Michele Macedo Weber

Teoria da Computação - Uma Nova Perspectiva

Trabalho de Graduação apresentado como requisito parcial para Graduação em Ciência da Computação, setor de Ciências Exatas, da Universidade Federal do Paraná.

Área de concentração: *Ciência da Computação*.

Orientador: André Vignatti.

Curitiba PR
2016

Resumo

O trabalho tem como intuito apresentar slides de aulas sobre teoria da computação, no qual qualquer aluno de computação possa ler e entender os tópicos abordados facilmente e que possam adquirir uma visão mais abrangente sobre computação. Os assuntos presentes no trabalho são: conjuntos, em que é abordado os conceitos da lógica de primeira ordem e axiomas da teoria dos conjuntos; infinitos, demonstrando a existência de infinitos maiores que outros e a hipótese do continuum; teorema da completude e os teoremas da incompletude; máquinas, explicando o que são reduções e como funciona uma máquina universal de Turing; tese de Church-Turing e hipercomputadores; discussão sobre inteligência artificial, o quanto um computador pode imitar uma pessoa; decibilidade; P, NP, NP-Completo, coNP, teorema de Ladner e hierarquia polinomial; apresentação do que é aleatoriedade, quais são as classes de complexidade, qual a diferença entre aleatoriedade verdadeira e pseudoaleatoriedade e, por último; criptografia, apresentando como funcionam os geradores de números pseudoaleatórios, a cifra de césar, o one-time pad, as chaves públicas e por último, o RSA.

Abstract

The objective of this term paper is to present slides of lectures about theory of computation, in which any computation student can read and understand the topics covered easily and acquire a bigger vision about computer. The topics present are: sets, in which is discussed the concepts of first order logic and the axioms of set theory; infinity, demonstrating the existence of infinities bigger than others and the continuum hypothesis; completeness and incompleteness theorems; machines, explaining what are reductions and how the universal Turing machine work; Church-Turing thesis and different models of hypercomputers; discussion about artificial intelligence, how a computer could imitate a person; decidability; P, NP, NP-complete, coNP, Ladner's theorem and polynomial hierarchy; presentation of what is randomness, what are the complexity classes and what's the difference between true randomness and pseudorandomness and, lastly; encryption, showing how the pseudorandom generators works, the Caesar cipher, the one-time pad, the public keys and the RSA.

Sumário

Capítulo 1

Introdução

1.1 Motivação

A computação pode ser definida como a busca de uma solução para um problema que utiliza entradas e obtém seus resultados (saídas) a partir de um algorítmo. É com isto que lida a teoria da computação e é isto que nos motivou a escolher este tema de trabalho, encontrar um problema na computação e tentar resolvê-lo. A partir disso, reparamos que o "como"estudamos computação seria um problema muito interessante a ser estudado, afinal, é a partir dele que podemos medir o quanto estamos adiquirindo conhecimento e se essa é o melhor método de fazê-lo.

Desde que entramos no curso de Ciência da Computação da UFPR, pudemos perceber que o curso enfatiza assuntos específicos da área da computação com o intuito de tornar os alunos bons profissonais para as necessidades do mercado. O problema disso se dá ao fato de que a computação tem uma quantidade extensa de informações, mas que não são muito oferecidas durante o curso. O conhecimento repassado não cria uma perspectiva maior do quão grande a computação é e o quanto é possível aprender. Outro problema seria que, muitas vezes as informações são repassadas com muita formalidade, tornando difícil a compreensão, seja pelo vocabulário não usual ou pela forma em que as aulas são apresentadas.

Por estes motivos, propuzemos aulas para temas que possam proporcionar uma perspectiva maior da computação de uma maneira informal, para facilitar o aprendizado a qualquer pessoa que tenha um conhecimento mínimo sobre computação.

1.2 Inspiração

O trabalho utilizou como base o livro de Scott Aaronson, *Quantum Computing since Democritus* [Aaronson, 2013], pois, como pode-se deduzir a partir de seu título, o livro abrange bem os diferentes ramos da computação, desde o início da lógica até computação quântica. A

partir dele, nós escolhemos estudar os capítulos 2 até o 8 e criar aulas com base nos assuntos e explicações do livro.

1.3 Objetivo Geral

Esse trabalho tem por objetivo apresentar aulas informais através de slides, que possam ser de fácil compreensão para os alunos e proporcionar uma visão maior da computação, interligando os diferentes temas dentro da área, desde o início da lógica até a computação quântica.

1.4 Objetivos Específicos

- Pesquisar e entender os conceitos abordados no tema.
- Transmitir o conhecimento adquirido através de aulas em slides.
- Modelar os slides criados de uma forma simples, de fácil entendimento.
- Personalizar os slides para que segure a atenção de alunos utilizando figuras e cores chamativas.

1.5 Contribuição

Criar slides para aulas é uma forma de contribuir para que qualquer pessoa possa ter acesso a esse conteúdo e possa utilizá-lo, seja para entender algum assunto contido nesse trabalho ou para utilizá-los em aulas ou seminários.

Iremos disponibilizar o acesso aos slides neste documento através de um link de um diretório onde todas as aulas estão divididas de acordo com seu assunto e com a duração que cada aula possa ter.

Acreditamos que uma sequência de seminários baseado nos slides possam ajudar os alunos a entenderem o contexto do Curso de Ciência da Computação de uma forma simples e objetiva, criando uma perspectiva maior do que seria computação e o quanto pode ser aprendido nesta área.

1.6 Organização

No primeiro capítulo foi apresentado a introdução deste documento, contendo a motivação, a inspiração, o objetivo geral, os objetivos específicos e a contribuição do trabalho.

No segundo capítulo deste documento são apresentados os assuntos abordados nos slides e quais temas foram estudados dentro de cada assunto. Dentro de cada assunto há uma breve introdução e uma explicação de alguns conceitos abordados dentro de cada tema.

No terceiro capítulo é apresentada a proposta do trabalho, ou seja, a divisão das aulas, como os slides foram criados, o porquê de organizá-los dessa forma e como que as informações estão dispostas neles. Por último, o capítulo mostra o link onde os slides estão disponibilizados e o porquê de não conter o arquivo original que possa ser feita a edição deles.

No quarto capítulo está a conclusão sobre o trabalho, o que nós aprendemos, quais foram as dificuldades e quais são os trabalhos futuros. Por fim, são apresentadas as referências bibliográficas utilizadas durante o trabalho.

Capítulo 2

Tópicos Abordados

Neste capítulo nosso objetivo é dar uma breve introdução sobre cada tópico abordado, mostrando os diferentes conceitos apresentados em cada tópico. Nosso objetivo não é explicar detalhadamente os assuntos, mesmo que informalmente, pois isso já é fato no livro que usamos como base [Aaronson, 2013].

2.1 Conjuntos

Dentro de conjuntos foi apresentado os conceitos de Lógica de Primeira Ordem e suas regras, Axiomas de Peano para Inteiros, Axiomas da Teoria de Conjuntos e Hipótese do Continuum. Também foi discutido sobre infinitos, como existem infinitos maiores que outros infinitos, mostrando como isso é possível e alguns exemplos.

Começamos as aulas falando sobre lógica. A essência de uma discussão concisa nasce da lógica, pois é a partir dela que podemos discutir se algo é verdade ou não, fazendo deduções que nos ajudem a chegar em alguma conclusão, podendo assim, provar algo. Mas para isso, é necessário que tenhamos criado fundamentos básicos onde a lógica se torne óbvia e de senso comum para todos. Dessa forma, foram criados os axiomas que são verdades evidentes muitas vezes utilizados como base de argumentações. Eles são sentenças ou preposições que não são provados ou demonstrados, mas são considerados verdadeiros por serem óbvios universalmente.

Um teorema é uma afirmação que pode ser provada como verdadeira por meio de um conjunto de axiomas ou a partir de outros teoremas já provados. Para isso, as regras da Lógica de Primeira Ordem nos ajuda a demonstrar teoremas onde as deduções da lógica são formalizadas através destas regras. Ela é a teoria aperfeiçoada da Lógica Booleana, pois permite a utilização de variáveis e quantificadores, podendo criar generalizações nas afirmações.

Outro conceito apresentado neste trabalho é o de modelos. Modelos são utilizados para tornar algo simplificado existente na realidade. A partir dessa simplificação, podemos explicar e estudar algo que queremos, como por exemplo, utilizamos um grafo para modelar a questão de

como chegar de um lugar ao outro pelo menor caminho ou, como explicado nos slides,o Axioma de Peano é um modelo para os números inteiros.

Depois disso, falamos do primeiro conjunto de infinitos, o infinito dos inteiros ou, como Cantor chamou, $\aleph$ (*Aleph-Zero*). $\aleph$ é considerado o primeiro conjunto de infinitos, pois é o menor infinito conhecido, o que significa que existem infinitos maiores do que o dos inteiros. Nos slides são apresentados dois infinitos maiores que $\aleph$, o *Power Set* ($2^{\mathbb{N}}$) e Ω (*Ômega*). O *Power Set* é um conjunto consistindo de subconjuntos de todos os elementos x, por exemplo, conjunto A = { a, b, c} e o *Power Set* de A: 2^A ={ $\emptyset$, {a}, {b}, {c}, {a,b}, {a,c}, {b,c}, {a,b,c}}. Outro exemplo é o *Power Set* do conjunto dos números naturais, onde tem todos os subconjuntos dos números naturais. Dessa forma podemos perceber que $|2^{\mathbb{N}}| > |N|$ e, como os dois conjuntos são infinitos, então temos infinitos de tamanhos diferentes.

O conjunto Ω são rótulos dados para os elementos que vêm depois quando acabam todos os números ordinais. Os números ordinais dão rótulos para os elementos dizendo como eles são organizados, ordenados, por exemplo, 1º, 2 º, 3º... Esses rótulos utilizam os números naturais mais a terminação º para essa ordenação, dessa forma podemos concluir que $|\Omega| > |\mathbb{N}|$. A questão que fica depois é se o *Power Set* dos naturais é igual, maior ou menor que o conjunto Ω, ou ainda, se existem infinitos intermediários, entrando assim na discussão sobre a Hipótese do Continuum de Cantor[Burton, 2011], onde em 1938 Godel[Feferman, 1990] demonstrou que a negação da hipótese não pode ser provada e em 1963, Paul Cohen [Cohen, 1963] demonstrou também que a hipótese não pode ser provada.

Logo, os assuntos abordados foram: [Smullyan, 1995] [Aaronson, 2013] [Hamilton, 1982]

- Lógica de primeira ordem e suas regras
- Axiomas de Peano para Inteiros
- Axiomas de Zermelo-Fraenkel para a Teoria de Conjuntos
- Infinitos
- Números Ordinais
- Hipótese do Continuum
- Axioma da Escolha

2.2 Completude

Um sistema formal é dito completo quando ele tem uma propriedade específica que qualquer fórmula que tem esta propriedade, pode ser derivada utilizando este sistema, ou seja, esta fórmula é um de seus teoremas; caso contrário o sistema é dito incompleto.

Dentro de completude foram apresentados o Teorema da Completude e os Teoremas da Incompletude, o que são estes teoremas e como eles funcionam, utilizando exemplos através do problema da parada para explicar. Por último, mostramos como o Teorema de Completude não contradiz os Teoremas da Incompletude.

O Teorema da Completude diz que toda afirmação pode ser demonstrada utilizando apenas as regras da Lógica de Primeira Ordem. O problema é que a Completude pode ser verdadeira apenas para alguns modelos específicos, como por exemplo, o Axioma da Teoria de Conjuntos - Zermelo-Fraenkel, mas não para todos os modelos, o que exclui uma teoria geral para a lógica, que é onde entra o Primeiro Teorema da Incompletude.

O Primeiro Teorema da Incompletude diz que, dado um modelo axiomático capaz de expressar verdades básicas, este não pode ser completo e consistente ao mesmo tempo, ou seja, existe uma afirmação aritmética verdadeira que não pode ser provada. Como exemplo, temos a Hipótese do Continuum e o Problema da Parada.

O Segundo Teorema da Incompletude diz que qualquer modelo axiomático em questão que não inclui afirmações de sua própria consistência, não consegue provar sua própria consistência.

Logo, os assuntos abordados foram: [Franzén, 2007]

- Teorema da Completude
- Teorema da Incompletude 1
- Teorema da Incompletude 2

2.3 Máquinas

Dentro deste assunto foram apresentados o conceito de reduções, como um problema pode ser reduzido ao outro, como funciona uma Máquina de Turing Universal, a Tese de Church-Turing e como a Hipercomputação refuta essa tese, mostrando diferentes modelos da Hipercomputação. Por último, é apresentada uma discussão sobre a questão de o quão longe um computador pode chegar, se ele pode pensar e se pode imitar um ser humano, apresentando também a Conjectura de Robin nesta discussão.

Esse assunto começa falando sobre reduções. Reduzir um problema a outro significa que existe uma transformação na entrada do algoritmo A para a entrada do algoritmo B e que o algoritmo B serve como caixa preta (ou oráculo) para resolver o algoritmo A, retornando assim uma saida transformada que servira de resposta para o algoritmo A. Em outras palavras, se um problema é tão difícil quanto outro problema, um pode se reduzir ao outro. Por exemplo, se um problema A pode ser resolvido usando um algoritmo para B, A não é mais difícil do que B, e assim dizemos que A se reduz a B. Existem dois tipos diferentes de redução, reduções de

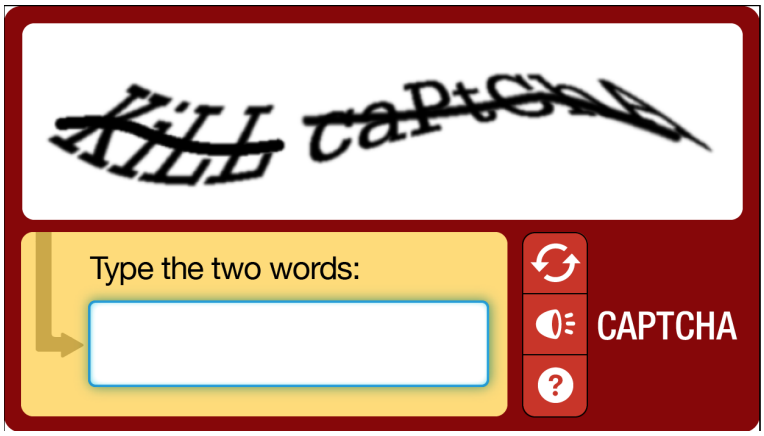
Cook-Levin e reduções de Karp. A diferença entre os dois se dá ao fato de que na redução de Cook-Levin a caixa preta só pode ser utilizada uma única vez, enquanto que a de Karp pode ser utilizada várias vezes.

A máquina de Turing Universal é uma máquina que pode simular qualquer outra máquina de Turing, até ela mesma, por exemplo, uma máquina de café que é ao mesmo tempo uma torradeira, um liquidificador, um microondas e várias outras máquinas. Em outras palavras, essa máquina universal receberia como entrada qual máquina ela deveria simular e o que ela deveria fazer, criando assim a mesma saída que uma máquina específica para tal ação reproduziria.

A hipercomputação refere-se a um modelo de uma máquina de Turing mais poderosa, que resolve problemas onde a máquina de Turing comum não consegue, como por exemplo, o problema da parada. Até hoje ninguém conseguiu criar um hipercomputador e acredita-se que não pode ser criado fisicamente, caso contrário, violaria a Tese de Church-Turing na qual afirma que qualquer função "eficientemente computável"que possa ser computada por uma pessoa utilizando apenas um conjunto de algoritmos, ela também pode ser computada por uma máquina de Turing.

Por último, as aulas entram na discussão sobre se um computador pode pensar ou imitar um ser humano utilizando o Jogo da Imitação para dar inicio a esta discussão. Este jogo funcina da seguinte maneira: existem 3 pessoas, uma mulher, um homem e um interrogador (independente do sexo). O interrogador deve ficar em um local onde ele não possa ver ou ouvir a voz dos outros 2 participantes. Ele deve descobrir quem é homem e quem é mulher através de perguntas, utilizando um computador ou outra ferramenta para a troca de informações e, o homem e a mulher podem mentir em suas respostas para despistar o interrogador. A questão desse jogo é, e se substituirmos o homem ou a mulher por um computador? O computador poderia imitar um dos dois? Ou ele realmente só está imitando porque ele foi programado por alguém que o deixou com essa capacidade? Ou ainda, ele adquiriu conhecimento suficiente e realmente aprendeu a imitar um ser humano?

Em contra-partida é apresentado a questão dos *captchas*, onde os computadores não conseguem, por exemplo, ler uma palavra rabiscada por cima, demonstrando assim, uma limitação para imitar ou pensar igual a um ser humano.



Isso ocorre, pois os computadores não foram programadas para identificar todas as formas que as letras possam assumir, apenas para identificar padrões pré-definidos. Mas apesar dessas limitações, o computador já teve a capacidade de resolver problemas que os seres humanos não conseguiram, como por exemplo, a Conjectura de Robin onde, dados três axiomas: A or (B or C) = (A or B) or C, A or (B or C) = (A or B) or C, Not(Not(A or B) or Not(A or Not(B))) = A, um deles pode derivar Not(Not(A)) = A? Por fim, a última questão que é apresentada é qual a diferença entre o pensar, entender e gravar na memória de um ser humano com o computador, entrando assim no problema da Sala Chinesa, onde um ser humano que não fala chinês tenta resolver perguntas em chinês utilizando um livro para ajudá-lo. Neste problema, a pessoa pode decorar o livro e resolver essas questões corretamente, mas não significa que ela entendeu. Esse é o caso do computador, ele utiliza um banco de dados (que no nosso caso seria a nossa memória) para resolver as perguntas.

Logo, os assuntos abordados foram: [Syropoulos, 2008] [Russell et al., 2003]

- Reduções
- Máquina Universal de Turing
- Tese de Church-Turing
- Hipercomputadores
- Jogo da Imitação
- Captchas
- Conjectura de Robin

2.4 P e NP

Dentro deste assunto são apresentados os conceitos de P, NP e todos os outros problemas que estão relacionados a ele, como NP-Completo, se P=NP, coNP, se coNP = NP ou se NP $\cap$ coNP, PSPACE, EXP, Decibilidade, TIME(f(n)), SPACE(f(n)), Teorema da Hierarquia do Tempo, Teorema de Aceleração de BLUM, Algoritmo de Gale-Shapley, Teorema de Ladner, Fatoração e Algoritmo de Shor.

TIME($f(n)$) é a classe dos problemas no qual, para qualquer instância de tamanho n, são solucionáveis em uma máquina de Turing em uma quantidade de tempo proporcional a $f(n)$. Esta classe tem relação com P e EXP onde em P é a união de todos os inteiros positivos K de TIME(n^k) e EXP é a união de todos os inteiros K de TIME(2^k). SPACE($f(n)$) é a classe dos problemas solucionáveis por uma máquina de Turing usando uma quantidade de espaço que

cresce proporcionalmente a $f(n)$. Esta classe tem relação com PSPACE onde é a união de todos os inteiros K de SPACE(n^k).

A classe NP é a classe onde estão todos os problemas resolvidos em tempo polinomial por uma máquina de Turing não-determinística, ou seja, sua resposta pode ser verificada em tempo polinomial. O Teorema de Ladner diz que se P é diferente de NP então existem problemas intermediários entre eles, isto é, um problema pode ser resolvido em P ou em NP dependendo da entrada de tal problema.

Os problemas NP-Completos são os problemas onde todos eles são o mesmo problema, em outras palavras, um problema pode-se reduzir a outro fazendo com que, se você conseguir resolver um desses problemas em um oráculo, todos os outros, consequentemente, podem ser resolvidos também. Da mesma forma, resolvendo um problema NP-Completo, seria possível reduzir um problema NP em um problema NP-Completo. Um exemplo de problema NP-Completo é o problema do caixeiro viajante que consiste em determinar o circuito mais curto possível entre as cidades de uma determinada lista, de modo que cada cidade seja visitada uma única vez.

Uma questão que discutimos nas aulas é se P é igual a NP e como podemos definir se um problema pertence a P ou NP. O Problema do Casamento Estável é um exemplo de problema que aparentemente é complicado, que parece pertencer a classe NP-Completo, mas na verdade não é, existe um algoritmo de tempo polinomial para resolver este problema, isto é, nem sempre é evidente que um certo problema é fácil de resolver então não podemos julgar se um problema é NP ou P apenas olhando para ele, temos que tentar resolvê-lo primeiramente para então definir a que classe ele pertence.

Logo, os assuntos abordados foram: [Arora e Barak, 2009] [Gusfield e Irving, 1989]

- Tarski - Decibilidade
- TIME(f(n))
- SPACE(f(n))
- Teorema da Hierarquia do Tempo
- Teorema de Aceleração de Blum
- P
- NP
- PSPACE
- EXP
- P = NP?
- Algoritmo de Gale-Shapley

- NP-Completo
- Circuit SAT
- SAT
- Redução de qualquer problema NP para um NP-Completo
- Mapa de Coloração
- Empacotamento
- Clique
- Teorema de Ladner
- coNP
- NP = coNP?
- NP $\cap$ coNP
- Fatoração
- Algoritmo de Shor
- Hierarquia Polinomial

2.5 Aleatoriedade

Dentro de aleatoriedade é mostrado o que é probabilidade utilizando exemplos, como funciona a Linearidade de Esperança e o Limite de Chernoff. Também é explicado porque a aleatoriedade é importante dentro da computação utilizando um algoritmo como exemplo, explicando como que a aleatoriedade funciona dentro dele e se realmente é importante. Dentro da explicação desse algoritmo, é apresentado como encontrar um número primo através do Pequeno Teorema de Fermat, qual o problema deste teorema e qual algoritmo funciona de fato para resolver o problema do número primo em tempo polinomial, mas com uma probabilidade de erro. Dentro deste assunto também é falado sobre as Classes de Complexidades existem tais como, PP, BPP, RP, coRP, ZPP, P/f(n) e P/poly e algumas questões sobre esses assuntos tais como se P=P/Poly, se P=BPP e como BPP $\subseteq$ P/Poly.

Aleatoriedade é quando algo que irá acontecer é incerto, sujeito ao acaso, imprevisível, que não há um padrão determinístico, mas sim uma distribuição de probabilidade. Por este motivo, as aulas começam com o assunto de probabilidade, descrevendo e mostrando exemplos do que é probabilidade.

Depois é mostrada a aplicação da aleatoriedade na computação. Intuitivamente, utilizando a aleatoriedade em algoritmos torna o algoritmo mais poderoso, pois permite tempo de execução esperado onde sua corretude é menor que 100 por cento, ao invés de determinístico onde a corretude é sempre 100 por cento. Na teoria, ainda ninguém sabe ao certo se ela é de fato mais poderosa, o único fato é que ela é mais flexível.

O algoritmo de Monte Carlo são sempre rápidos, mas apenas provavelmente corretos. Por outro lado, os algoritmos de Las Vegas estão sempre corretos, mas apenas provavelmente rápidos. Os algoritmos de Atlantic City é a "junção" desses dois algoritmos, pois são algoritmos probabilísticos de tempo polinomial onde são provavelmente corretos e provavelmente rápidos.

Logo, os assuntos abordados foram: [Mitzenmacher e Upfal, 2005] [Burton, 2011]

- Probabilidade
- Linearidade de Esperança
- Limite de Chernoff
- Pequeno Teorema de Fermat
- PP
- BPP
- RP
- coRP
- ZPP
- P/f(n)
- Desuniformidade
- P/Poly
- BPP $\subseteq$ P/Poly
- Desaleatoriedade
- P = BPP?

2.6 Criptografia

Sobre este assunto foram mostrados diferentes tipos de criptografias, desde uma das mais antigas, como a Cifra de César, como uma das mais atuais, o RSA. Foi mostrado o funcionamento da Cifra de César, do One-Time Pad, Chaves públicas, RSA e Geradores Pseudo-Aleatórios. Sobre os Geradores também foi explicado o que ele é e pra que serve. Por último, é introduzida a questão de computadores quânticos, se eles poderão quebrar os algoritmos de criptografias atuais e como poderíamos resolver este problema.

Criptografia são técnicas que transformam informações em algo ilegível, onde apenas o(s) destinatário(s) tem capacidade de transformar essa informação ilegível em algo legível utilizando uma chave, que nada mais é do que uma informação que permite essa transformação.

A criptografia tem quatro objetivos principais: gerar confidencialidade da mensagem onde apenas o destinatário deve ser capaz de decifrar a mensagem, integridade mensagem onde apenas o destinatário deverá ser capaz de determinar se a mensagem foi alterada durante a transmissão, autenticidade onde o destinatário deverá ser capaz de identificar o remetente e verificar se foi ele mesmo quem enviou a mensagem e, por último, de não repúdio onde o destinatário não pode negar a autoria da mensagem.

A Cifra de César também é conhecida como cifra de troca, código de César ou troca de César e é uma das mais simples e conhecidas técnicas de criptografia. Ela recebeu este nome em homenagem a Júlio César, que o usou para se comunicar com os seus generais.

O motivo pelo qual é impossível que o computador possa gerar números verdadeiramente aleatórios é que ele é determinístico. As duas formas de se obter números aleatórios são então, ou utilizar algum componente físico que gere sequências verdadeiramente aleatórias, como por exemplo, componente que captura os ruídos de som e transforma esses ruídos em números, ou podemos utilizar um computador que gere sequências que pareçam ser aleatórias, que é o caso dos geradores pseudoaleatórios.

Criptografia de chave pública também é conhecida como criptografia assimétrica. Ela requer duas chaves, uma delas sendo secreta (ou privada) onde ela não é distribuída, apenas o(s) destinatário(s) tem acesso a ela, e a outra delas sendo pública, que é conhecida por todos. A chave pública é usada para criptografar informação ou para verificar uma assinatura digital; já a chave privada é usada para a operação oposta, para descriptografar a informação ou para criar uma assinatura digital. O termo assimétrico vem do uso de diferentes chaves para realizar essas funções opostas, diferente da criptografia simétrica, onde ela depende da mesma chave para realizar ambas as funções.

Logo, os assuntos abordados foram:[Paar e Pelzl, 2009] [Peterson e Davie, 2007]

- Cifra de Cesar
- One-Time Pad

- Geradores de Números Pseudoaleatórios
- Chaves públicas
- Chaves privadas
- RSA
- Shortest Vector Problem

Capítulo 3

Proposta

Neste capítulo nosso objetivo é apresentar a proposta deste trabalho, como os assuntos foram organizados de acordo com uma estimativa de tempo de duração de uma aula e como foram desenvolvidos os slides, mostrando alguns exemplos.

Os slides por si só muitas vezes não explicam bem os conceitos, mas estamos partindo do pressuposto que o livro do Scott Aaronson[Aaronson, 2013] seja usado como livro texto e que as explicações sejam feitas durante as aulas.

3.1 Divisão das Aulas

Os assuntos foram divididos em 16 aulas, sendo cada aula contendo aproximadamente uma hora de duração.

As três primeiras aulas seriam sobre Conjuntos, onde a primeira fala sobre as regras da Lógica de Primeira Ordem, Axioma de Peano para Inteiros e o Axioma de Teoria de Conjuntos,a segunda começa falando sobre o quão grande os conjuntos podem ser e termina a aula dando o exemplo dos conjuntos egoístas e os não-egoístas. A terceira aula termina o assunto de conjuntos, começa falando sobre os números ordinais e termina a aula falando sobre o Axioma da Escolha.

A quarta aula fala sobre o Teorema da Completude e os Teoremas da Incompletude.

A quinta aula começa sobre o assunto de máquinas, falando sobre Reduções, Tese de Church-Turing e começando o assunto sobre Hipercomputadores.

A sexta aula continua o assunto sobre Hipercomputadores e entra na discussão sobre se máquinas podem pensar.

Na sétima aula, começa o assunto sobre P e NP, falando sobre Decibilidade, TIME(f(n)), SPACE(f(n)), relação entre essas duas classes, Hierarquia do Tempo e Teorema de aceleração de Blum.

Na oitava aula é explicado o que é um problema P, NP, PSPACE, EXP, se P=NP, o Algoritmo de Gale-Shapley e começa a explicação de como reduzir um problema NP-Completo em outro NP-Completo.

Na nona aula continua a explicação sobre como reduzir um problema NP-Completo ao outro e fala sobre o Teorema de Ladner, coNP, Fatoração e Algoritmo de Shor.

Na décima aula começa o assunto sobre Aleatoriedade. Nesta aula começa falando sobre Probabilidade, Linearidade de Esperança e Limite de Chernoff e depois, por último, termina a aula falando sobre porquê aleatoriedade é importante.

Na décima primeira aula é continuado o assunto sobre o porquê a aleatoriedade é importante, mostrando e explicando um algoritmo que utiliza a aleatoriedade.

Na décima segunda aula começamos o assunto sobre Classes de Complexidade e mostramos o que é desuniformidade. Na décima terceira aula terminamos o assunto sobre aleatoriedade explicando o que é P/Poly, Desaleatorização e se P=P/Poly, se P=BPP e como BPP $\subseteq$ P/Poly.

Na décima quarta aula começa o último assunto abordado no trabalho, Criptografia. Nesta aula começa a contar um pouco sobre a história da criptografia e como que funciona a Cifra de César e, por último, é explicado o funcionamento do One-Time Pad.

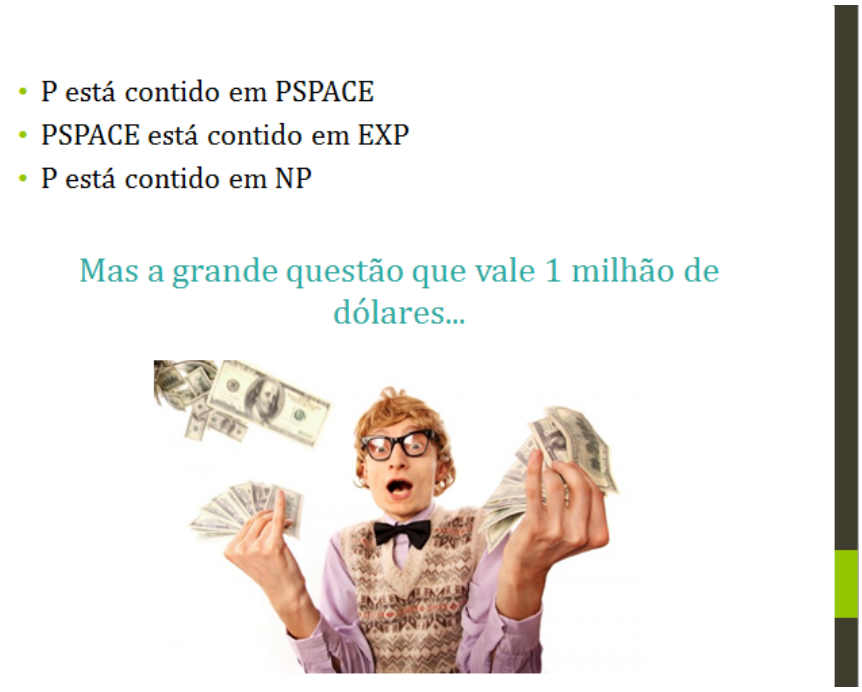
A décima quinta aula é dedicada exclusivamente a explicação de Geradores de Números Pseudoaleatórios, dando um exemplo de um gerador e como ele funciona e qual a diferença entre números verdadeiramente aleatórios e números pseudoaleatórios.

Na décima sexta e última aula é terminado o assunto sobre criptografia. Nele é explicado o que é e como funciona as Chaves Públicas e o RSA e, por último, como o computador quântico poderá mudar a forma que utilizamos a criptografia hoje.

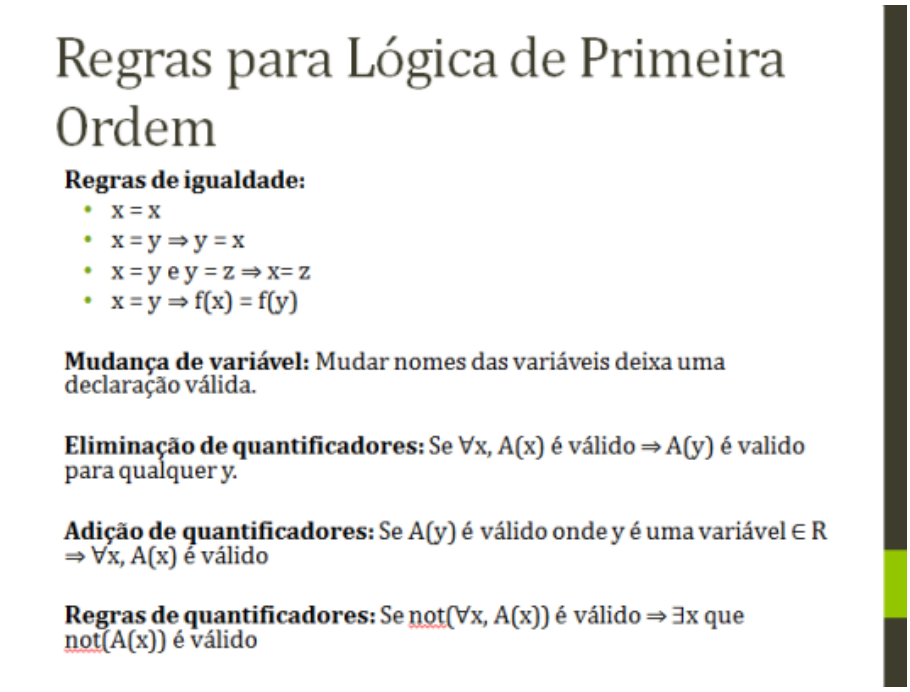
3.2 Apresentação dos Slides

Os layouts dos slides foram feitos de forma que ficassem simples, objetivos, com figuras e que fosse possível o aluno ler e entender o conteúdo só olhando para eles, mas ao mesmo tempo, tentando evitar muito texto para que as explicações do conteúdo fossem falados durante a aula.

Exemplo de slide onde tem pouco texto e os detalhes sobre o porquê de P está contido em PSPACE, PSPACE está contido em EXP e P está contido em NP estão "ocultos"para serem explicados durante as aulas. Mesma a explicação em aula é suposta ser uma explicação principalmente da ideia, deixando de lado o formalismo e os detalhes.

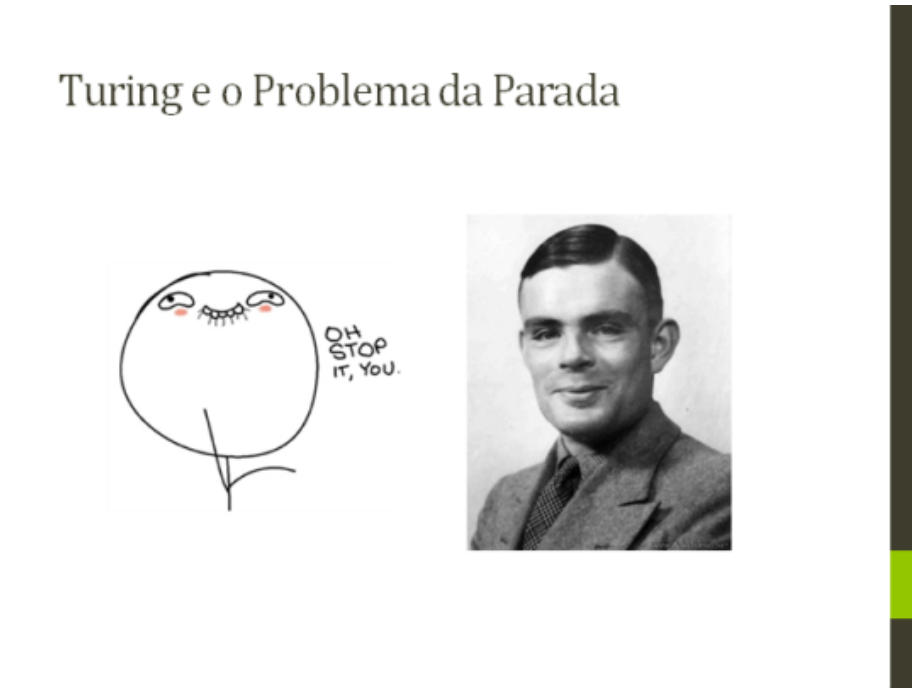


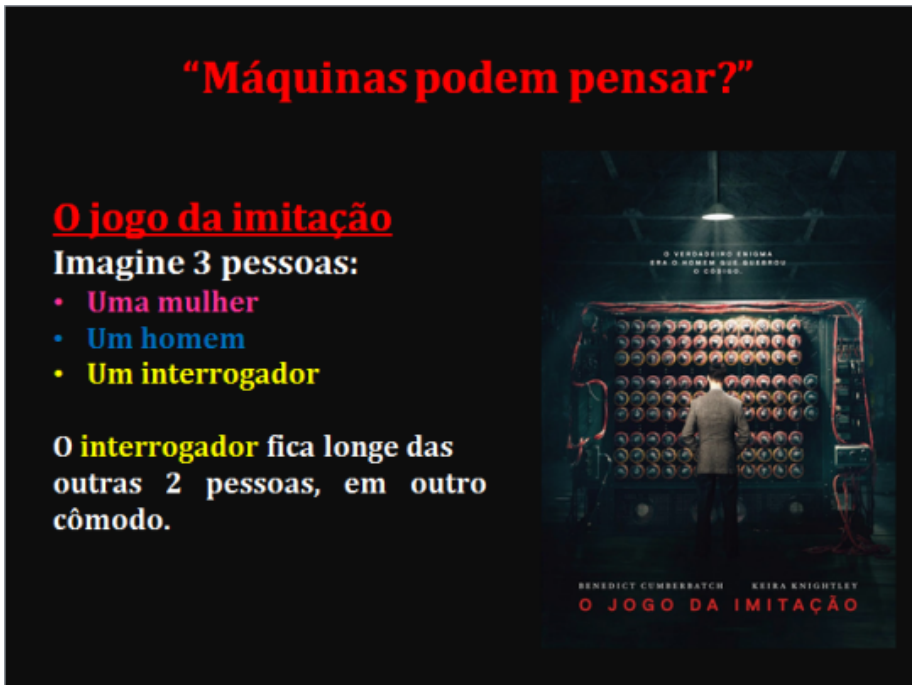
O conteúdo foi escrito de forma coloquial, tentando evitar a formalidade para que as aulas não ficassem cansativas e os alunos entendessem sem muita dificuldade. Em alguns casos, alguma formalidade não foi possível evitar, como por exemplo, as Regras de Primeira Ordem ou a definição de teoremas:



Exemplo de slide de fácil entendimento:

Um outro detalhe nos slides é que eles tentassem chamar a atenção dos alunos, seja por cores ou imagens, para que, dessa forma, ficasse mais fácil dos alunos lembrarem do conteúdo relacionando com essas imagens ou cores e também, para que a aula não ficasse cansativa.





O objetivo geral foi criar aulas simples para que qualquer aluno, sem conhecimento muito profundo sobre teoria da computação, pudesse entender sem muitas dificuldades.

3.3 Link dos Slides

Os slides estão disponibilizados neste link:

https://goo.gl/sWTNLq

Os slides disponibilizados não contém o arquivo original, ou seja, um arquivo editável, apenas o PDF. A razão disso se dá ao fato de que nós não queremos que uma pessoa qualquer possa editá-los como queira e dar-se os créditos. Caso um profissional ou uma pessoa que tenha uma mínima formação acadêmica venha pedir para nós o acesso, nós poderemos dar, dessa forma poderemos ter certeza que os slides não serão utilizados de forma inadequada.

Capítulo 4

Conclusão

Ler, entender e ensinar para alguém o que foi estudado de uma forma simples não é fácil pelo fato de que, na maioria das vezes que acreditamos que tenhamos entendido algo, podemos perceber que na verdade não, pois, caso contrário, seria fácil de repassar para os outros o que foi estudado. Em compensação, pudemos perceber que essa é a melhor forma de aprender, pois só assim, teremos certeza de que algo foi de fato compreendido.

Depois de estudarmos de forma mais geral os assuntos, pudemos ter uma perspectiva maior da computação e assim entender o porquê aprendemos certos assuntos na faculdade. Um exemplo seria os problemas NP-Completo. No curso de Ciência da Computação da UFPR não é explicado a razão pela qual é importante estudar P, NP e NP-Completo, mas depois de estudarmos neste trabalho sobre hipercomputação e máquina universal de Turing, pudemos entender qual a relação deles com os problemas NP-Completo e assim, ter uma perspectiva maior de como isso é importante.

4.1 Trabalhos Futuros

Dar continuidade ao estudo através do livro base [Aaronson, 2013], desenvolver novas aulas sobre assuntos não abordados e colocar em prática o que foi feito, dando palestras ou aulas de fato.

Referências Bibliográficas

[Aaronson, 2013] Aaronson, S. (2013). *Quantum computing since Democritus*. Cambridge University Press.

[Arora e Barak, 2009] Arora, S. e Barak, B. (2009). *Computational Complexity: A Modern Approach*. Cambridge University Press.

[Burton, 2011] Burton, D. (2011). *The history of mathematics: An introduction.* McGraw-Hill Companies.

[Cohen, 1963] Cohen, P. J. (1963). The independence of the continuum hypothesis.

[Feferman, 1990] Feferman, S. (1990). Collected works, volume it, publications 1938-1974.

[Franzén, 2007] Franzén, T. (2007). *Godel's Theorem: An Incomplete Guide to Its Use and Abuse*. Springer.

[Gusfield e Irving, 1989] Gusfield, D. e Irving, R. W. (1989). *The stable marriage problem: structure and algorithms*. MIT press.

[Hamilton, 1982] Hamilton, A. G. (1982). *Numbers, Sets and Axioms: The Apparatus of Mathematics*. Cambridge University Press.

[Mitzenmacher e Upfal, 2005] Mitzenmacher, M. e Upfal, E. (2005). *Probability and Computing: Randomized Algorithms and Probabilistic Analysis*. Cambridge University Press.

[Paar e Pelzl, 2009] Paar, C. e Pelzl, J. (2009). *Understanding cryptography: a textbook for students and practitioners*. Springer Science & Business Media.

[Peterson e Davie, 2007] Peterson, L. L. e Davie, B. S. (2007). *Computer networks: a systems approach*. Elsevier.

[Russell et al., 2003] Russell, S. J., Norvig, P., Canny, J. F., Malik, J. M. e Edwards, D. D. (2003). *Artificial intelligence: a modern approach*. Prentice hall Upper Saddle River.

[Smullyan, 1995] Smullyan, R. M. (1995). *First-order logic*. Courier Corporation.

[Syropoulos, 2008] Syropoulos, A. (2008). *Hypercomputation: computing beyond the Church-Turing barrier*. Springer Science & Business Media.