de ensino e aprendizagem, nomeada JDedutor, para ajudar na dedução de sentenças lógicas, visando ajudar os alunos no aprendizado de Lógica Matemática e fornecer auxílio aos professores em seu ensino. O escopo foi o processo de dedução de proposições, fornecendo um apoio ao seu ensino tradicional (SANTOS; FRANCO; CORSINI ,2015).

## 1.2. Objetivos

### 1.2.1. Objetivo geral

Criar uma ferramenta de ensino e aprendizagem para auxiliar na dedução de sentenças lógicas nas aulas de Lógica Matemática e avaliá-la de forma qualitativa.

### 1.2.2. Objetivos específicos

Para alcançar o objetivo principal, foram realizados os seguintes objetivos:

1. Escolha da linguagem de programação apropriada.
2. Definição das regras que poderão ser deduzidas e uma interface adequada para a ferramenta.
3. Definição do escopo da ferramenta.
4. Desenvolvimento de uma análise léxica, sintática e semântica utilizando a linguagem Java.
5. Criação de um questionário para a avaliação qualitativa da ferramenta.
6. Realização de testes na ferramenta desenvolvida.
7. Execução de testes da ferramenta pelos alunos das disciplinas de Lógica Computacional I e Lógica Computacional II.
8. Análise dos resultados baseando-se na resposta dos alunos ao questionário.

# 2. Revisão de Literatura

A educação tem conciliado cada vez mais os recursos oferecidos pelo desenvolvimento tecnológico com suas metodologias de ensino. Katz (2010) expõe que a inserção de computadores no aprendizado vem concretizando-se pelo desenvolvimento que os mesmos sofreram com o passar do tempo. Antes vistos como máquinas de difícil uso e de lento processamento de informações, hoje, são muito mais potentes, baratos e com interfaces interativas. Com isso, graças a tecnologia, a modernização do ensino oferece uma variedade de funcionalidades que muitos livros não conseguem oferecer, como recursos gráficos e animações (DIM e ROCHA, 2011).

Katz (2010) abrangeu a evolução do computador e todas suas crescentes funcionalidades nos últimos anos, tomando como foco a implantação das aplicações em nuvem. O conceito de nuvem representa a disponibilização online de toda a infraestrutura de comunicação na forma de serviços para os usuários (RUSHCEL et al., 2010). Em prol de ilustrar a diferença entre o que era considerada tecnologia no passado e como essa imagem evoluiu para os dias atuais, a Tabela 1 exibe uma comparação entre computadores no passado com computadores nos dias atuais adaptada para a língua portuguesa. Ela foi apresentada por Katz (2010) e contém uma lista comparando oito aspectos que mostram a evolução dos computadores

Tabela 1 – Comparação entre Computadores no passado e nos dias atuais ( KATZ, 2010).

| De | Para |
|---|---|
| Computadores são caros e difíceis de encontrar | Computadores com preços acessíveis e encontrados em todo lugar |
| Computação e Computadores são isolados | Computadores se conectam |
| Computadores são estacionários | Computadores são móveis |
| Computação em rede é um esforço | Rede de Computador é um recurso amplamente detido e atividade que |

| | |
|---|---|
| profissional em escala empresarial | se estende por consumidores de todas as idades e uma ampla gama de papéis pessoais e profissionais |
| Computação, rede e capacidade de armazenamento de dados são fixos e devem ser administrados para um crescimento. | Computadores, redes, e capacidade de armazenamento podem ser virtualizados, compartilhados, e estimulados ou não de acordo com a demanda |
| Sistemas de Informação, recursos e serviços podem ser organizados, montados e mediados por empresas | Sistemas de Informação, recursos e serviços podem ser organizados, montados e mediados de qualquer lugar por qualquer pessoa |
| Recursos de redes de informação são escassos | Recursos de redes de informação são abundantes |
| Encontrar informações de rede requer habilidades específicas e treinamento | Encontrar informações de rede é relativamente fácil e natural |

Um estudo realizado por Ellison e Wu (2008) revela a importância destas ferramentas para o aprendizado. O estudo com objetivo de mostrar os impactos do uso de tecnologias de comunicação envolveu 68 estudantes, numa faixa etária de 16 à 28 anos, da Universidade de Midwesterm. Estes estudantes foram submetidos a testes utilizando a antiga metodologia (respostas escritas em papéis), e testes com suas respostas sendo postadas em blogs podendo ser vistas e discutidas pelos demais colegas. Como conclusão da análise dos resultados, Ellison e Wu (2008) constataram que essas tecnologias de comunicação não podem ser consideradas a fonte de um conhecimento mas sim como mecanismos auxiliares. Eles seriam de grande importância para interação entre estudantes, pois proporcionam um ambiente mais dinâmico e possivelmente mais confortável para os envolvidos.

Liu et al. (2012) apresentaram o conceito de Web 2.0 e seu uso na educação superior. A definição fornecida por O'Reilly (2005) e citada por Liu et al. (2012), diz que Web 2.0 são as aplicações de web que promovem a interatividade e a troca de informação entre os usuários. Liu. et al. (2012) tem como resultado as cinco aplicações mais utilizadas no ensino superior nos anos de 2007 a 2009, sendo elas

blogs, wikis, podcasts, ambientes virtuais, e redes sociais. Como características comuns dessas tecnologias de Web 2.0, compartilhamento de fotos e vídeos demonstrativos e educativos e chats para conversas em tempo real são vistas como vantagens que complementam o desempenho didático dos estudantes.

Budas et al. (2010) realizou um estudo baseado na plataforma de rede sociais Ning (NING, 2010). Nele, o autor relata a importância que os meios de comunicação que utilizam internet tem em relação a transmissão de conhecimento. Budas et al. (2010) chama de “software sociais” as ferramentas que proporcionam a interação entre pessoas sem limitar idade ou países. Blogs, sites de redes sociais e podcasts são exemplos dessas aplicações onde os usuários podem compartilhar seus aprendizados e experiências, e podem também aprender com o que foi compartilhado por outros usuários. Tomando Ning como o foco do estudo, Budas et al. (2010) afirma que esta plataforma oferece recursos amplia as possibilidades de comunicação entre professores e alunos.

Neil (2012) mostra que durante os últimos anos, objetos de aprendizagem desenvolvidos para plataformas online e móveis vem ganhando popularidade. Este fato ocorre devido a praticidade de acesso que não fica limitada à apenas um local físico, mas que dá liberdade geográfica para que se acesse os conteúdos educacionais disponíveis nesses dois tipos de plataformas. Bento e Cavalcante (2013) realizaram um estudo de caso com professores do Ensino Médio de uma escola estadual publica, do Vale do Paraíba do Sul. Neste estudo, Bento e Cavalcante (2013) analisaram o uso do celular como recurso de auxílio nas salas de aula do ponto de vista dos professores. Os resultados foram positivos em relação ao uso do celular, uma vez que os professores concordam com a ideia de que o este dispositivo não tenha sua utilidade vista apenas para entretenimento.

Percebendo a vantagem da integração entre ensino e tecnologia, muitos desenvolvedores passaram a ter como suas criações, softwares ou aplicativos voltados para o ambiente educacional. Com isso, esses aplicativos começaram a ser classificados de acordo com seus objetivos e formas de manuseio. Bertoldi e Ramos (1999) classifica os softwares educativos em modalidades de acordo com determinadas características em comum:

1. Exercício e prática: Trata-se de um software que pode fortalecer o ensino em sala e aumentando as habilidades do aluno por meio de resolução de

exercícios. Os alunos selecionam problemas ou questões as resolvem quantas vezes forem necessárias para o entendimento. Esta maneira é de mais fácil desenvolvimento e é considerada “a forma tradicional que os computadores tem sido utilizados em educação” (BERTOLDI e RAMOS, 1999).

2. Tutoriais: Também são vistos como apoio ou um guia ao aprendizado dentro de classe, porém são mais complexos que os exercícios pois oferecem uma interface mais atrativa com gráficos e animações, e sinais sonoros. Fornecem a possibilidade do usuário escolher o nível do dificuldade para facilite a aplicação das habilidades já adquiridas e das novas (BERTOLDI e RAMOS, 1999).
3. Simulações e Modelagem: Representam um modelo simbólico da realidade para caracterizar os aspectos principais de um fenômeno. Ou seja, essas ferramentas são utilizadas depois que o aprendizado já foi apresentado. Uma importante característica é que essa metodologia funciona como uma ponte entre o abstrato e o real. Ajuda a exibir de forma concreta o que muitas vezes fica superficial devido a didática usada caso seja ultrapassada. Consequentemente, os estudantes passam a compreender problemas mais complexos graças a esses softwares. Esse modelo de ferramenta vem se desenvolvendo com o passar do tempo. Com isso, graças a evolução do potencial computacional, simular o ambiente real, criar modelos, e testá-los tem sido cada vez mais útil para dinamizar o aprendizado (BERTOLDI e RAMOS, 1999).
4. Jogos: Jogos visam principalmente a parte recreativa do aprendizado. Um jogo educativo é uma peça chave para mudar o antigo conceito de monotonia educacional para algo mais divertido e que consiga atrair a atenção, principalmente do público infantil. Eles, além de dinamizar uma didática, também servem para promover a interação entre pessoas, competição, curiosidade, desafio e fantasia. Os jogos permitem que o aluno aprenda a negociar, persuadir, a cooperar e a respeitar a inteligência de seus amigos devido as habilidades estimuladas no decorrer dos jogos (BERTOLDI e RAMOS, 1999).

Os cursos de áreas tecnológicas também estão entre os cursos que utilizam essas novas aplicações para tornar mais claro e dinâmico o aprendizado de determinadas disciplinas. A ausência de disciplinas que desenvolvem o raciocínio lógico-computacional em outros níveis de educação é um problema particular dos cursos desta área. Com isso, o desejo de aperfeiçoar as formas de aprendizagem para que os alunos consigam maiores índices de aprovação em disciplinas, como por exemplo, programação, vem dos próprios professores (DIM e ROCHA, 2011).

A disciplina de Lógica Computacional, presente nos cursos de computação do ensino superior, é responsável por apresentar os conceitos da Lógica Proposicional, foco deste trabalho. Conforme Silva, Finger e Melo (2006, p.8) detalham, a Linguagem Proposicional é composta por vários elementos. Conjunto infinito e contável de símbolos proposicionais, o conectivo unário de negação, conectivos binários e o uso de parênteses como elementos de pontuação, são citados como os elementos que compõem o alfabeto para as regras de formação desta linguagem (SILVA; FINGER; MELO, 2006, p.8).

Silva, Finger e Melo (2006), explicam que na Lógica Proposicional as ideias do mundo real são expressas em forma de símbolos proposicionais, ou variáveis, e são relacionadas por conectivos lógicos. Isto faz com que estes fatos possam ser analisados e deduzidos utilizando as regras da Lógica Computacional. Porém, como o objetivo desta disciplina é trazer os problemas do mundo real para uma representação lógica até que seja encontrada uma conclusão, ela é dita como de difícil compreensão para muitos alunos.

Diversas aplicações foram desenvolvidas para tornar mais compreensível estas disciplinas, como o Tree Proof Generator (SCHWARZ, 2015), uma ferramenta web que possui um algoritmo para gerar tableaux semânticos para exibir deduções de predicados lógicos e suas premissas. A ferramenta foi criada por Wolfgang Schwarz, filósofo da Universidade Edinburgh no Reino Unido, ele também trabalha em outras áreas, dentre elas, Lógica (SCHWARZ, 2015). Para realizar a dedução, o usuário deve entrar com a premissa desejada. O gerador irá buscar um contra modelo para a fórmula informada e assim tentar provar por contradição a veracidade da mesma.

Outra ferramenta é o WinKE, um software desktop criado por Ulle Endriss voltado para a dedução de premissas. Utiliza cálculos KE (PITT e CUNNINGHAM,

1996). Esta ferramenta visa exibir a prova de premissas passo a passo e também pode ser utilizado como provador de teoremas . Entretanto, apesar de mostrar cada passo, ele não oferece a informação de qual regra foi utilizada (WINKE, 2015). Esse software foi designado para auxiliar o ensino introdutório de Lógica, sendo originalmente elaborado para plataforma Windows, mas adaptado também para outras, como a Linux (CAMPOS, 2015). WinKE também é utilizado para disciplinas como inteligência artificial, filosofia e resoluções automatizadas em universidades de todo o mundo (WINKE, 2015).

Já para a criação e auxílio com tabelas verdade, foi desenvolvido o Truth Table Constructor, criado por Brian Boroski. Esse construtor de tabelas verdade recebe como entrada um predicado lógico e exibe como saída os valores da tabela verdade correspondente às suas premissas. É uma ferramenta desktop programada utilizando a linguagem Java (ORACLE, 2015). O Truth Table Constructor é de extrema utilidade principalmente para os alunos que estão apenas começando a se familiarizar com a linguagem Lógica (BOROWSK, 2015).

Outro exemplo é a ferramenta APIN (Agência Planetária de Inteligência), que foi elaborada com o intuito de estimular o raciocínio lógico e a capacidade de solução de problemas no Ensino Médio. O fato dessas disciplinas computacionais não serem lecionadas nos níveis mais básicos de educação fez com que aplicações como APIN fossem desenvolvidas para que os estudantes começassem a se familiarizar com as mesmas (DIM e ROCHA, 2011). APIN contém 3 módulos distintos. O primeiro é chamado Academia Apin onde tutoriais de lógica e programação são disponibilizados. O segundo é conhecido como Missões Apin onde todo o conhecimento adquirido na academia pode ser aplicado. Finalmente, Operações de Treinamento de Inteligência é o terceiro modo. Neste, a lógica é estimulada através de competições entre os alunos (DIM e ROCHA, 2011).

O Inlogic é outro exemplo destes softwares. Segundo Zeni (2015), desenvolvido entre 2001 e 2003, o Inlogic pode ser visto como uma calculadora lógica que dispõe de diversos recursos, dentre eles, gráficos e tabelas É voltado para alunos que estudam lógica formal. O software oferece para estes alunos a possibilidade de interagir com Cálculo de Proposições, Tabela Verdade, Prova Automática, Formas Normais e Transformações de Equivalência. “O software Inlogic

visa facilitar o trabalho, tanto do professor, que muitas vezes não conta com outros recursos além do quadro negro; quanto do aluno, que agora dispõe de um instrumento que permite a ele experimentar conceitos e métodos, oferecendo novas perspectivas para a aprendizagem e aumentando o interesse pela disciplina ” (ZENI, 2015).

Os softwares citados foram elaborados em diferentes linguagens de programação. A linguagem para o desenvolvimento deve ser a mais adequada ao tipo de plataforma na qual a ferramenta será desenvolvida. Isso deve-se ao fato de que muitas linguagens, apesar de eficientes, não possuem uma interface adequada para o usuário, ou não possuem recursos para determinadas plataformas, entre outros problemas. Em seu livro, Mendes (2009, p.23) defende que devido à uma integração de muitas funcionalidades, Java recebe destaque dentre outras linguagens conhecidas. Java, usando os paradigmas da orientação à objetos, não só oferece suporte para aplicações desktop, mas também possibilita a migração para o desenvolvimento web. O fato dela não ser limitada a um sistema operacional específico também foi ponto forte para ser usado como característica desta linguagem (MENDES, 2009).

Todas as ferramentas expostas anteriormente possuem a função de auxílio no ensino e na aprendizagem. Porém, apesar de algumas terem uma ideia de criação semelhante a ferramenta proposta por este trabalho, elas não possuem algumas funcionalidades. A WinKE, por exemplo, é uma aplicação desktop que gera uma árvore exibindo os passos da dedução, mas não exibe qual regra foi utilizada em cada um deles. The Proof Generator é outro exemplo, porém web, que também exibe os passos das deduções, mas não fornece qual regra utilizou-se para tal conclusão. Isto faz com que, apesar de existir muitas ferramentas auxiliares, a ferramenta, JDedutor, desenvolvida neste trabalho apresente como diferencial a capacidade de exibir os as premissas informadas, as regras encontradas e a possível conclusão deduzida.

Os conceitos utilizados para elaboração de provadores de teorema se mostraram importantes para a concretização deste trabalho. Silva, Finger e Melo (2006) direcionam o desenvolvimento de provadores de teoremas, expondo estratégias computacionais, como direção de análise das regras e a estrutura de dados utilizada, podem influenciar na eficiência da ferramenta. Outra parte

importante é a escolha das estratégias de seleção de regras. As estratégias de busca em árvore, como busca em profundidade e busca em largura, são determinantes para o desempenho, e também podem facilitar a adição de mais regras a serem analisadas (SILVA; FINGER; MELO, 2006, p.68).

A dedução lógica pode ser definida, segundo Abe (2002), pelo processo de análise lógica de premissas verdadeiras até que uma conclusão, também verdadeira, seja obtida. Este processo, também chamado de inferência ou raciocínio, deve obedecer às chamadas regras de inferência Lógica (ABE,2002). Como exemplos de regras de inferência, pode-se citar as Regras Modus Ponens, Modus Tollens, Dupla Negação, e Simplificação.

Como definido por Abe (2002), Modus Ponens é formada por uma premissa contendo uma condicional, com seu antecedente e consequente assumindo o valor de verdadeiro. De acordo com a Figura 2, a premissa contendo a condicional é exemplificada por “A → B”. Ao encontrar, durante a análise das outras premissas, uma premissa igual ao antecedente da condicional, representada por A na Figura 2, conclui-se o consequente da condicional, representado por B na Figura 2. A Figura 2 ilustra uma dedução utilizando a regra Modus Ponens.

$$\frac{A \rightarrow B, A}{B}$$

Figura 2 - Modus Ponens

Também segundo Abe (2002), a regra Modus Tollens constitui-se basicamente de duas premissas. A primeira, exemplificada por “A → B” , deve ser uma condicional onde o precedente e o consequente devem representar verdadeiro. A segunda premissa, exemplificada por “¬B“, deverá ser a negação do consequente. Como conclusão determinada por essa regra, obtém-se a negação do precedente do condicional, expressado por (¬A). Observa-se na Figura 3 uma demonstração da regra Modus Tollens.

$$\frac{A \rightarrow B, \neg \mathrm{B}}{\neg \mathrm{A}}$$

Figura 3 – Modus Tollens

A regra da Dupla Negação é definida por uma premissa sendo negada duas vezes, expressada por ¬ (¬A), e uma conclusão, que neste caso, será a afirmação da premissa A (Abe, 2002). Segue na Figura 4 um exemplo de aplicação da regra da Dupla Negação.

$$\frac{\neg(\neg A)}{A}$$

Figura 4 - Dupla Negação

Carmelo e Andrade (2006) explicam que a regra da Simplificação é composta por premissas conectadas pelo operador unário ^ (lê-se: “e”). Em outras palavras, quando encontram-se em sua forma normal conjuntiva e estas premissas assumem valores verdadeiros, todas, em particular, assumem também o valor de verdade. As Figuras 5 e 6 exibem as possíveis conclusões provenientes do exemplo fornecido. Tanto a premissa A quanto a premissa B podem receber valores positivos.

$$\frac{A \wedge B}{A}$$

Figura 5 - Simplificação 1

$$\frac{A \wedge B}{B}$$

Figura 6 - Simplificação 2

Para que uma ferramenta tenha a capacidade de realizar deduções lógicas, um compilador para a linguagem lógica deve ser implementado. A criação de compiladores recebe a ajuda de bibliotecas encarregadas de realizarem determinadas análises que são consideradas partes de um compilador. A biblioteca JFlex faz parte de um grupo de bibliotecas criadas para o reconhecimento léxico, e sintático também conhecido como “*parser*˜. Esta foi desenvolvida para realizar o escaneamento léxico de um código. JFlex é, então, responsável por receber como entrada, expressões regulares que determinarão a criação de autômatos. Por sua vez, estes autômatos serão utilizados para realizar o reconhecimento dos símbolos (GESSER, 2003) característicos desta linguagem.

Existem outras bibliotecas que possuem a função semelhante à JFlex. Como outro exemplo deste tipo de biblioteca, pode-se citar a Java CUP (*Constructor of Useful Parsers*) (HUDSON, 2016). Apesar de não fazer uso da Java CUP neste

trabalho, ambas são desenvolvidas em Java e, quando utilizadas de forma integrada, produzem uma análise léxica, com a JFlex, e a sintática, com a Java CUP (HUDSON, 2016).

Visando a criação de um objeto de aprendizado mais completo, viu-se de grande utilidade a criação de um software para que alunos, mais precisamente os dos cursos de Ciência da Computação e Sistemas de Informação do CCENS-UFES, pudessem utilizar como um auxílio na evolução de seu aprendizado. Começou, então, um planejamento do que seria necessário, a escolha da linguagem adequada, e a estipulação das regras que seriam utilizadas para que o projeto fosse posto em prática.

# 3. Metodologia

Após pesquisar e analisar ferramentas que possuem a mesma finalidade que a proposta por este trabalho, escolheu-se uma plataforma e uma linguagem adequada para que a ferramenta fosse desenvolvida. Java, como define MENDES (2009), é uma linguagem simples, robusta, multithread, interpretada, neutra de arquitetura, e portável. Estas foram características chaves para que Java fosse escolhida como linguagem para o desenvolvimento da aplicação desenvolvida neste trabalho. Uma vez que a proposta inicial era a criação de um software desktop, é importante que haja uma flexibilidade em relação a possibilidade de que esta ferramenta também migre para o ambiente móvel.

A ferramenta JDedutor foi criada para ajudar os alunos a aprender e entender o processo de dedução da lógica computacional. A interface foi elaborada para oferecer um ambiente interativo, exibindo as regras utilizadas, as interações realizadas por suas aplicações e a possível conclusão após suas aplicações. Exibir cada etapa da dedução é importante para que o aluno não só tenha em mãos o resultado, mas para que também possa entender as etapas de sua resolução

O escopo inicial são os alunos cursando as disciplinas Lógica Computacional I e II do Campus da cidade de Alegre da Universidade Federal do Espírito Santo. Futuramente, o objetivo é que alunos de diferentes períodos, e outras universidades possam ter a ferramenta como parte de suas estratégias de aprendizado.

A princípio, três das regras foram escolhidas para o desenvolvimento da ferramenta: Modus Ponens, Modus Tollens e Regra da Dupla Negação. Estas, são classificadas como regras de inferência lógica e utilizadas para o desenvolvimento e dedução de predicados (ABE, 2002). A regra Modus Ponens e Modus Tollens foram selecionadas pois seus conceitos se entrelaçam com os utilizados para a construção das estruturas “se“ e “então“ presentes nos códigos para desenvolvimento de programas. Com a aparição de negações e duplas negações provenientes das conclusões da regra Modus Tollens, observou-se que a regra de Dupla Negação seria útil para o funcionamento do Dedutor.

Posteriormente, a regra de inferência conhecida como Simplificação foi adicionada para que o arquivo selecionado pelo usuário com as proposições pudesse conter menos premissas em uma mesma linha, facilitando a leitura e

dedução do compilador. A aplicação da regra de Simplificação seria exigida caso uma premissa informada fosse composta por múltiplas conjunções.

Para que a leitura da Linguagem Lógica fosse realizada, foi necessária a criação de um compilador para a mesma. O compilador foi elaborado para realizar a análise léxica, sintática e semântica da linguagem (GESSER, 2003) para que esta pudesse ser utilizada nas deduções. Para a análise léxica, a ferramenta fez uso da biblioteca JFlex (JFlex, 2015), que realizou o reconhecimento dos símbolos. Este reconhecimento foi importante para que fosse possível aplicar uma análise sintática do código fornecido pelo usuário para o programa.

A biblioteca JFlex foi, então, adicionada ao projeto criado utilizando a IDE Netbeans. A partir disto, o projeto estava apto a realizar a leitura das expressões regulares que definiriam os símbolos escolhidos para a linguagem lógica utilizada. Os métodos responsáveis para realizar a chamada da JFlex foram desenvolvidos na classe principal do projeto criado. Desta forma, ao compilar o projeto, a biblioteca era automaticamente acionada, fazendo das expressões regulares e gerando os autômatos responsáveis pelo reconhecimento da linguagem escolhida. Cada símbolo reconhecido era adicionado à um vetor, que seria utilizado pelas outras etapas de construção do projeto.

Com os símbolos reconhecidos, o processo de análise sintática foi desenvolvido para que ocorresse a verificação do código informado de acordo com a gramática da linguagem lógica. Gesser (2003) define que uma gramática de uma linguagem é composta pelos símbolos terminais e não terminais (responsáveis por gerarem as produções gramaticais) que caracterizam uma linguagem. Para o reconhecimento de cada produção, métodos foram criados para realizar a análise sintática. As produções definidas por esta gramática foram divididas em 3 grupos: Sentenças complexas, sentenças atômicas e símbolos.

Os símbolos são a representação, em forma literária, de algum fato que deseja ser analisado. Por exemplo, a frase “Juliana possui um cachorro“, pode ser representada pela letra P. De acordo com a gramática definida para este compilador, P é considerada um símbolo terminal. Este literal pode assumir o valor verdadeiro ou falso. As sentenças atômicas podem assumir valores, sem representação literal, verdadeiros ou falsos, ou, ainda, reproduzirem um novo símbolo. E por fim, as sentenças complexas são constituídas por sentenças simples ou complexas se

relacionando pelos conectivos lógicos de negação, conjunção, disjunção inclusiva, disjunção exclusiva, condicional, bi condicional. Estas relações são demonstradas na Figura 7.

- **Sentença** → SentençaAtômica | SentençaComplexa
- **SentençaAtômica** → Verdadeiro | Falso | Símbolo
- **Símbolo** → P | Q | R | ...
- **SentençaComplexa** → ¬Sentença
  | (Sentença ∧ Sentença)
  | (Sentença ∨ Sentença)
  | (Sentença ⇒ Sentença)
  | (Sentença ⇔ Sentença)

Figura 7 - Demonstração das produções para leitura da gramática

Seguindo suas produções, um compilador consegue ler tal gramática e reconhecer se o código informado é compatível ou não com a linguagem que lhe foi designada (GESSER, 2003). A gramática para dedução lógica desta ferramenta foi desenvolvida também em Java utilizando como base os conceitos lógicos característicos de cada regra.

Como último passo de análises, foi realizado o desenvolvimento da estrutura responsável pela análise semântica dos códigos compilados. Essa é responsável por validar o sentido da gramática e extrair todas as informações para que o código seja interpretado corretamente (GESSER, 2003). A etapa da análise semântica foi a responsável por conter cada regra de inferência proposta para a ferramenta. Classes foram criadas para que o reconhecimento de cada regras fosse realizado de forma separada. No caso, as regras Modus Tollens, Modus Ponens, e Dupla Negação foram analisadas em classes diferentes. Cada classe teve a responsabilidade de analisar, reconhecer, e aplicar no vetor de símbolos as alterações referentes à definição de cada regra e suas respectivas conclusões.

Durante o desenvolvimento da ferramenta, observou-se que a possibilidade do usuário informar premissas consideradas inadequadas para o reconhecimento, faria com que a precisão dos resultados fosse prejudicada. Premissas inadequadas para este compilador são as que se enquadram na aplicação da regra da Simplificação, ou que possuem parênteses desnecessários. Parênteses estes identificados como símbolos na análise léxica. Com o intuito de diminuir a quantidade de premissas por linha, a regra de simplificação foi adicionada. Seu

desenvolvimento foi realizado para que, ao compilar o código, premissas que se conectassem por conjunção fossem separadas em linhas diferentes. Esta etapa aconteceria antes da aplicação das 3 regras de inferência inicialmente determinadas para esta ferramenta de dedução. Juntamente com a regra da simplificação, um método com a finalidade de eliminar parênteses desnecessários também foi aplicado. Este método atua eliminando parênteses desnecessários que podem causar divergências nos resultados após a aplicação de determinas regras.

Como exemplo dos problemas causados por parênteses desnecessários, pode-se citar o fato de uma premissa poder ser representada de diferentes formas. Uma premissa B deve ser considerada igual à uma premissa representada por ((B)). Esta regra de eliminação de parênteses foi criada para tentar eliminar problemas desta natureza, onde os parênteses de premissas como a ((B)) seriam ignorados e ela seria comparada como apenas um símbolo B.

Para utilizar o JDedutor, o usuário deve criar um arquivo texto contendo as premissas desejadas. Após isso, ele deve executar o aplicativo, selecionar o arquivo criado e, então, executar a dedução. A Figura 8 demonstra a interface da ferramenta ao ser executada.

Quando o arquivo de texto é selecionado, seu código é apresentado no campo Expressão (Figura 9). Quando clica-se no botão Executar, as regras deduzidas pelo aplicativo são apresentadas no campo Regras (Figura 10) , e a possível conclusão gerada pela dedução é apresentada no campo Conclusão (Figura 10).

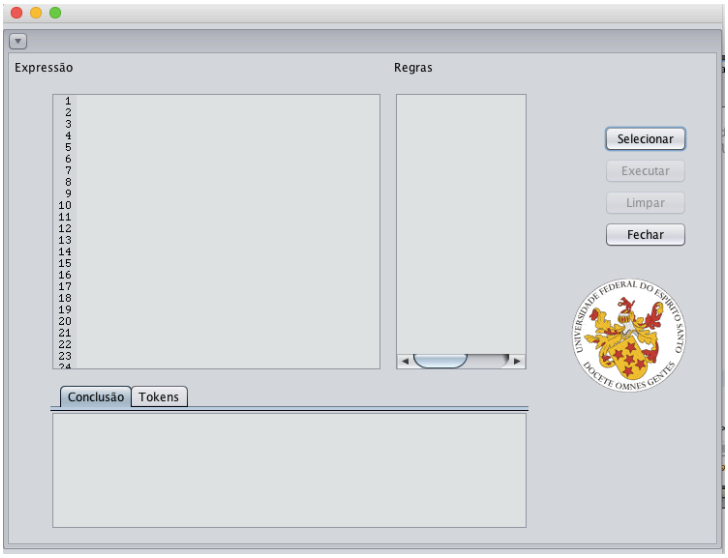


Figura 8 - Tela inicial do JDedutor

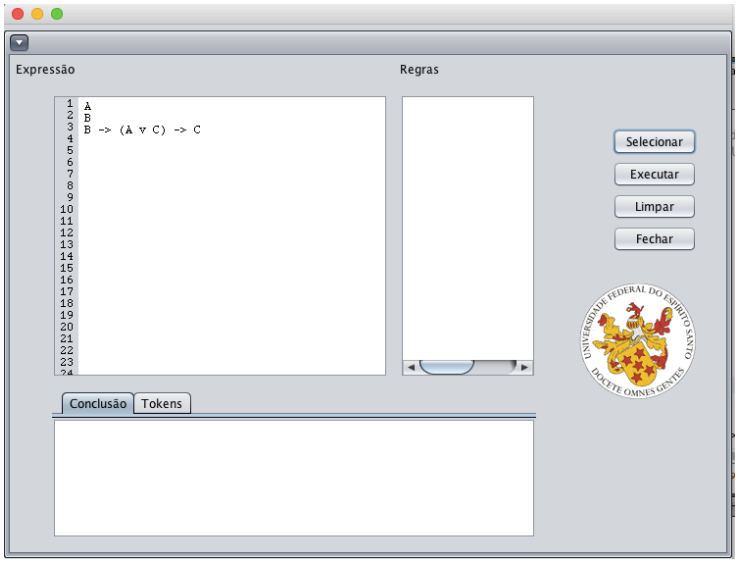


Figura 9 - Tela após arquivo selecionado

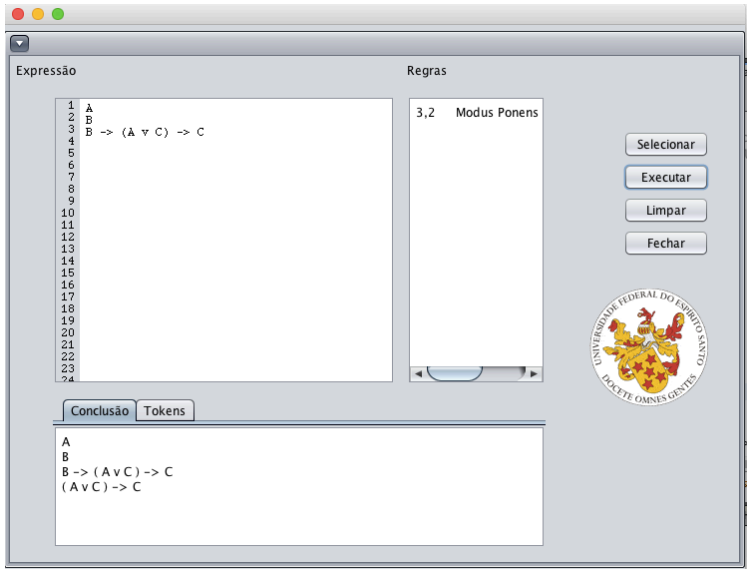


Figura 10 - Tela após executar o arquivo

A ferramenta possui as seguintes limitações para que o arquivo seja reconhecido:

1. Deve-se substituir → por -> (Sinal de menos seguido do sinal de maior);
2. Deve-se substituir o símbolo ∧ por ^ (Acento Circunflexo);
3. Deve-se substituir o símbolo v por v (Letra “v“ minúscula);
4. Use apenas letras maiúsculas.

Cada premissa adicionada também precisa ser elaborada de acordo com os passos abaixo:

1. Deve-se posicionar cada premissa em uma linha. Em caso de mais de uma regra numa mesma linha, separadas por vírgulas, o programa irá acusar um erro, pois a vírgula não é reconhecida neste compilador.
2. Deve-se colocar premissas simples em cada linha. Premissas simples colaboram para que a dedução seja mais precisa.
3. O cursor deverá estar posicionado depois do último caractere digitado no código. Caso haja espaçamentos no ultimo caractere digitado no código, erros serão reportados.
4. Deve-se evitar: Parênteses desnecessários; Premissas compostas ou que possuam parênteses sendo colocadas em uma mesma linha. Por exemplo: A -> (B ^ C) v B v C -> A. Os parênteses pode causar erros caso haja a aplicação de Modus Ponens ou Tollens no decorrer da execução.

Ao final do desenvolvimento, o software foi disponibilizado para os alunos dos cursos de Ciência da Computação e Sistemas de Informação que cursam a disciplina Lógica Computacional I e Lógica Computacional II no CCENS da UFES, em Alegre. Os alunos foram instruídos a criarem seus próprios arquivos com regras, executa-lo e observar como o JDedutor reagiria com as premissas fornecidas.

Os testes foram realizados em laboratórios disponibilizados no campus da universidade. Após testarem, os alunos responderam um questionário expondo suas críticas positivas e negativas em relação a ferramenta apresentada. Esta etapa foi importante para apresentar para os alunos uma alternativa que possa auxiliar os estudos e as metodologias de ensino para estas disciplinas. Adicionalmente, as reações ao testarem e as respostas fornecidas serão importantes para o aprimoramento da ferramenta de dedução proposta.

# 4. Análise dos Resultados

Após a aplicação dos testes, todas as respostas fornecidas pelos alunos por meio do questionário aplicado foram analisadas. Como se tratava de 2 turmas selecionadas para a aplicação da ferramenta, as respostas também estão divididas em 2 etapas: uma contendo as informações da turma de Lógica Computacional I e a outra contendo as informações da turma de Lógica Computacional II. Esta divisão ocorreu para que se pudesse observar as diferenças entre opiniões de alunos que já cursaram Lógica Computacional I, e superaram as dificuldades com este primeiro contato sendo aprovados para Lógica Computacional II, e os alunos que estão atualmente passando pelo processo de aprendizado de Lógica Computacional I.

Para melhor visualização dos resultados obtidos, serão apresentados os gráficos das duas turmas participantes separados pelas perguntas realizadas. Somente os alunos matriculados nas respectivas disciplinas participaram deste teste. A turma de Lógica Computacional I era composta por 28 alunos, do sexo masculino e feminino, e de diferentes períodos. A turma de Lógica Computacional II, 14 alunos também de diferentes períodos, e do sexo masculino e feminino.

A primeira pergunta realizada foi para que o pouco contato com a disciplina e suas aplicações em níveis de educação inferiores ao ensino superior, pudesse ser comprovado. Observa-se nas Figuras 11 e 12, respectivamente, que 64.3% dos alunos de Lógica Computacional I não conheciam a disciplina antes da faculdade, e 78.6% dos alunos de Lógica Computacional II também não tiveram contato. Em ambas as situações, mais da metade do número de alunos, então, tiveram seu primeiro contato apenas após o ingresso na faculdade. A partir destes números, confirma-se ausência de um primeiro contato com disciplinas que envolvam resolução de problemas utilizando Lógica, antes do nível superior de educação.

Essa ausência é um fator que dificulta a troca de informações, principalmente, entre os próprios alunos, fazendo com que toda a fonte de conhecimento seja o professor. Isto também limita o trabalho do professor. Em turmas em que há alunos que não possuem conhecimento algum sobre os conceitos de tal disciplina, o foco do ensino acaba direcionado para conteúdos mais básicos, tomando o tempo que poderia ser utilizado para a explicação de conteúdos com uma complexidade maior. O trabalho do professor como ponte de conhecimento acaba se

tornando mais difícil do que quando alguns alunos já possuem certa noção das disciplinas, funcionando como colaboradores para as metodologias utilizadas pelo educador.

Figura 11 - Porcentagem para resposta 1 de Lógica Computacional I

Figura 12 - Porcentagem para resposta 1 de Lógica Computacional II

A segunda pergunta abrangeu o grau de dificuldade que cada aluno sentia com a disciplina. Conforme se observa na Figura 13, Lógica Computacional é uma disciplina que possui um nível muito alto de dificuldade, dentre as disciplinas lecionadas no curso de Ciência da Computação e Sistemas de Informação. Mais da metade dos alunos de Lógica I, como observado na Figura 13, sentem dificuldades na disciplina. De acordo com a Figura 13, 10,7% dos alunos sentem um nível muito alto de dificuldade, 25% sentem um nível alto e 42,9% classificam sua dificuldade como média. Porém, na disciplina de Lógica II, os alunos mostraram enfrentar menos dificuldades, já que não houve respostas referentes a níveis altos ou muito altos. Esta informação pode ser observada na Figura 14. Cerca de 57,1% disse apresentar um nível médio de dificuldade, e o restante de alunos, igualmente divididos entre os níveis baixo, muito baixo, e nenhuma dificuldade.

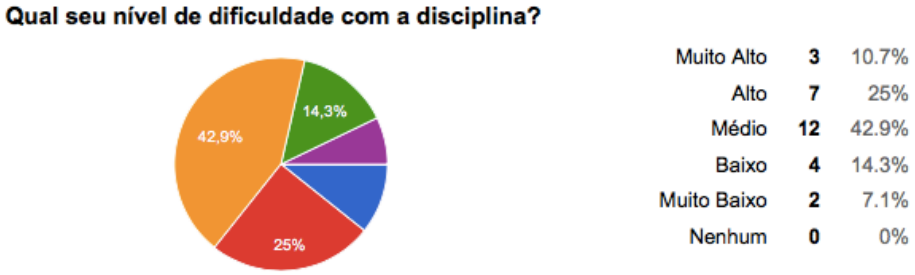


Figura 13 - Porcentagem para resposta 2 de Lógica Computacional I

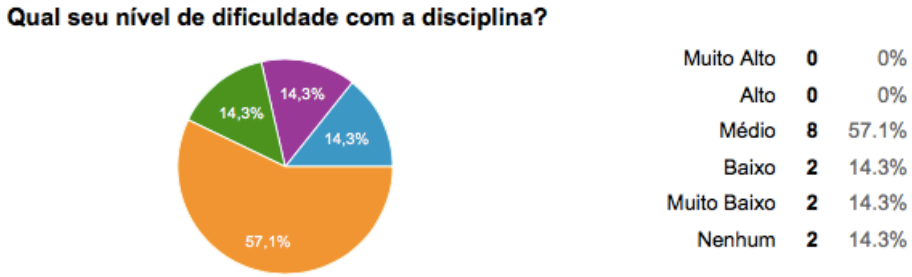


Figura 14 - Porcentagem para resposta 2 de Lógica Computacional II

Tomando como base os resultados expostos pelas Figuras 13 e 14, pode-se confirmar que a ideia da elaboração de uma ferramenta com intuito de diminuir os altos níveis de dificuldade com a disciplina de Lógica Computacional é válida.

Ao serem questionados, na terceira pergunta, sobre o nível de dificuldade em fazer deduções lógicas no inicio do aprendizado, os alunos da turma de Lógica Computacional I preencheram as opções referentes a altos níveis de dificuldade. Observa-se na Figura 15 que cerca de 82,2% da turma posicionou-se entre os níveis médio, alto e muito alto de dificuldade. A opção de nível alto foi a mais escolhida representando 39,3% dos alunos. Essa grande porcentagem ajuda a confirmar o fato de que a dedução é um mecanismo da resolução de problemas que ainda é visto como um empecilho para o aprendizado desta disciplina. Adicionalmente, 17,9% da turma afirmaram ter sentido um baixo nível de dificuldade para realizar as atividades, e as opções de nível muito baixo ou nenhum não receberam marcações.

Figura 15 - Porcentagem para resposta 3 de Lógica Computacional I

Na turma de Lógica Computacional II, como observado na Figura 16, as opções que representavam os altos níveis de dificuldade para fazer deduções lógicas no inicio do aprendizado, também receberam a maioria das marcações. Do total de 14 alunos, 71,4% se enquadraram nas respostas dos níveis muito alto, alto e médio. Como opção mais escolhida, o nível médio recebeu 35,7% das marcações. Apenas 28,6% afirmaram ter enfrentado um baixo nível de dificuldade, enquanto as opções de nível muito baixo e nenhuma dificuldade não receberam marcações. Perante aos resultados, observa-se como que deduções não foram uma barreira no início do aprendizado apenas para a turma atual de Lógica I, mas também para alunos que a realizaram em outro momento acadêmico.

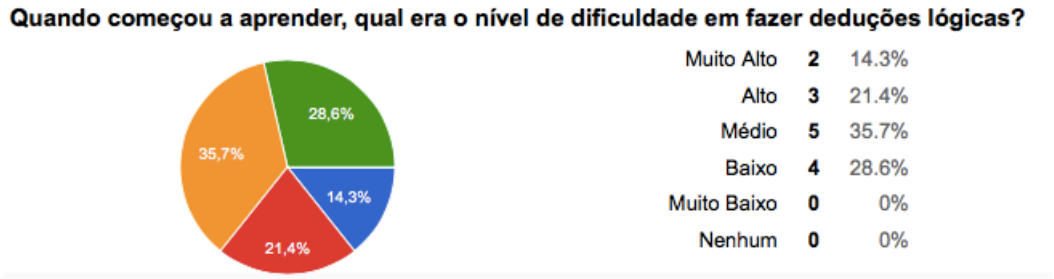


Figura 16 - Porcentagem para resposta 3 de Lógica Computacional II

Os resultados exibidos nas Figuras 15 e 16 justificam a ideia de criação de uma ferramenta que pudesse fazer com que o contato com deduções lógicas possa ser algo mais dinâmico. O objetivo deste trabalho é poder fazer com que essa ferramenta possa ajudar a diminuir os níveis de dificuldade com, pelo menos, esta disciplina, independente de conhecerem ou não seus conceitos previamente.

A quarta questão tinha o intuito de identificar a dificuldade atual que os alunos ainda sentem com a disciplina de Lógica. A Figura 17 vem para expor os resultados dos alunos de Lógica Computacional I, que apresentaram uma pequena melhora em

relação aos níveis de dificuldades. Nesta questão, o nível médio, com 35,7% foi que apresentou maior número de respostas. Esta mudança pode ser deduzida pelo fato de na pergunta anterior, onde questionava-se o nível de dificuldade no inicio do aprendizado, o nível alto foi a opção que apresentou maior número de alunos. Desta vez, 75% dos 28 alunos entrevistados responderam as opções mais altas de dificuldades, dividas entre nível médio, alto, e muito alto. Porém, um numero maior de alunos marcaram as opções de baixo e muito baixo nível de dificuldade. Agora, 7,1% dos alunos alegaram enfrentar um nível muito baixo de dificuldade.

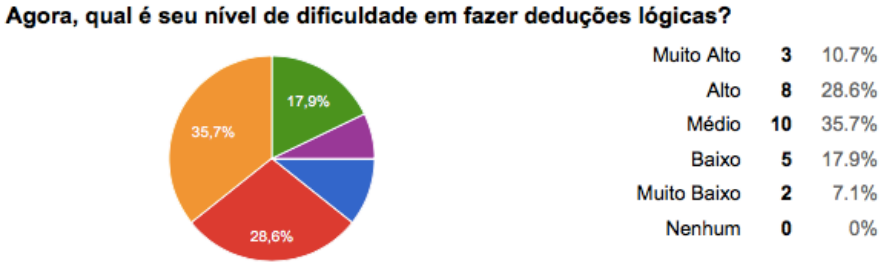


Figura 17 - Porcentagem para resposta 4 de Lógica Computacional I

A Figura 18 exibe os resultados, também da quarta questão, da turma de Lógica Computacional II quando questionados sobre o nível atual de dificuldade com deduções. Houve uma diminuição de respostas relacionadas a níveis altos de dificuldade e um aumento nas respostas para os níveis baixos. O nível médio foi o mais escolhido, apresentando 50% das respostas dos 14 alunos que realizaram os testes. As opções de níveis muito alto e alto de dificuldades não receberam marcações, enquanto, a opção de nível baixo foi escolhida por 42,9%, dos alunos. 7,1%, inclusive, afirmaram não ter mais dificuldade com deduções.

Com isso, observa-se que com o passar do tempo, os alunos foram aprendendo a assimilar os métodos oferecidos pela dedução. Como a ferramenta criada não exige que o usuário fique escrevendo e reescrevendo suas premissas e resultados a cada interação, e cada regra reconhecida, mais testes podem ser postos em prática por parte dos alunos. Os alunos então, ganham uma ferramenta que irá tornar mais fácil sua prática, dando oportunidade para que um contato maior com deduções lógicas ocorra.

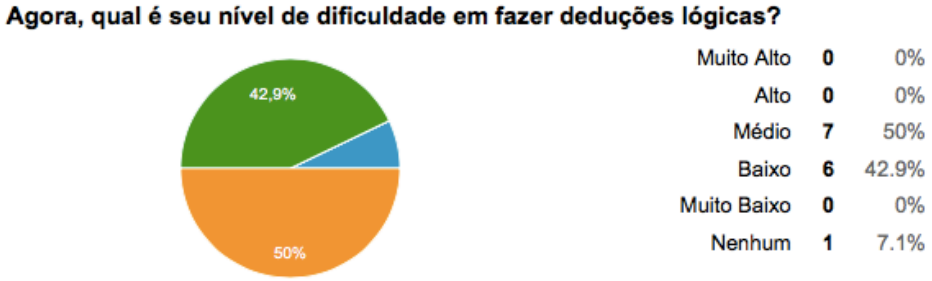


Figura 18 - Porcentagem para resposta 4 de Lógica Computacional II

Com as Figuras 15,16,17, e 18, pode-se então presumir que, a prática pode fazer com que as dificuldades enfrentadas no ensino de Lógica Computacional diminuam com o tempo. Quanto maior o contato com a disciplina, menos abstrato seus conceitos acabam se tornando, o que facilita o entendimento de suas aplicações. Desta forma, cria-se as chances de que um ambiente mais atrativo de aprendizado seja oferecido.

A quinta pergunta do questionário era relacionada a utilidade de uma ferramenta que auxiliasse nas deduções. Ambas as turmas tiveram respostas positivas. Na turma de Lógica I, apenas 1 aluno não achou útil o uso de uma ferramenta auxiliar. Na turma de Lógica II, todos os alunos presentes no teste consideraram útil o uso da ferramenta. Com os resultados desta questão, prova-se que o JDedutor será algo indispensável para os momentos de estudos sobre lógica computacional. Essas opiniões estão expressas nas Figuras 19 e 20, mostrando os resultados de Logica Computacional I e Lógica Computacional II, respectivamente. Estes resultados reafirmam que a idéia de criar a ferramenta de dedução é útil, principalmente, para alunos da disciplina de Lógica.

Figura 19 - Porcentagem para resposta 5 de Lógica Computacional I

Figura 20 - Porcentagem para resposta 5 de Lógica Computacional II

A sexta pergunta era de abrangência mais geral sobre a ferramenta. O intuito era saber se o software despertou uma boa impressão, no geral, após sua apresentação. Como observado na Figura 21, a ferramenta teve uma avaliação muito positiva, onde, 50% da turma de Lógica Computacional I o classificou como ótimo e 39.3% como bom. Num total de 18 alunos, 2 opinaram como sendo ruim, e apenas 1 optou como péssima ferramenta.

Figura 21 - Porcentagem para resposta 6 de Lógica Computacional I

As opiniões sobre a ferramenta na turma de Lógica Computacional II, expressas pela Figura 22, também foram muito positivas. Mais da metade da turma de 14 alunos deram opiniões entre bom (49,9%) e ótimo (28,6%). A repercussão negativa também não foi grande, onde 2 alunos disseram ser ruim, e 2 avaliaram como péssimo e não reportaram o motivo destas marcações. Todas as críticas serão importantes para que aperfeiçoamentos sejam planejados e desenvolvidos em busca de um maior número de opiniões positivas. Com bases nessas opiniões, conclui-se que há aspectos a serem melhorados com relação a interface.

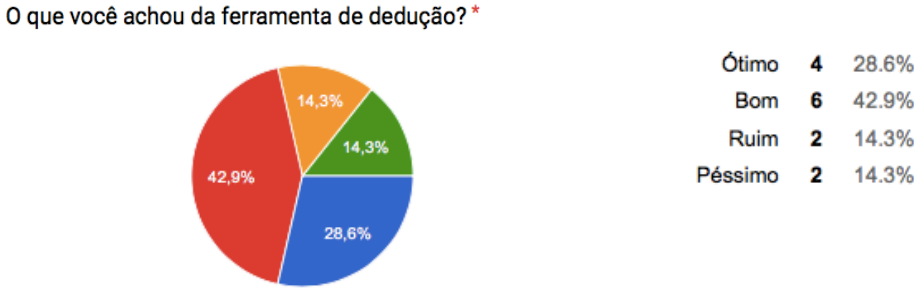


Figura 22 - Porcentagem para resposta 6 de Lógica Computacional II

A sétima pergunta foi realizada com objetivo de avaliar a interface da ferramenta. Uma porcentagem de 75% dos 28 alunos da turma de Lógica Computacional I avaliaram como boa a aparência, e 14.3% como ótima. Apenas 1 aluno avaliou como péssima aparência, e 2 como ruim. As avaliações negativas foram baixas, porém, necessárias para u futuro melhoramento da interface. Estes dados podem ser observados na Figura 23.

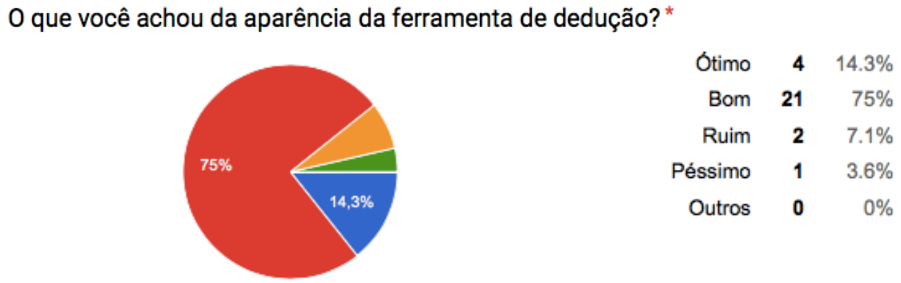


Figura 23 - Porcentagem para resposta 7 de Lógica Computacional I

Na Figura 24, observa-se que a grande maioria dos 14 alunos matriculados em Lógica Computacional II também fizeram avaliações positivas no quesito aparência. A opção ótima, 28,6% das respostas recebeu e a opção boa 57,1%. Apenas 2 alunos avaliaram como ruim. Todos esses dados também serão utilizados como base para possíveis melhorias na ferramenta, uma vez que um dos objetivos deste trabalho é que a mesma possua uma interface fácil e que estimule sua utilização.

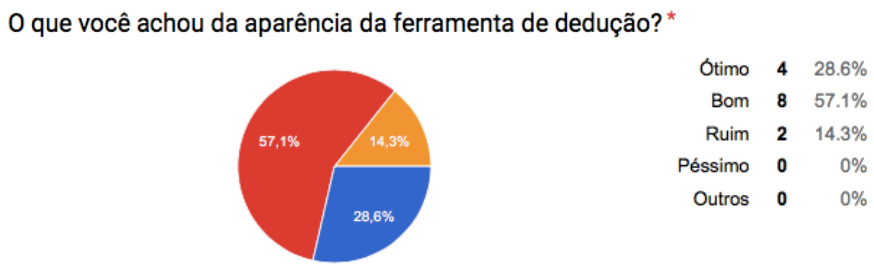


Figura 24 - Porcentagem para resposta 7 de Lógica Computacional II

Um dos objetivos para a elaboração deste trabalho foi trazer algo que pudesse tornar a metodologia de ensino da disciplina mais estimulante. Por isso, a oitava pergunta foi elaborada para que esta utilidade pudesse ser avaliada. Para a turma de Lógica Computacional I, a ferramenta se mostrou mais atrativa, uma vez que 50% da turma a avaliou como sendo de utilidade média. Ou seja, nada extremamente estimulante, mas que ajuda a passar uma imagem mais positiva das deduções realizadas na disciplinas. De um total de 28 alunos, 25% afirmou ser muito estimulante ter esta ferramenta de dedução para os estudos. As opções pouco estimulante e nada estimulante receberam 14,3% e 10,7% das avaliações, mostrando que alguns alunos não seriam motivados, mesmo com uma ferramenta de ajuda.

Figura 25 - Porcentagem para resposta 8 de Lógica Computacional I

A turma de Lógica Computacional II não se mostrou muito estimulada pela ferramenta. Analisando os 14 alunos da turma de Lógica II, 42,9% classificaram como uma ferramenta de médio potencial estimulante, enquanto, grande parte avaliou como pouco estimulante, sendo representado por 28,6% dos alunos, e ruim, 21,4% dos mesmos. Estas informações, expressadas pelas Figura 25 e 26, poderão também servir como base para futuros aprimoramentos, uma vez que apesar de se mostrar, útil, a ferramenta não trouxe um grande estímulo para o aprendizado de Lógica Computacional

Figura 26 - Porcentagem para resposta 8 de Lógica Computacional II

A nona questão foi elaborada para que se pudesse avaliar, de forma direta, a real possibilidade de utilização da ferramenta. Uma total 26 dos 28 alunos da turma de Lógica Computacional I disseram que gostariam sim de usar o JDedutor para o aprendizado na disciplina, enquanto apenas 2 alunos disseram que não gostariam de utilizar. Na turma de Lógica Computacional II, 11 dos 14 alunos disseram que gostariam de utilizar a ferramenta, e, por outro lado, 3 afirmaram que não gostariam de usar. Pode-se concluir, de acordo com a nona questão, que o software proposto neste trabalho, seria algo que se tornaria popular entre os alunos, uma vez que a grande maioria das respostas se mostrou positiva em relação à sua utilização.

Para melhor observação destes resultados, as Figura 27 e 28 foram elaboradas representando as turmas de Lógica Computacional I e II, respectivamente.

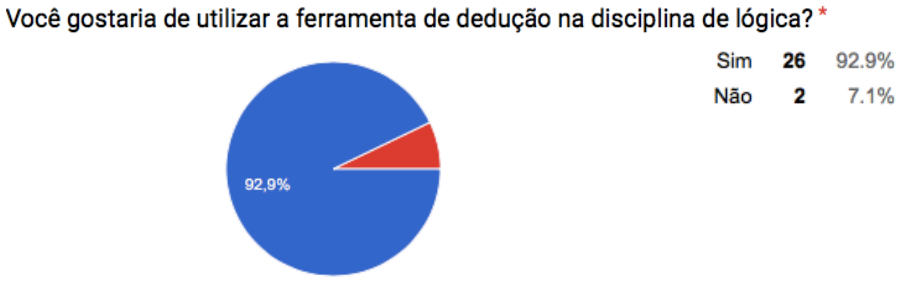


Figura 27 - Porcentagem para resposta 9 de Lógica Computacional I

Figura 28 - Porcentagem para resposta 9 de Lógica Computacional II

A décima pergunta foi elaborada para que futuras melhorias fossem estimuladas. Com isso, na turma de Lógica I, expressada pela Figura 29, 85,7% dos alunos acreditam que mais funcionalidades adicionadas ao software, poderiam diminuir as limitações apresentadas atualmente. Na turma de Lógica II, 92,9% dos alunos participantes também concordam que melhorias deverão ser realizadas para que a ferramenta se torne útil para um número maior de deduções, como observado na Figura 30. Atualmente o software possui algumas limitações, principalmente em relação ao número de regras que é capaz de analisar. De acordo com esses resultados, pode-se concluir que, há um real interesse em possíveis melhorias para que a ferramenta se torne mais completa. Quanto mais completa, maior a ajuda que a mesma pode oferecer para os alunos.

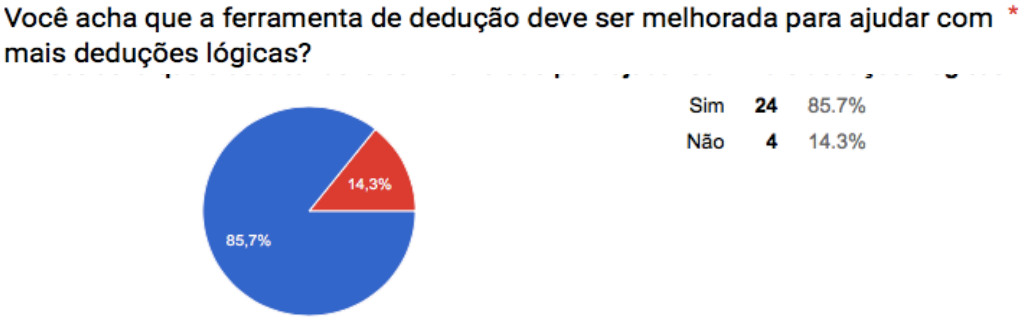


Figura 29 - Porcentagem para resposta 10 de Lógica Computacional I

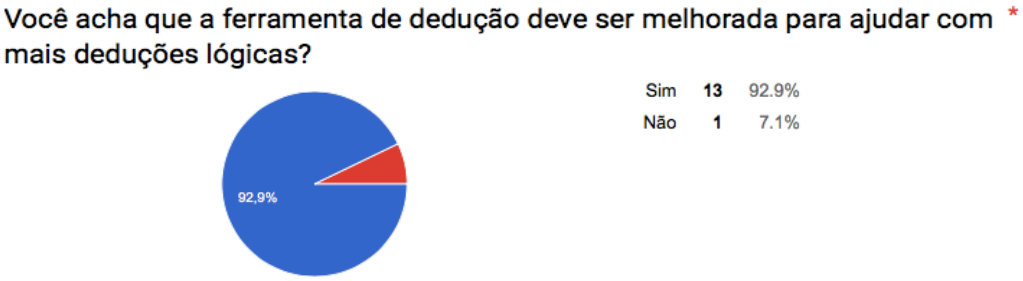


Figura 30 - Porcentagem para resposta 10 de Lógica Computacional II

Nas Figuras 31 e 32, pode-se observar os resultados das turmas de Lógica Computacional I e I, respectivamente, em relação a décima primeira pergunta. Esta, foi elaborada para que os alunos expressassem o quanto o uso da tecnologia como auxílio tem se tornado presente e se feito necessário para, não só a lógica mas também, as disciplinas em geral na faculdade. Apenas 1 aluno da turma de Lógica Computacional I e 1 de Lógica Computacional II se mostraram não simpatizantes com o uso destas ferramentas no meio acadêmico. Os demais, avaliaram de forma

positiva a presença destes softwares. Com estes resultados observa-se um interesse de parte dos alunos por ferramentas que possa fazer com que o aprendizado seja mais fácil, ou dinâmico.

Figura 31 - Porcentagem para resposta 11 de Lógica Computacional I

Figura 32 - Porcentagem para resposta 11 de Lógica Computacional II

# 5. Conclusão

Conclui-se que a ferramenta construída representa um grande promessa para facilitar o aprendizado da disciplina de Lógica Computacional. O interesse dos alunos pelo uso da ferramenta, e a boa reputação que o primeiro protótipo obteve ao ser apresentado foram refletidos em resultados positivos no decorrer do questionário. Segundo Moura et al. (2001), não cabe ao professor apenas ministrar uma aula, mas também “ ensina-los a aprender ”. A internet e a tecnologia auxiliam para que o professor possa ser uma ponte que leve ao aluno não só um conhecimento limitado, mas também, o acesso inúmeras ferramentas que auxiliam esta transmissão de conhecimento (MOURA et al., 2001). A ferramenta de dedução desenvolvida neste trabalho, provou-se estar dentre as que poderá representar uma ajuda para professores e alunos em suas rotinas de estudo.

Tomando como base os resultados obtidos através do questionário aplicado após os testes realizados pelos alunos na ferramenta JDedutor, o desenvolvimento da ferramenta teve um retorno positivo para disciplina de Lógica. Foi comprovado, pelas respostas dos alunos participantes, que a maioria não teve contato com Lógica em níveis de educação anteriores ao ensino superior. Como consequência, a dificuldade de visualizar, ou entender, a real aplicação da lógica computacional se torna algo muito abstrato.

A ideia de aplicar o questionário para todos os alunos do curso, infelizmente, não pode ser atendida. Estes seriam resultados importantes, principalmente os resultados de pessoas que deixaram o curso, pra que fosse possível criar uma relação mais forte entre as desistências e as dificuldades com Lógica Computacional. Parte dos aprimoramentos propostos para trabalhos futuros está relacionado, também, a ampliação do número de perguntas do formulário e da população selecionada para testar a ferramenta. Quanto maior o número de pessoas e períodos analisados, mais fiel os resultados serão para representar a realidade, e mais completa será a base para as possíveis melhorias.

A dedução de premissas lógicas, essencial para o entendimento sobre resolução de problemas e foco deste trabalho, provou-se ser parte que eleva grau de dificuldade da disciplina de Lógica Computacional. Porém, com a análise dos resultados, uma ferramenta que seja capaz de exibir as etapas de uma dedução, as

regras aplicadas na mesma, e uma possível conclusão, é uma forte aliada para que as barreiras que dificultam o aprendizado possam diminuir.

A ferramenta de dedução ainda não poder ser classificado como uma completa para o ensino de lógica, sendo que não reconhece todas as regras de inferências, e também possui limitações, como o a disposição de parênteses nas premissas informadas. Entretanto, esta ferramenta trouxe consigo uma nova oportunidade de aprendizado para os que enfrentam grandes dificuldades com esta área computacional.

# 6. Referências

ABE, Jair Minoro. **Introdução à Lógica para a Ciência da Computação**. Arte & Ciência, 2002.

ALENCAR FILHO, Edgard de. **Iniciação à lógica matemática**. NBL Editora, 2002.

BENTO, Maria Cristina Marcelino; CAVALCANTE, Rafaela dos Santos. Tecnologias Móveis em Educação: o uso do celular na sala de aula. **Educação, Cultura e Comunicação**, v. 4, n. 7, 2013.

BERTOLDI, S.; RAMOS, E. M. F. Avaliação de Software Educacional–Impressões e Reflexões. **Florianópolis: UFSC**, 1999.

Borowski, B., Truth Table Constructor. Disponível em: <http://www.brian-borowski.com/software/truth/>. Acesso em Outubro 2015.

BUBAS, Goran; CORIC, Ana; OREHOVAČKI, Tihomir. The evaluation of the use of online community tool Ning for support of student interaction and learning. In: **CECIIS-2010**. 2010.

CAMPOS, A. O que é Linux. Disponível em:< http://br-linux.org/2008/01/faq-linux.html>. Acesso em Outubro de 2015.

CARMELO, Emerson L. Monte; ANDRADE, Doherty. Elementos de Lógica Proposicional, 2006.

DIM, Cleyton Aparecido; ROCHA, FEL da. APIN: Uma Ferramenta Para Aprendizagem de Lógicas e Estímulo do Raciocínio e da Habilidade de Resolução de Problemas em um Contexto Computacional no Ensino Médio. In: **XIX Workshop sobre Educação em Computação**. 2011.

ELLISON, Nicole; WU, Yuehua. Blogging in the classroom: A preliminary exploration of student attitudes and impact on comprehension. **Journal of Educational Multimedia and Hypermedia**, v. 17, n. 1, p. 99-122, 2008.

GESSER, Carlos Eduardo. GALS: gerador de analisadores léxicos e sintáticos. 2003. 150 f. **Trabalho de Conclusão de Curso (Bacharelado em Ciência da Computação)–Centro Tecnológico, Universidade Federal de Santa Catarina, Florianópolis. Disponível em:< http://projetos. inf. ufsc. br/arquivos_projetos/projeto_353/Gals. pdf>. Acesso em: Novembro 2015**, v. 19, 2012.

HUDSON, Scott. **LALR Parser Generator for Java**. Disponível em: http://www2.cs.tum.edu/projects/cup/. Acesso em Outubro de 2016.

JFLEX. Disponível em: <http://jflex.de>. Acessado em: 06 de Novembro de 2015.

KATZ, Richard N. **The tower and the cloud: Higher education in the age of cloud computing**. Educause, 2010.

LIU, Min et al. Web 2.0 and its use in higher education from 2007-2009: A review of literature. **International Journal on E-Learning**, v. 11, n. 2, p. 153-179, 2012.

MEDINA, Marco; FERTIG, Cristina. Algoritmos e Programação. **São Paulo, Novatec Editora**, 2005.

MENDES, Douglas Rocha. **Programação Java com ênfase em Orientação a Objetos**. Novatec Editora, 2009.

MORTARI, Cezar A. **Introdução à lógica**. SciELO-Ed. UNESP, 2001.

MOURA, Ana Maria Mielniczuk de; AZEVEDO, Ana Maria Ponzio de; MEHLECKE, Querte. As teorias de aprendizagem e os recursos da Internet auxiliando o professor na construção do conhecimento. In: **Anais do VIII do Congresso Int. de Educação a Distância**. 2001.

NEIL, Theresa. **Padrões de design para aplicativos móveis**. Novatec Editora, 2012.

Ning. **What is Ning?.** Disponível em: < http://www.ning.com/what-is-ning/ >. Acesso em Outubro de 2015.

Oracle. Software Java. Disponível em:< http://www.oracle.com/br/java/overview/index.html>. Acesso em Outubro 2015.

PITT, Jeremy; CUNNINGHAM, Jim. Theorem proving and model building with the calculus KE. **Logic Journal of IGPL**, v. 4, n. 1, p. 129-150, 1996.

RUSCHEL, Henrique; ZANOTTO, Mariana Susan; MOTA, Welton Costa. Computação em Nuvem. **Especialização em Redes e Segurança de Sistemas. Pontifícia Universidade Católica do Paraná. Disponível em:< http://www. ppgia. pucpr. br/~ jamhour/RSS/TCCRSS08B/Welton% 20Costa% 20da% 20Mota**, 2010.

SANTOS, Carlos J.; FRANCO, Matheus E.; CORSINI, Hagar CC. ALGbr: Uma Nova Ferramenta Para Apoio ao Ensino/Aprendizagem de Lógica Computacional por Meio da Construção e Testes de Algoritmos. Disponível em: http://www.lbd.dcc.ufmg.br/colecoes/wei/2013/0022.pdf. Acessado em: 15 de Setembro 2015.

SANTOS, Rodrigo Pereira dos; COSTA, Heitor Augustus Xavier. Análise de Metodologias e Ambientes de Ensino para Algoritmos, Estruturas de Dados e Programação aos iniciantes em Computação e Informática. **INFOCOMP Journal of Computer Science**, v. 5, n. 1, p. 41-50, 2006.

SCHWARZ, Wolfgang. **Tree Proof Generator**. Disponível em: <http://www.umsu.de /logik/trees/info.html>. Acesso em Setembro de 2015.

SILVA, Eraylson Galdino da; Barbosa, A. F., Neto, S. R., HO, R., e Lopes, A. N. Análise de ferramentas para o ensino de Computação na Educação Básica. 2014.

SILVA, Flávio Soares Corrêa da; FINGER, Marcelo; MELO, Ana Cristina Vieira de. **Lógica para computação**. Thomson Learning, 2006.

SOUZA, João Nunes de. **Lógica para ciência da computação**. Elsevier Brasil, 2008.

WinKE. Disponível em: <https://staff.fnwi.uva.nl/u.endriss/WinKE/.software.php >. Acesso em Agosto de 2015.

ZENI, José Ricardo de Rezende. Inlogic: Uma Ferramenta para o Ensino e Aprendizagem de Lógica. 2007.

# 7. Apêndices

## Formulário sobre Ferramenta de Dedução de Premissas

Formulário com intuito de avaliar e criar melhorias para a ferramenta de Dedução de Premissas desenvolvida.

*Obrigatório

1 **Quanto de lógica você conhecia antes da faculdade? ***

( ) Muito
( ) Médio
( ) Nenhum contato

2 **Qual seu nível de dificuldade com a disciplina? ***

( ) Muito Alto
( ) Alto
( ) Médio
( ) Baixo
( ) Muito Baixo
( ) Nenhum

3 **Quando começou a aprender, qual era o nível de dificuldade em fazer deduções lógicas? ***

( ) Muito Alto
( ) Alto
( ) Médio
( ) Baixo
( ) Muito Baixo
( ) Nenhum

4 **Agora, qual é seu nível de dificuldade em fazer deduções lógicas? ***

( ) Muito Alto
( ) Alto
( ) Médio
( ) Baixo
( ) Muito Baixo
( ) Nenhum

5 **Você acha útil um software para ajudar na dedução lógica? ***

( ) Sim
( ) Não

6 **O que você achou da ferramenta de dedução? ***

( ) Ótimo
( ) Bom
( ) Ruim

( ) Péssimo

7 **O que você achou da aparência da ferramenta de dedução? ***

( ) Ótimo
( ) Bom
( ) Ruim
( ) Péssimo
( ) Outro:

8 **Quanto a ferramenta de dedução te animou a continuar estudando lógica? ***

( ) Muito
( ) Médio
( ) Pouco
( ) Nada

9 **Você gostaria de utilizar a ferramenta de dedução na disciplina de lógica? ***

( ) Sim
( ) Não

10 **Você acha que a ferramenta de dedução deve ser melhorado para ajudar com mais deduções lógicas? ***

( ) Sim
( ) Não

11 **Você gostaria de utilizar softwares que ajudassem com as disciplinas na faculdade? ***

( ) Sim
( ) Não