Hellon Canella Machado

# O Uso do Pensamento Computacional como Recurso para o Desenvolvimento da Apredizagem Científica.

Rio de Janeiro. Brasil

2018

# 1 Computadores e a prática científica

A evolução da computação nas últimas décadas, aliada ao seu barateamento, tem produzido impactos notáveis tanto na academia quanto na industria. A disponibilidade de dispositivos mais baratos e com maior capacidade de processamento e armazenamento tem tornado a computação ubiqua.

Na academia, esse fenômeno tem favorecido o surgimento de novas estratégias para exploração de fenômenos. Até meados do século 20, todo progresso cientifico foi conduzido apenas por interações entre atividades experimentais e analíticas[1]. O surgimento da computação e o seu desenvolvimento desde então trouxe consigo novas formas de fazer ciência, tais como a simulação e a modelagem computacional, e mais recentemente, a mineração de dados e o aprendizado de máquina, úteis para a análise de grande volume de informação (DJORGOVSKI, 2005; WING, 2006).

O uso de simulações numéricas, por exemplo, se justifica ao permitir que um grande número de fenômenos muito complexos sejam analiticamente tratáveis. Em muitos casos essa é única forma de exploração possível. Mesmo na mecânica newtoniana mais simples, é possível resolver exatamente, apenas, o problema de dois corpos. Para $N \geq 3$, soluções numéricas são necessárias. Da astronomia podemos retirar alguns exemplos, como a formação de estrelas e galáxias e explosão estrelares - de modo geral, qualquer evento envolvendo turbulência (DJORGOVSKI, 2005).

O uso de métodos computacionais, tal como a simulação, tem expandido a abrangência de sistemas não lineares que tem sido explorados pelo modelos matemáticos e computacionais. Como lembra Weintrop et al. (2016), campos da ciência estão experimentando um renascimento de abordagens experimentais em razão do acesso facilitado a mais poder computacional.

O autor destaca que num passado recente, para muitos pesquisadores apenas o estudo de sistemas determinísticos era viável, tendo o termo 'não-linear', praticamente, o sinônimo de 'insolúvel'. Havia desse modo a propensão à investigação computacional apenas de sistemas lineares. Esse quadro era especialmente verdade para muitas pesquisas em biologia e química.

Esse processo ganha especial relevância ao nos darmos conta da natureza caótica da ampla maioria dos fenômenos físicos. Sistemas lineares e determinísticos são portanto exceções, e não a regra (WEINTROP et al., 2016).

[1] Em sentido mais amplo, quando mencionamos 'atividade analítica' estamos nos referindo à utilização de aparato matemático para resolução teórica de problemas. Ao mesmo tempo, o termo análise também pode fazer referência ao emprego da análise matemática, ramo da matemática que lida com conceitos do cálculo diferencial, tais como diferenciação, integração, e séries infinitas.

Outros agentes de transformações da prática científica, embora mais recentes e em fase nascente, são as novas possibilidades trazidas pela abundância e pelo barateamento do armazenamento grandes volumes de dados. Vivemos uma era onde a disponibilidade de dados gerados por câmeras, sensores, execução de simulações e registro de interação humana crescem exponencialmente.

Nesse cenário, como ressalta Djorgovski (2005), o foco de valores tem mudado da propriedade de dados ou de instrumentos para a reunião de *expertise* que tornam possíveis a extração de significados desse volume de informação.

A abundância traz consigo muitos desafios. A taxa com que cientistas e engenheiros têm coletado e produzido dados vem exigindo avanços nas estratégias de análise. O acúmulo chegou a um nível de complexidade que, de certo modo, tem sido impossível fazer qualquer tipo de investigação superficial utilizando técnicas convencionais, tendo como instrumento apenas a percepção humana. Lidar com esse conjunto desestruturado e rico de dados, extraindo dele significado, tem sido portanto uma das batalhas da revolução científica e industrial em curso (DJORGOVSKI, 2005). Nesse contexto, o emprego de inteligência artificial tem sido essencial.

Em linhas gerais, tal "inteligência", expressa em técnicas de aprendizagem de máquina (do inglês: *machine learning*), baseia-se no uso algoritmos que instruem computadores como avaliar dados e deles extrair padrões e correlações, permitindo-os, de forma extraordinária, a fazer predições. E esse processo tem um componente recursivo: quanto mais análises são feitas, mais experiência e competência são adquiridas. Ou seja, mais 'inteligentes' essas máquinas se tornam.

Escobar (2017) propõe uma visão simplificada das interações humanas que facilita o entendimento desses algoritmos. Como descreve, ao conhecemos alguém pela primeira vez, baseando-nos em modelos pessoais, somos capazes de dizer nos primeiros minutos se essa pessoa nos transmite boa ou má impressão. Para cada nova pessoa que encontramos, avaliamos algumas de suas características e as registramos. Esse processo nos permite refinar e recompor modelos sociais que irão influenciar outras percepções em interações futuras.

É exatamente nesse princípio recursivo que se fundamenta o aprendizado de máquina: categorizar dados de acordo com suas características com o fim de compor e refinar modelos.

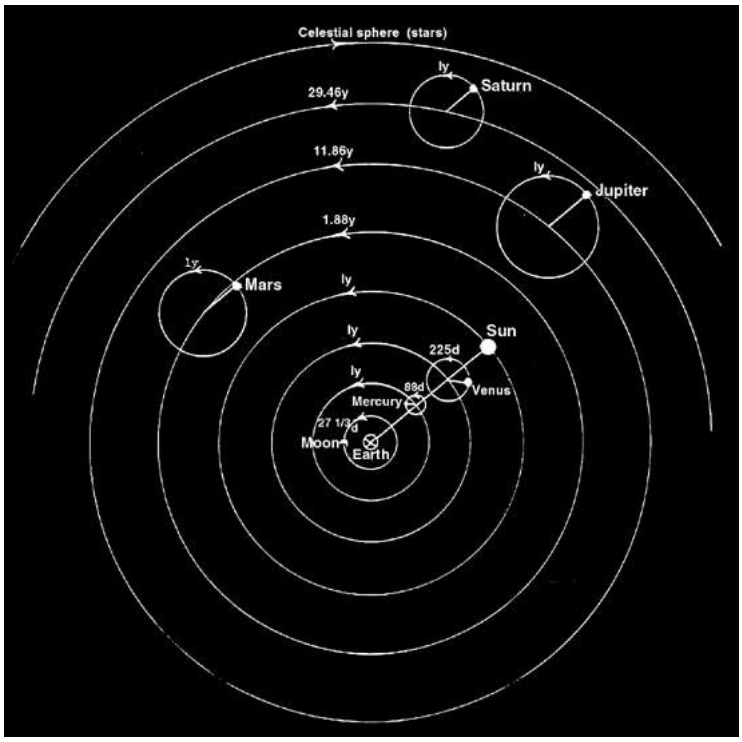
Esse processo tem se provado extremamente eficiente na predição da configuração de sistemas estocásticos. Tomemos por exemplo a necessidade de prever o tempo, onde há dominância de comportamentos turbulentos. Em um estudo recente (PATHAK et al., 2018 apud VUTHA, 2018), mostrou-se a eficiência do uso de algoritmos de inteligência artificial para previsões ao longo de um período muito maior do que se imaginou possível.

Um ponto digno de nota: o algoritmo utilizado não continha nenhuma informação sobre as equações subjacentes.

Há atualmente questionamentos sobre até que ponto o uso intensivo de 'robôs oráculos' levará à perda do interesse dos pesquisadores em descrever fenômenos em termos de equações e princípios unificadores. Esse debate é sintetizado pela confronto das noções de "predição" e "entendimento".

Vutha (2018) expõe os elementos desse debate com um exemplo histórico. Como destaca, durante mais de um milênio, o movimento dos planetas eram descritos a partir de um modelo elaborado por Ptolomeu, cujos métodos lançavam mão de cálculos misteriosos envolvendo sobreposição de círculos. Apesar de ignorar a teoria da gravidade e de ter a Terra no centro do Universo, essa representação foi extremamente eficiente na sua capacidade preditiva. Ao mesmo tempo, pode-se dizer que ela era incompleta na medida em que não oferecia nenhum entendimento capaz de explicar o seu funcionamento.

Figura 1 – Modelo geocêntrico de Ptolomeu




Fonte: Hatch (1998)

A descrição do movimento dos corpos celestes só foi finalmente compreendida a partir da equações diferenciais descobertas por Isaac Newton. Com elas tem sido possível, desde então, prever a trajetória de todo e qualquer planeta do sistema solar.

A qualidade da proposta de Newton estava na possibilidade de oferecer não apenas

a descrição de trajetórias, mas por permitir também que entendamos o porquê que elas são de uma e não de outra forma. Ao nos trazer equações, ele nos facultou a compreensão do fenômeno do movimento a partir da ótica de princípios unificadores. E é exatamente nesse ponto que reside o poder da descrição matemática. Como analisa Vutha (2018), se formos capazes de extrair de um fenômeno complicado dois ou três princípios, podemos dizer então que o compreendemos.

Porém, dada a dominância de fenômenos complexos na natureza, tem sido extremamente difícil extrair de muitos deles princípios simples. Descrevê-los, portanto, a partir da obtenção de equações universalmente válidas pode ser uma forma um tanto ineficiente de gerar predições relevantes (VUTHA, 2018). O emprego de técnicas de *machine learning*, nesse sentido, tem sido essencial.

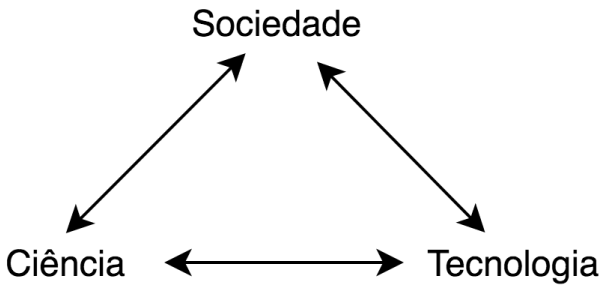
A inteligência artificial tem ocupados outros espaços além da pesquisa científica. Desde robôs que leem a web para tomar decisão de investimentos (VERHAGE, 2017) e carros que se auto dirigem, até contextos relativamente mais simples como tradutores automáticos e mecanismo buscas. Já no mundo do trabalho, um sem-número de projeções apontam para a substituição progressiva de quantidade considerável de atividades laborais por robôs.


Figura 2 – Carro autônomo da Waymo, subsidiária de veículos da Alphabet (matriz do Google)

A produção estratosférica de dados e a invasão da inteligência artificial nos meios científicos e na indústria é sintomático da interação dinâmica entre a ciência, a tecnologia, e a sociedade em ciclos que se retroalimentam.

Novas descobertas levam a novas perguntas que exigem mais coleta de dados. Quando analisadas, essas informações são utilizadas para refinar e ajustar modelos, levando a novas perguntas e mais produção de dados. Esse processo tende a culminar na criação de novas tecnologias. Tecnologias, por seu turno, inspiram usos sociais criativos que quase sempre criam demandas por mais descobertas científicas e mais tecnologia. Sob esse aspecto, podemos dizer que esses três atores desempenham o papel de forças motrizes da computação (WING, 2008).

Figura 3 – Forças motrizes da computação




Adaptado de Wing (2008)

O barateamento dos custos de produção científica é outro aspecto relevante dessas transformações. À medida que a internet e dados, consequentemente, se tornam mais acessíveis, qualquer pessoa com boas ideias e bons hábitos de trabalho, de qualquer lugar, pode fazer ciência de primeira linha, comunicar seus resultados e aprender com a comunidade científica. Essa possibilidade é especialmente benéfica para países e instituições que não dispõem de instalações de pesquisa sofisticadas (DJORGOVSKI, 2005).

Sobre o papel da ciência da computação no desenvolvimento científico, Djorgovski (2005) faz um consideração provocativa:

> [...] a ciência da computação aplicada está desempenhando o papel que a matemática fez do século XVII ao século XX: fornecer uma estrutura ordenada e formal e um aparato exploratório para outras ciências. Além de seu aparentemente feliz caso com a teoria das cordas, é difícil dizer o que a matemática está fazendo para outras ciências hoje. A maioria dos cientistas de matemática que utilizamos hoje foi desenvolvida há mais de um século. (DJORGOVSKI, 2005, p. 131. Tradução nossa)

E é do do seio dessa reflexão que nasce a noção de um "pensamento computacional". Tema que desdobraremos a seguir.

# 2 Pensamento computacional

A noção de pensamento computacional tem ganhado força desde 2006, ano em que a professora Jeannette Wing, então professora da Universidade Carnegie Mellon, publica um artigo seminal no qual propõe um conjunto de atitudes, ou abordagens, para a resolução de problemas fundamentadas na forma como cientistas da computação e programadores tratam problemas computacionais.

A definição clássica que muitos autores dão para o conceito é extraído do próprio trabalho da autora que estabelece que pensamento computacional consiste "numa abordagem para a solução de problemas, desenho de sistemas e entendimento do comportamento humano que se vale de conceitos fundamentais para ciência da computação" (WING, 2006, Tradução nossa), ou ainda, na capacidade de formular problemas e expressar suas soluções de forma suficientemente clara de tal modo que um computador possa executá-las.

O desenvolvimento do tema requer algumas discussões preliminares. Comecemos com uma apresentação sobre o papel dos algoritmos.

## 2.1 Algoritmos

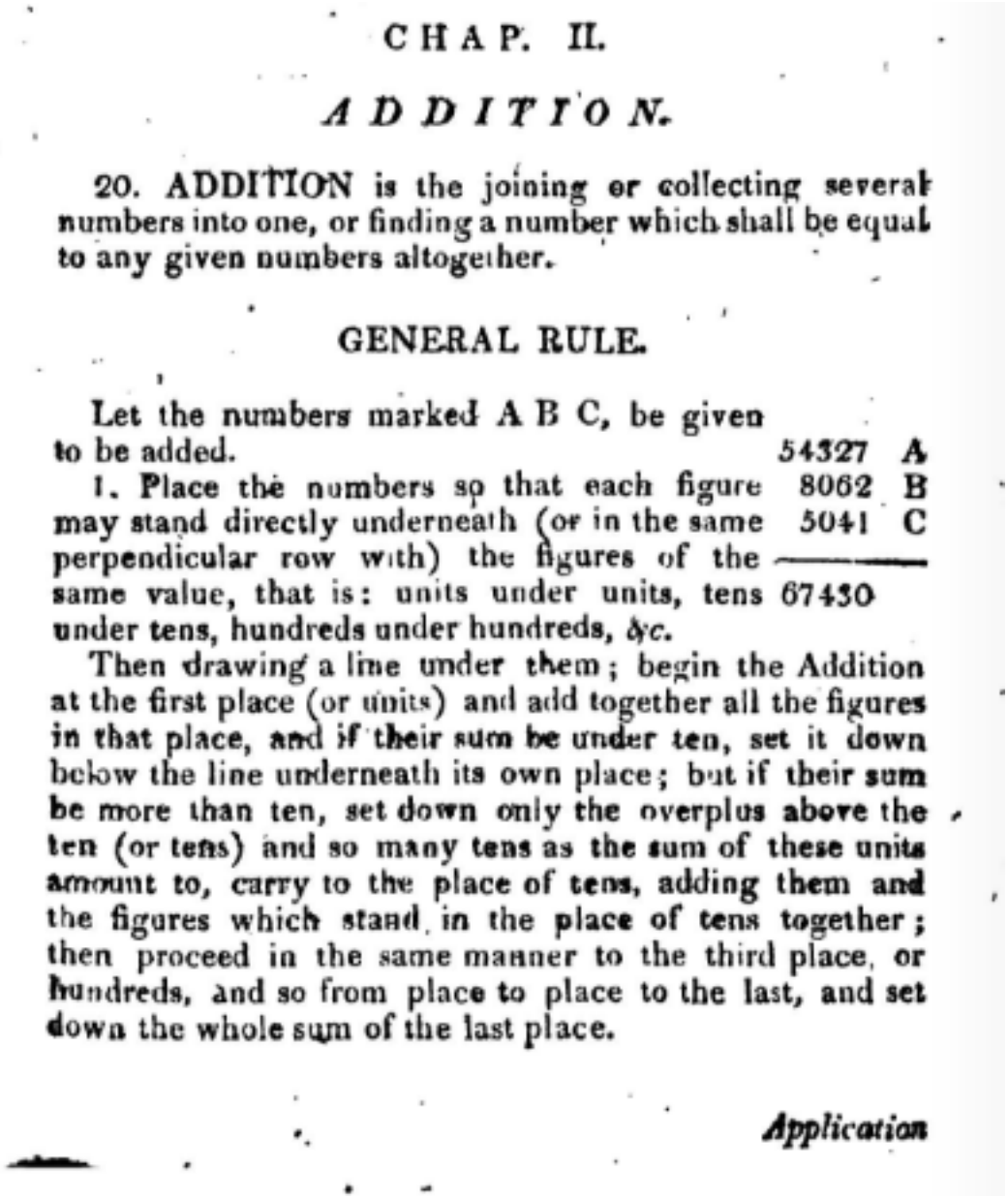
Em termos simplificados, um algoritmo nada mais é do que uma solução para um problema que satisfaz as seguintes condições (PIWEK, 2016):

- deve apresentar uma lista sequenciada de passos que levam à solução do problema
- deve ser um processo finito
- deve resolver qualquer instância do problema em questão

Na figura 4 lê-se um conjunto de instruções para somar três números inteiros, extraídos do livro *Practical Arithmetick in Four Books* do autor John Gough, publicado pela primeira vez em 1767. Sob o título 'General rules', lemos

> Coloque os números de modo que cada algarismo possa ficar diretamente abaixo (ou na mesma linha perpendicular) dos algorismos de mesmo valor, ou seja: unidades em unidades, dezenas em dezenas, centenas em centenas.[...]Em seguida, desenhando uma linha abaixo deles; comece a adição no primeiro lugar (ou unidades) e some todas os algorismos naquele lugar, e se a soma delas for menor que dez, coloque-a abaixo da linha abaixo de seu próprio lugar;mas se a soma for superior a dez, estabeleça apenas o excedente acima das dez (ou dezenas) e algumas dezenas, à medida que a soma dessas unidades se eleva, carregue para o local das dezenas, adicionando-as e os algorismos que estão no lugar de

Figura 4 – Algoritmo de soma de três números inteiros



CHAP. II.

ADDITION.

20. ADDITION is the joining or collecting several numbers into one, or finding a number which shall be equal to any given numbers altogether.

GENERAL RULE.

Let the numbers marked A B C, be given to be added.

1. Place the numbers so that each figure may stand directly underneath (or in the same perpendicular row with) the figures of the same value, that is: units under units, tens under tens, hundreds under hundreds, &c.

Then drawing a line under them; begin the Addition at the first place (or units) and add together all the figures in that place, and if their sum be under ten, set it down below the line underneath its own place; but if their sum be more than ten, set down only the overplus above the ten (or tens) and so many tens as the sum of these units amount to, carry to the place of tens, adding them and the figures which stand in the place of tens together; then proceed in the same manner to the third place, or hundreds, and so from place to place to the last, and set down the whole sum of the last place.

| | |
|---|---|
| 54327 | A |
| 8062 | B |
| 5041 | C |
| 67430 | |

*Application*

Fonte: Gough (1813)

> dezenas juntos; em seguida, proceda da mesma maneira até o terceiro lugar, ou centenas, e de um lugar para outro até o último, e estabeleça a soma total do último lugar.

Perceba como essas instruções satisfazem as condições elencadas anteriormente: nela vemos um problema (cálculo da soma de três número inteiros) sendo resolvido de forma encadeada ao longo de um processo finito. Nota-se também que os passos acima facultariam a soma de quaisquer outros três números inteiros, atendendo a exigência de "resolver qualquer instância do problema".

A tarefa de definir as instâncias de um problema equivale a determinar para quais perguntas um algoritmo deve oferecer respostas. Em problema de divisão, por exemplo, onde é preciso calcular a razão entre $a$ e $b$, o par $a = 1.5$ e $b = 9.365$ representa uma delas. Já $a = 65.4$ e $b = 77$ compõe outra. Numa única sentença podemos resumir que qualquer par de reais, onde $b \neq 0$, representa uma instância válida.

A decisão de usar um computador para a resolução de qualquer problema exige que definamos tais instâncias. Sob o ponto de vista da ciência computação, ao procedermos dessa forma, distinguindo claramente os contornos do problema, definimos um "problema computacional".

A definição clara das instâncias do problema (criação de um problema computacional) e dos passos exatos para sua solução[1] (elaboração de um algoritmo) nos permitirá automatizar a sua execução. E é exatamente nessas duas habilidades que se apoiam o pensamento computacional.

## 2.2 Abstração

A noção de abstração é outro componente que necessita ser compreendido ao tratarmos do assunto. Nas palavras de Wing (2006),

> A abstração é usada na definição de padrões, generalização de instâncias e parametrização. É usado para deixar um objeto representar muitos. Ele é usado para capturar propriedades essenciais comuns a um conjunto de objetos enquanto oculta distinções irrelevantes entre eles (WING, 2006, p. 1. Tradução nossa)

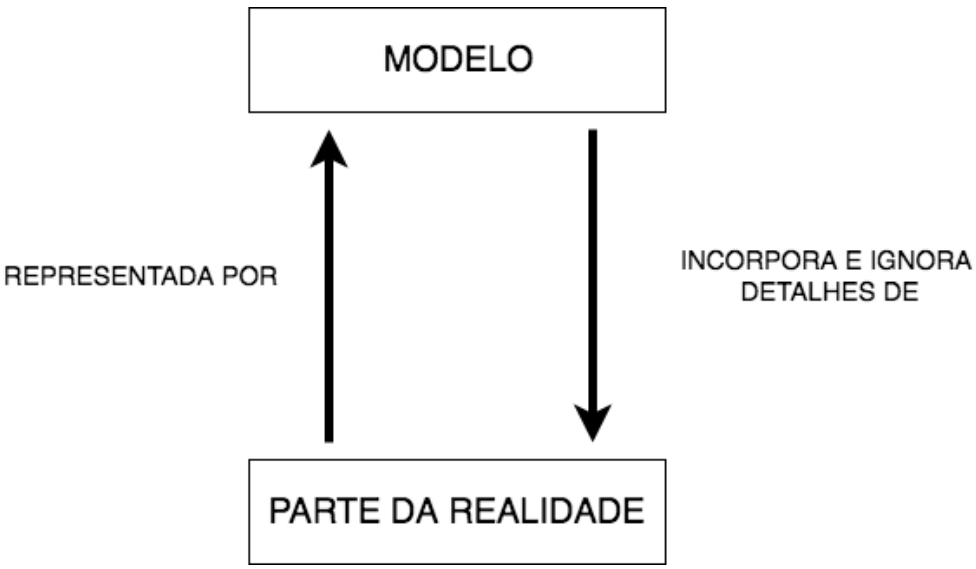
Abstrair, como enfatiza o trecho acima, equivale simplesmente a conduzir um processo de generalização, onde incorporamos parte dos detalhes da realidade em um modelo, ao mesmo tempo que descartamos outros. Estabelece-se assim uma relação entre dois níveis: a realidade e sua representação.

A figura 6 ilustra essa discussão. Nela vemos a obra "A Traição das Imagens " por René Magritte (1898-1967), onde lê-se *"Ceci n'est pas une pipe"* ("Isto não é um cachimbo"). Temos aí provocação que evidencia que um modelo nada mais é do que uma representação, e não a realidade em si.

O grau de detalhamento do modelo depende das circunstâncias do problema, bem como das necessidades de quem modela. Por exemplo, ao analisarmos a trajetória de lançamento de um foguete, podemos descartar suas dimensões e eliminar possíveis efeitos aerodinâmicos. Essa abordagem é extremamente conveniente para o ensino de primeiras

[1] Ao tratarmos computacionalmente um problema devemos ser capazes também de distinguir quando e porque, eventualmente, ele não tem solução.

Figura 5 – Relação entre a realidade e o seu modelo




Adaptado de Piwek (2016)

Figura 6 – "A Traição das Imagens" por René Magritte, 1929


noções de física básica, mas demasiadamente simplória no contexto da condução de um programa aerospacial.

Na figura 7 temos a ilustração dessa ideia: a mesma "realidade" vaca representada por quatro modelos com diferentes graus de detalhes incorporados.

É interessante notar a proximidade do significado para abstração discutido até aqui com aquele proposto pelo filósofo John Locke, segundo o qual formação de uma abstração corresponde a uma transformação durante a qual "ideias tomadas de seres particulares se tornam representantes gerais de todos do mesmo tipo" (LOCKE, 1690/1979 apud

Figura 7 – Quatro representações de uma vaca por Theo van Doesburg, 1917–1918

Fonte: Piwek (2016)

SENGUPTA et al., 2013, p. 354. Tradução nossa).

## 2.3 Modelos e módulos: duas possibilidade de abstração

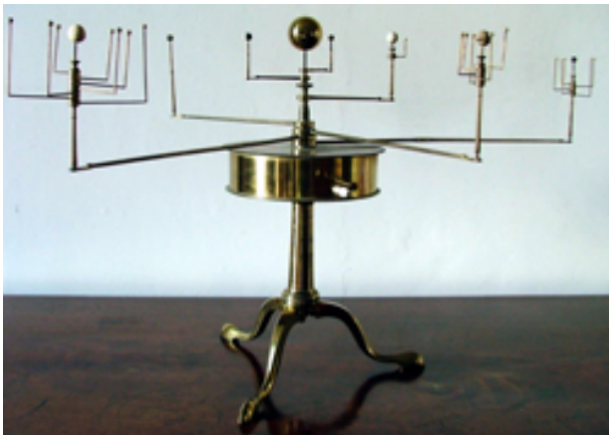
Piwek (2016) distingue dois tipos de abstrações: a "abstração como modelo", onde detalhes da realidade observada são desconsideradas em favor de outras – o mesmo trabalhado até aqui – e a "abstração como encapsulamento", processo no qual organizamos nosso modelo em módulos ou cápsulas que permitem a composição de sistemas mais complexos.

Para diferenciarmos esses dois tipos tomemos como ponto de partida o planetário mecânico ilustrado na figura 8. Nele vemos uma simulação do sistema solar, um modelo, que como tal ignora certas aspectos da realidade. Neste caso em específico temos a desconsideração típica de representações astronômicas onde as proporções entre as dimensões dos planetas e a distância relativas entre eles é extremamente grande. A construção de modelos, ignorando e incorporando detalhes, é o que Piwek (2016) chama de "abstração como modelo".

O mesmo dispositivo incorpora a noção do que o autor chama de "abstração como encapsulamento". Na figura 9 vemos um sistema de rodas dentadas típico do que podemos encontrar no interior de um planetário mecânico. São elas que permitem o movimento das esferas que representam os planetas. A superfície metálica que vemos na figura 8 cumpre o papel de interface do modelo ao escondê-las, deixando externamente visível apenas o que é relevante: o movimento de translação e rotação.

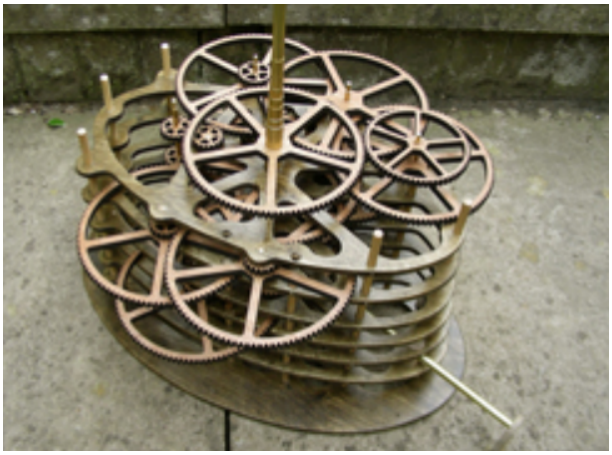
Distingue-se assim a formação de duas camadas durante o processo de encapsulamento: a interface com a qual se pode interagir com o modelo (o invólucro metálico do

Figura 8 – Planetário mecânico



Fonte: Piwek (2016)

Figura 9 – Engrenagem de um planetário mecânico



Fonte: Piwek (2016)

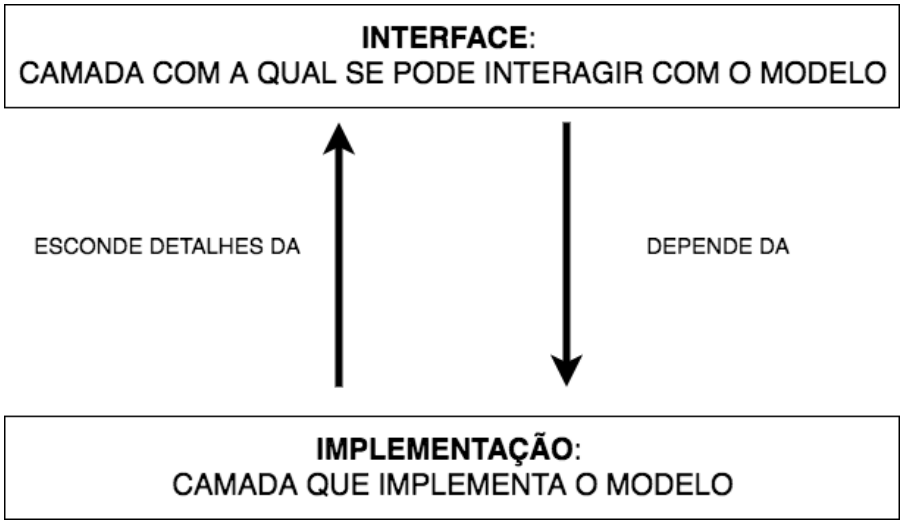
planetário) e a sua implementação propriamente dita (sistema de engrenagens encerradas pela interface). A figura 10 ilustra a relação entre elas.

O uso do encapsulamento é essencial na ciência da computação. Nesse contexto, essa estratégia toma forma em funções e estrutura de dados[2]. Tomemos um exemplo envolvendo funções. Considere a necessidade de calcularmos a seguinte expressão:

$$\frac{(2+3+5\times 7)}{5}$$

[2] Por estrutura de dados nos referimos às diversas possibilidade de organização e relacionamento de dados que uma linguagem oferece. Dentre as mais comuns estão os vetores, as listas, as pilhas...

Figura 10 – Interface e implementação - dois aspectos de um modelo




Fonte: Piwek (2016)

Usando uma linguagem de programação poderíamos reescrever a equação acima do seguinte modo:

$$razao(\boxed{soma(2, 3, \boxed{producto(5, 7)})}, 5)$$

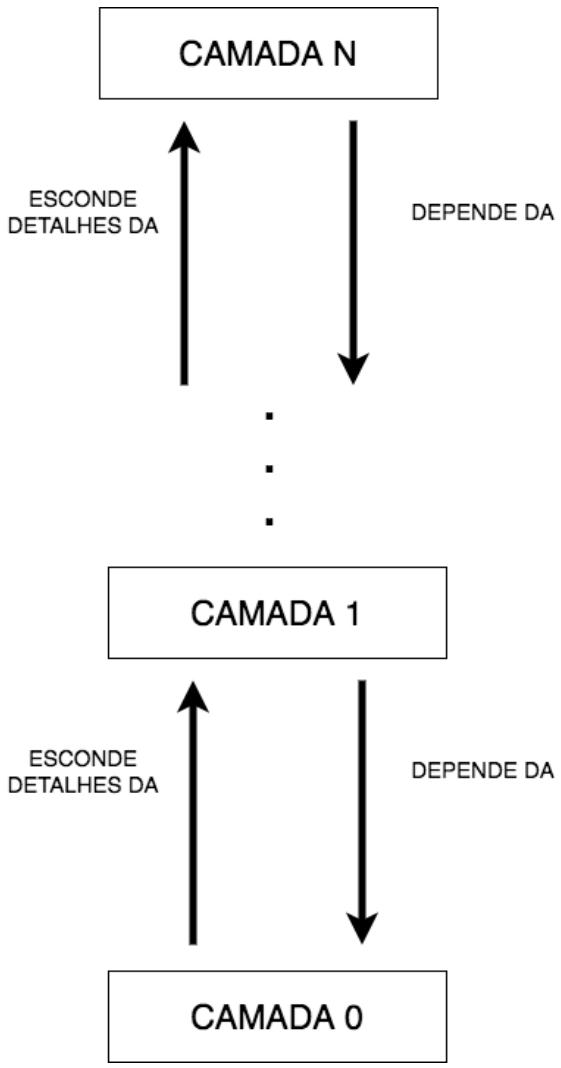
Perceba a formação de cápsulas. Por exemplo, aos escrevermos $produto(a, b, c \ldots)$, estamos modularizando o processo de multiplicação, pois todos os detalhes dessa operação estão invisibilizados atrás de uma interface, cujo nome decidimos chamar como "produto". Aplica-se o mesmo raciocínio aos casos das funções $soma(a, b, c...)$ e $razao(a, b)$.

A decomposição do modelo ou sistema pode ganhar tantas unidades quanto se queira, até atingirmos o nível binário (0 e 1s). É exatamente na possibilidade de aplicação recursiva dessa estratégia que a construção de sistemas complexos se tornam viáveis. Esse aspecto é ilustrado na figura 11.

De fato, ao encapsular, programadores buscam expressar-se apenas em termos de unidades representativas. Essa tática permite a eles raciocinar somente em termos do problema que estão resolvendo, e os protege assim das complexidades próprias do ambiente computacional com que estão trabalhando, tal como a necessidade de administrar diretamente o uso da memória, por exemplo. Ao mesmo tempo, por favorecer a organização e a estruturação lógica do código fonte, como vantagem adicional essa abordagem facilita manutenções futuras do sistema.

Todo o desenvolvimento da computação está assentado na decomposição modular de sistemas. Celulares, computadores, sistemas operacionais, automóveis... Toda e qualquer

Figura 11 – Decomposição em camadas de um sistema

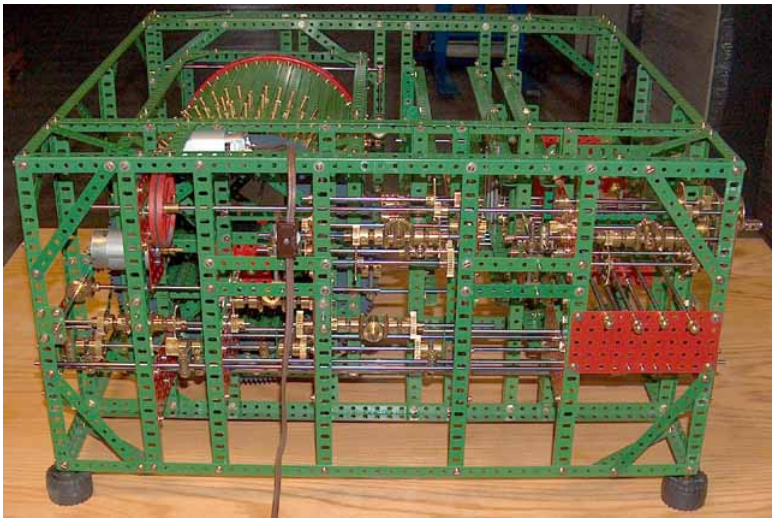



Fonte: Piwek (2016)

tecnologia operada por dispositivos eletrônicos. Há exemplos históricos, contudo, onde esse princípio é aplicado apesar de não haver nenhum substrato eletrônico. Na figura 12 vemos o primeiro computador programável desenvolvido com o propósito de executar operações matemáticas. Conhecido como Meccano, ele foi proposto por Charles Babbage (1791–1871). Movido a vapor, na imagem temos um exemplar construído em latão e ferro.

Em suma, a criação de abstrações tanto por modelagem quanto por encapsulamento nos permitem administrar as complexidades do mundo real. Criando modelos, buscamos diminuir os aspectos ruidosos do mundo real em favor de elementos relevantes para a análise do problema, enquanto que ao encapsularmos, como diz Piwek (2016), evitamos ser vitimizados pelos intricados detalhes do "mundo dos computadores".

Figura 12 – Meccano – Implementação do primeiro computador programável proposto por Charles Babbage



Fonte: Piwek (2016)

## 2.4 Um apanhado geral

Até agora apresentamos alguns conceitos sem relacioná-los apropriadamente. Alguns deles:

- modelo matemático
- problema computacional
- algoritmo
- abstração como modelo e encapsulamento

Afinal como esses elementos se articulam em torno do conceito de pensamento computacional, tema-central desse trabalho? Nessa seção, essa pergunta será o nosso fio-condutor.

Comecemos relacionando os conceitos de modelo matemático e problema computacional.

O propósito da construção de modelos matemáticos se confunde com o da atividade científica: entender e descrever fenômenos e o comportamento de sistemas em função de condições pré-estabelecidas.

Na construção de tais modelos, busca-se investigar as regras de interação entre os componentes internos ao sistema investigado e como o ambiente externo influencia essas

relações. Fixados alguns parametros, tenta-se entender qual é o papel da variação de alguns outros. Em muitos casos, o objetivo principal desse empreendimento é a formalização dessa descrição em equações matemáticas. Em outros, a validação de descrições já propostas.

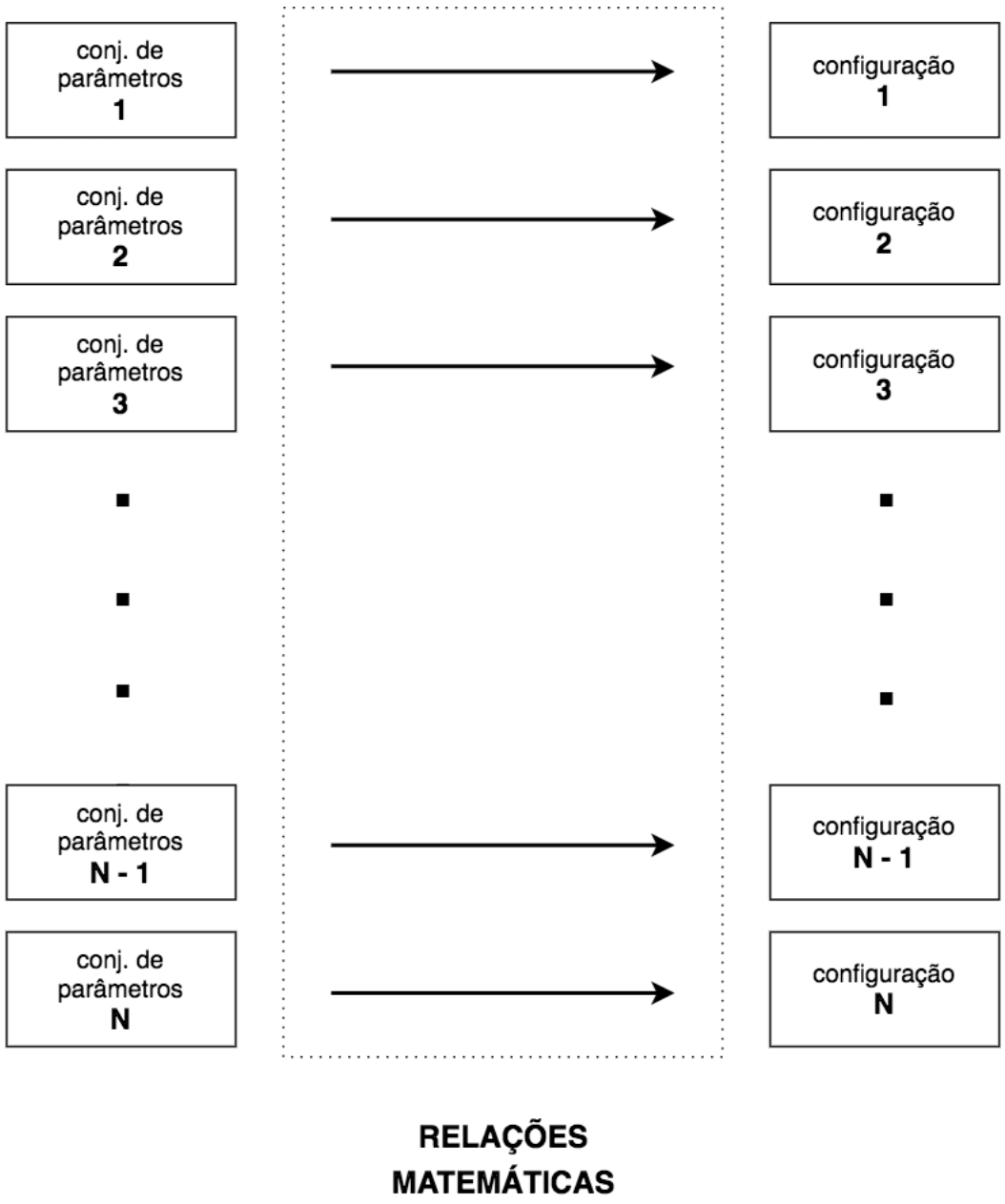
O surgimento do modelo matemático nasce portanto da consolidação das equações que regem o sistema.

Tanto na engenharia quanto nas ciências naturais, o uso de modelos matemáticos tem visado a construção de simulação computacionais. A adoção dessa estratégia favorece o entendimento do comportamento do sistema sob condições limites o que pode evitar, em alguns casos, a ocorrência de tragédias. Ao mesmo tempo, por oferecer um contexto que proporciona o ganho de *insights*, o uso de simulação facilita o desenho de estratégias para otimização de recursos materiais e humanos.

A figura 13 torna explícita a proximidade entre a construção de um modelo matemático e a resolução de um problema computacional. Como definido na seção 2.1, entende-se como problema computacional um conjunto de perguntas que um computador potencialmente responder. A tarefa de resolver o problema computacional, como visto, se resumirá a construção de um algoritmo que leve a cada uma dessas respostas. Igualmente, as equações que compõe um modelo matemático tem como papel tecer uma associação entre um conjunto de parâmetros e uma configuração específica assumida pelo sistema sob observação. Nesse sentido, relações matemáticas e algoritmos cumprem a mesma finalidade.

Essa forte correlação estrutural é o que tem tornado efetiva a relação entre a ciência da computação e a pesquisa científica. Os elos que integram esses domínios estão visíveis na figura 14.

Figura 13 – Elementos de um modelos matemático



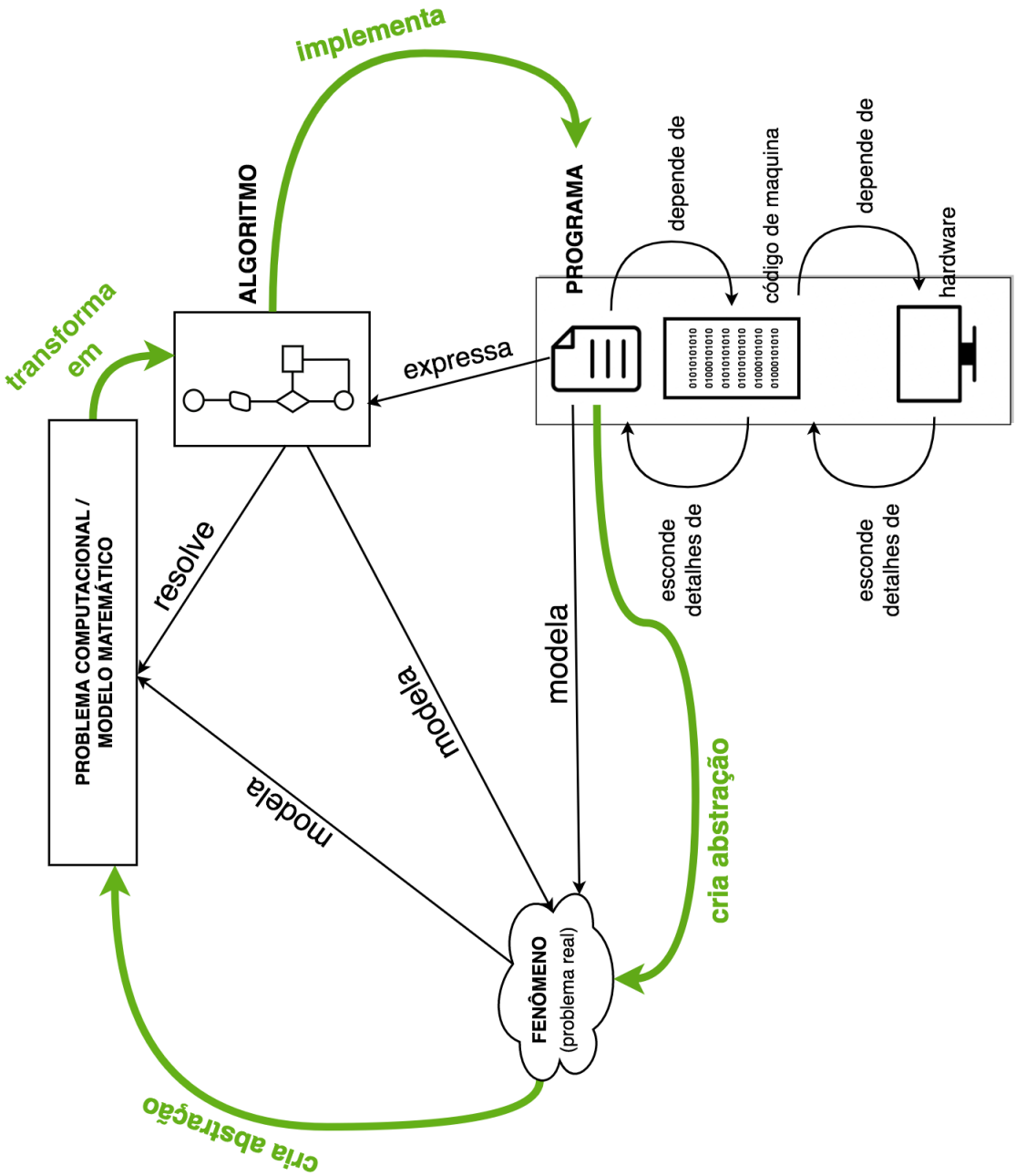



Figura 14 – Visão panorâmica do pensamento computacional

# 3 Implicações para aprendizagem científica

O desenvolvimento da computação e o amadurecimento da noção de pensamento computacional traz consigo várias implicações e oportunidades educacionais.

De partida, pode-se dizer que a efetivação do uso do computador como ferramenta para resolução de problemas exige a formação de capital humano. Normalmente, espera-se que estudantes tenham acesso à universidade para introdução das primeiras noções de programação e ciência da computação. Sabe-se contudo que eles se mostram cada vez familiarizados com *smartphones* e outros dispositivo computacionais. Por isso podemos nos perguntar: por que não antecipar essa instrumentalização?

Esse questionamento se torna ainda mais pertinente em face do percentual reduzido de portadores de grau universitário. A demanda por profissionais com experiência na resolução de problemas computacionais tem crescido numa proporção superior a que academia tem formado. Esse desequilíbrio tem levado grandes empresas de tecnologia como o Google e IBM a dispensar a formação universitária como pre-requisito para contratação (PURTILL, 2018).

A motivação para essa antecipação não é apenas de caráter laboral. Como discutiremos ao longo desse capítulo o uso de computadores e do pensamento computacional oferece excelentes contextos para desenvolvimento de competências científicas. O reverso também é verdadeiro: problemas científicos e matemáticos são domínios excelentes para a aplicação e exercício do pensamento computacional. É nessa tese que esse trabalho se apoia.

No capítulo 2 mostramos o como conceitos de abstração, decomposição de problemas e simulação, aliados a praticas de representação de problemas se estruturam em torno do pensamento computacional. Ao mesmo tempo que são centrais na ciência do computação, estes elementos também são fundamentais para o desenvolvimento de modelos e para a compreensão e resolução de problemas num largo espectro de disciplinas matemáticas e científicas (SENGUPTA et al., 2013).

A literatura tem demonstrado de diversas formas o como a aprendizagem de programação - uma das praticas de representação - em conjunto com conceitos de outros domínios pode ser mais fácil do que aprender cada um desses tópicos individualmente. Essa abordagem é destacadamente diferente da forma como o ensino de computação tem sido trabalhado no ambiente escolar, onde as atividades propostas focam em aspectos apartados da realidade.

Em um estudo citado por Weintrop et al. (2016), descobriu-se que a introdução

de programação no contexto da modelagem de fenômenos físicos e químicos, durante as atividades de alunos de graduação, resultaram no aprendizado mais efetivo de programação, e no aumento do engajamento com a área de domínio das tarefas.

A literatura também nos apresenta outros exemplos de contextualização de atividades computacionais com o ensino de ciências. Dentre algumas propostas práticas, podemos destacar o uso eficiente do software Netlogo (WILENSKY, 1999) na introdução de noções de probabilidade e estatística, como relatado por Abrahamson e Wilensky (2005), e na modelagem de fenômenos epidemiológicos, tal como reportado em (LEE et al., 2011).

Vários estudos apontam que o enriquecimento trazido por essa abordagem conjunta, associada com o caráter "mão na massa" de muitas dessas atividades, produz impactos positivos na forma como os alunos percebem as aulas de ciências (LEE et al., 2011; BARR; STEPHENSON, 2011). A sensação de estar resolvendo um problema autêntico com real aplicabilidade e, em muitos casos, inserido na realidade do qual fazem parte, produz aumentos sensíveis dos níveis de engajamento.

Weintrop et al. (2016) sugere que o aumento da atratividade dessas disciplinas, apresentadas dessa forma, também pode favorecer o aumento da representação de grupos minoritários nos meios científicos, tais como mulheres, negros e transgêneros.

A atenção crescente que educadores têm dado ao tema do pensamento computacional tem sido apoiado também pela expectativa de inverter a relação entre os estudante e a tecnologia. A pergunta que se coloca é: como de meros usuários eles podem vir a tornar criadores?

Resnick (2012) argumenta que apesar de serem vistos como "nativos digitais"[1], a geração nascida durante esse século esta familiarizada apenas com o consumo de tecnologia, e não necessariamente com a sua criação ou produção. Para ele, o beneficio de dotá-la com essa capacidade criativa não estará apenas nessa ou naquela tecnologia desenvolvida pelo estudante, mas sim nas competências cognitivas adquiridas por ele durante o processo de criação. Dentre algumas delas, o autor destaca:

1. a capacidade de pensar em termos sistêmicos, ou seja, em função do comportamento agregado resultante da interação dos componentes de sistema - "o todo não é a soma das partes";
2. a habilidade para trabalhar em equipe;
3. a competência para estruturar e planejar as etapas de um projeto;
4. a facilidade para experimentar novas ideas (prototipagem);

[1] A expressão "nativo digital" foi cunhada pelo escritor Marc Prensky para designar aqueles que desde o seu nascimento tiveram a oportunidade de interagir com tecnologias digitais, tais como videogames, computadores, telefone celular, iPhones, iPads...

5. a destreza para decompor ideias complexas em unidades menores (abstrações);
6. a perícia necessária para investigar defeitos ou comportamentos inesperados (depuração);
7. a inteligência emocional necessária para lidar com frustrações e perseverar em face de dificuldades que surgem ao longo de um projeto.

Como ressalta, mesmo que o estudante não venha a se tornar um engenheiro ou um cientista da computação todas essas competências são importantes para todo e qualquer tipo de atividades. Analogamente, mesmo que poucos deles se profissionalizem como escritores, a capacidade para redigir textos é fundamental para qualquer um.

Resnick (2012) direciona o seu argumento para justificar os benefícios do ensino de programação. Veremos, contudo, que o desenvolvimento dessas competências criativas - todas elas associadas ao pensamento computacional - podem passar por caminhos diferentes da habilidade de escrever código, propriamente dita.

Outras oportunidades educacionais estão associadas às respostas que essa abordagem oferece para vários problemas relacionados ao tema da avaliação. O ensino de ciências é dominado por uma tradição explicativa cujo objetivo é a simples transmissão de dados e conceitos pelo professor. Nesse contexto, convencionou-se que o ciclo de ensino e aprendizagem se fecha quando os alunos são capazes de reproduzir essas informações numa prova cronometrada. No ensino de física, em especial, entende-se que o aprendizado se efetiva quando eles se mostram capazes de resolver três ou quatro variações de um problema discutido em sala de aula - quase sempre de natureza algébrica.

O que se observa, portanto, é um modelo avaliativo indutor de um comportamento autômato do estudante, em prejuízo do desenvolvimento da sua capacidade crítica e analítica.

Por outro lado, o desenvolvimento de atividades sustentadas pelo pensamento computacional tem como foco a resolução de problemas abertos, quase sempre relacionados à construção e não à imposição de modelos (abstrações). Conceitos e princípios científicos são trabalhados sob a perspectiva de que não há modelos “corretos”, mas sim “apropriados”, que melhor respondem a determinadas questões.

Dessa forma, a absorção de tais conteúdos conceituais se dá pela necessidade de articulá-los, por exemplo, na construção de uma animação, ou até mesmo de um jogo. Nesse ambiente, o aluno tem a oportunidade de atestar a validade de um princípio físico pela “qualidade” do caminhar do seu personagem, por quão bem o seu comportamento reflete a realidade. O aprendizado das três leis de Newton nesse cenário nascerá da análise, da observação, e não da simples “decoreba”.

Os marcos avaliativos introduzidos por essa abordagem estão ancorados na perspectiva de que a aprendizagem de conteúdos científicos devem ser encarados como um meio, um *playground* para o desenvolvimento de competências cognitivas, e não como um fim em si.

Por esse motivo pode-se afirmar que o uso do pensamento computacional estimula comportamentos em sala de aula mais compatíveis com a natureza da investigação científica. E por ela se desenvolver cada vez mais em bases computacionais, como discutido no capítulo 1, apoiar a sua aprendizagem na construção e na análise de modelos computacionais permite que os alunos tenham uma visão mais realista e coerente com a forma que ela é exercida profissionalmente.

Dadas as motivações para exercício do pensamento computacional, ainda permanecem algumas questões de natureza práticas a serem respondidas e conciliadas para que essa abordagem se viabilize em sala de aula. Mesmo que até aqui tenhamos esboçado algumas definições, elas ainda não foram capazes de oferecer respostas precisas sobre como esse conceito pode ser materializado em sala de aula. Essa é tarefa que desejamos cumprir no capítulo seguinte.

## 3.1 Significado do pensamento computacional no contexto da sala de aula

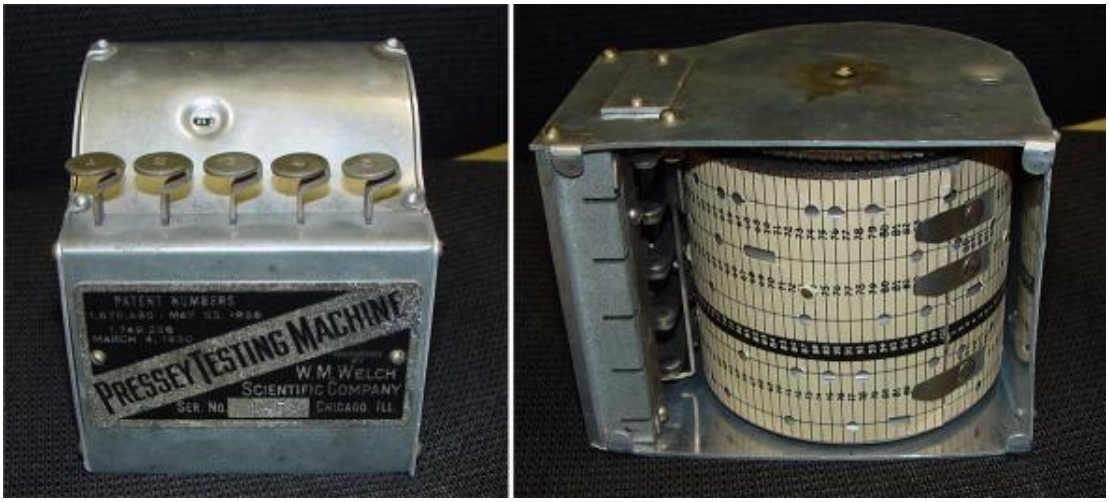
O emprego de dispositivos autômatos como ferramentas educacionais remonta às "máquinas de ensinar", cujo movimento dependia da seleção da alternativa correta de uns dos itens de uma questão múltipla escolha. Esse mecanismo foi proposto e desenvolvido pelo professor de psicologia da Universidade de Ohio, Sidney L. Pressey.

O uso da expressão "pensamento computacional", contudo, tem uma origem mais recente. O seu primeiro registro é observado no livro *Mindstorms: Children, computers and powerful ideas*, da década de 1980, onde Seymour Papert discorre sobre a oportunidade trazida por computadores para o ensino de matemática.

A popularização da expressão e o consequente interesse da comunidade acadêmica veio apenas com a publicação do artigo por Wing (2006), no qual é proposto "um conjunto de atitudes e habilidades universalmente aplicáveis" assentadas na forma como cientistas da computação e programadores resolvem problemas.

Apesar do intenso debate despertado pelo artigo e do reconhecimento das questões e oportunidades educacionais implicadas no tema, o desenvolvimento de atividades e abordagens para sala de aula tem sido especialmente desafiador em face da parcial ou completa inexistência de um significado consensual para o conceito que seja adequado para este ambiente. A definição para o pensamento computacional resultante da discussão

Figura 15 – "Máquina de ensinar" de Pressey




Fonte: Oremus (2015)

proposta por Wing (2006) foca apenas na contribuição da ciência da computação para as várias áreas da atividade humana, e acaba por não demonstrar como esse conceito deve se realizar no plano educacional.

Essa indefinição tem tornado difícil o estabelecimento de formas mensuráveis de avaliar o progresso dos alunos e tem redundado na inviabilização do desenvolvimento de um currículo, e consequentemente, na oferta de treinamento de professores.

Na tentativa de preencher essa lacuna, um número significativo de conferências e congressos tem sido realizado ao redor do mundo, em especial nos Estados Unidos e na Europa, financiados em boa parte por secretarias de educação e grandes empresas de tecnologia, como a Microsoft. Ao mesmo tempo, desde da publicação de Wing (2006), uma quantidade expressiva de artigos reportando experimentações em sala de aula tem sido publicados.

O ponto de convergência desses debates tem sido a busca de uma definição clara sobre o significado desse conceito para o contexto da sala de aula que ao mesmo tempo dialogue com as particularidades e dificuldades pertinentes a esse ambiente.

Dentre as questões para as quais se tem buscado resposta incluem-se (WEINTROP et al., 2016):

1. Como o pensamento computacional se relaciona com as atuais disciplinas e práticas correntes em sala de aula?

2. Como se estabeleceria a progressão de assuntos a serem trabalhados?

3. Quais são as práticas educacionais refletidas pelo pensamento computacional, e como preparar professores para que possam empregá-las em sala de aula?

4. Qual é melhor forma de avaliar o uso e o exercício dessas práticas pelos alunos?

Alguns exemplos concretos da formulação desses e de outros questionamentos podem ser encontrados no relatório produzido pelo Conselho Nacional de Pesquisas dos Estados Unidos (NRC), resultante de um encontro realizado no ano de 2010. Nele são listados vinte habilidades e práticas identificadas com o pensamento computacional, tais como a abstração e decomposição de problemas, formulação de soluções heurísticas e de estratégias de busca, bem como conceitos pertinentes à ciência da computação, tais como processamento paralelo e uso de recursão (NATIONAL RESEARCH COUNCIL, 2010 apud WEINTROP et al., 2016).

Também vale registrar o trabalho de Barr e Stephenson (2011), no qual o esforço de significar o pensamento computacional na perspectiva da sala de aula traduziu-se no mapeamento de algumas competências a disciplinas comuns à educação básica. Esse conteúdo esta disponível nas tabelas 1 e 2.

Outro trabalho merecedor de registro é o de Brennan e Resnick (2012). Nele o pensamento computacional é trabalhado sob a perspectiva da aprendizagem de programação e é estruturado em três dimensões:

1. **conceitos computacionais** - conceitos com os quais os estudantes se envolvem ao aprender a programar, tais como iteração e paralelismo.

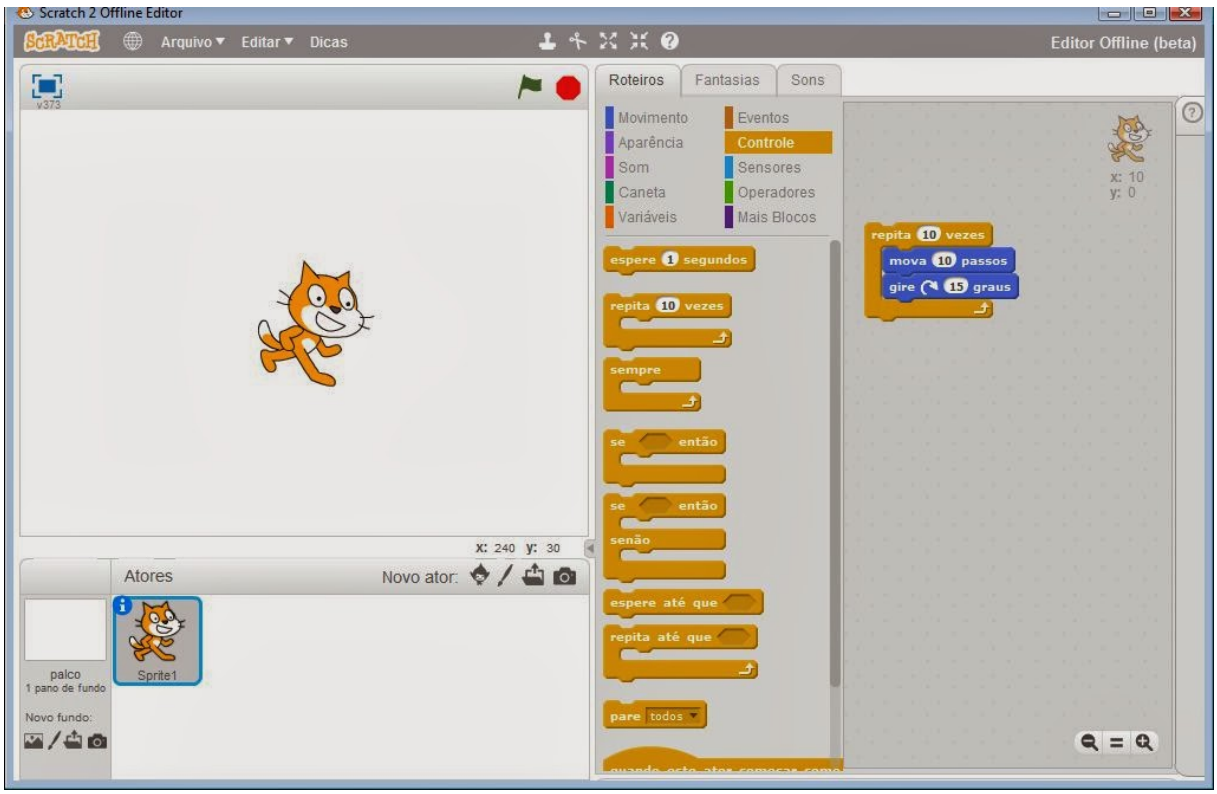
2. **práticas computacionais** - práticas que os estudantes desenvolvem quando se engajam com esses conceitos.

3. **perspectivas computacionais** - perspectivas que os estudantes adquirem sobre o mundo ao seu redor e sobre si mesmo, resultantes desse aprendizado.

Acompanhando essa proposta, são sugeridas formas de avaliar o progresso dos alunos em cada uma delas. O uso do ambiente *Scratch*, ilustrado na figura 16, desenvolvido pelo *MIT Labs*, marca o desenvolvimento do artigo.

Ao longo das páginas seguintes, iremos apoiar as nossas análises na proposição de Weintrop et al. (2016). Entre os trabalhos analisados, esta contribuição é a que melhor responde à nossa necessidade de oferecer uma perspectiva prática do pensamento computacional, coerente com as necessidades do ensino de ciências. Isso porque nele podemos observar a delimitação clara das competências a serem exercitadas pelos alunos e como elas tendem a favorecer a tão almejada aprendizagem científica.

Figura 16 – *Scratch* - linguagem de programação visual desenvolvido pelo *MIT Labs*.




Essa definição é alcançada pela criação de uma taxonomia que contemple as principais práticas identificadas pelo autor que estão relacionadas ao pensamento computacional. Na seção seguinte iremos detalha-la, mas, previamente, descrever a metodologia adotada. E como definições só podem ser úteis se forem acompanhada de exemplos práticos, iremos, em seguida, demonstrar como o exercício dessas competências podem ser incorporadas a temas das aulas de ciências, ao mesmo tempo que ilustramos a discussão com a descrição de casos reais.

## 3.2 A criação de uma taxonomia para o pensamento computacional

A definição de uma taxonomia segue na esteira dos empreendimentos cujo objetivo tem sido uma significação funcional para o pensamento computacional. Nas palavras de Weintrop et al. (2016), o que se busca é

> [...] uma abordagem diferente para a definição do pensamento computacional, contando com a aplicação das práticas identificadas acima em contextos distintos da ciência da computação. Ao fazer isso, deixamos de confiar em ideias e práticas descontextualizadas e, em vez disso, recorremos a instanciações reais do pensamento computacional na natureza para fornecer clareza e especificidade sobre o que o termo significa. Isso reforça o argumento de que essas práticas são amplamente aplicáveis, ao

> mesmo tempo em que fornecem uma definição concreta e acionável do pensamento computacional. (WEINTROP et al., 2016, Tradução nossa)

A metodologia adotada envolveu o emprego de três recursos básicos:

1. amostras de atividades educativas envolvendo pensamento computacional tanto em matemáticas como em ciências.
2. levantamento de conceitos existentes sobre o pensamento computacional presentes na literatura.
3. entrevistas com matemáticos e cientistas.

Os passos dados pelos autores foram assim sequenciados:

1. **Delimitação das rotinas e habilidades relacionadas**

   Com o objetivo de identificar as competências e práticas que são associadas repetidamente ao pensamento computacional, uma revisão da literatura foi feita. Dentre o material analisado estiveram tanto documentações oficiais produzidas sobre pensamento computacional, como o já citado National Research Council (2010), como ementas curriculares de programas de engenharia e ciência da computação. Durante essa investigação, buscou-se a aquisição de uma visão panorâmica sobre o tema antes da delimitação do escopo da análise para as disciplinas matemáticas e científicas. Dessa etapa, dez competências básicas foram extraídas (Tabela 3).

2. **Coleção e classificação de atividades voltadas para a introdução do tema nas aulas de ciências e matemática**

   Foram analisadas trinta planos de aulas de uma coleção, permeadas por conteúdos de física, astronomia, química, biologia, ciências da terra, redes e programação. Todo material analisado tinha origem no programa "Reach for Stars"[2] cujo propósito é a integração entre estudantes de pós-graduação de disciplinas STEM, dependentes do uso de computação nas suas pesquisas, e alunos da educação básica. Tendo como meta a identificação de práticas relacionadas às que foram recortadas do primeiro passo, uma analise foi feita por dois pesquisadores independentes. Dela emergiram um conjunto de 208 facetas do pensamento computacional, das quais uma pequena amostra é exposta na tabela 4.

[2] Mais informações sobre o programa pode ser encontrado em http://gk12.ciera.northwestern.edu/,

Tabela 1 – Práticas identificadas no pensamento computacional, segundo Barr e Stephenson (2011), presentes/possíveis na educação básica

| **Práticas** | **Matemática** | **Ciência** | **Estudos Sociais** | **Linguagens** |
|---|---|---|---|---|
| *Coleta de dados* | Encontrar fonte de dados de um problema, lançando dados ou moedas, por exemplo | Recolhimento de dados de um experimento | Estudar estatísticas de batalha ou outras estatísticas populacionais | Fazer uma análise linguística de sentenças. |
| *Análise de dados* | Contagem da ocorrências de lançamentos de moeda ou dados e analisar resultados | Analisar os dados a partir de um experimentar | Identificar tendências nos dados a partir de estatísticas | Identificar padrões para frase de diferentes tipos. |
| *Representação de dados* | Use histograma, gráfico circular, gráfico de barras... Usar conjuntos, listas, gráficos, etc. | Resumir dados de um experimento | Resumir e representam tendências | Representar padrões de diferentes tipos de sentença |
| *Decomposição de problemas* | Aplicar ordem de operações em uma expressão | Fazer classificação de espécies | - | Escrever um esboço |
| *Abstração* | User variáveis em álgebra; estudar funções algébricas em comparação com as funções na programação | Construir um modelo de uma entidade física | Resumir os fatos; deduzir conclusões a partir de factos | - |

Fonte: Adaptado de Barr e Stephenson (2011)

Tabela 2 – *Continuação* - Práticas identificadas no pensamento computacional, segundo Barr e Stephenson (2011), presentes/possíveis na educação básica

| **Práticas** | **Matemática** | **Ciências** | **Estudos Sociais** | **Linguagens** |
|---|---|---|---|---|
| *Procedimentos algorítmicos* | Longa divisão, fatoração | Fazer procedimentos experimentais | - | Escrever instruções |
| *Automação* | - | Usar *Probware* ou *Origin* | Usar excel | Usar um corretor ortográfico |
| *Paralelização* | Resolver sistema lineares; fazer multiplicação de matrizes | Executar simultaneamente experimentos com diferentes parâmetros | - | - |
| *Simulação* | Representar graficamente uma função em um plano cartesiano e modificar os valores da variáveis | Simular o movimento de o sistema solar | Jogar Age of Empires; Trilha de Oregon | Fazer uma reencenação de uma história |

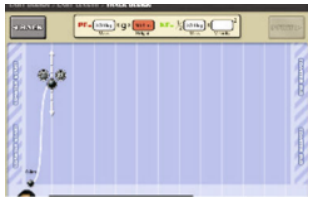
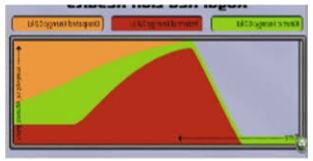
Fonte: Adaptado de Barr e Stephenson (2011)

Tabela 3 – Competências associadas ao pensamento computacional segundo análise da literatura feita por Weintrop et al. (2016)

| | |
|---|---|
| 1. Capacidade para lidar com problemas abertos | 6. Criar abstrações para aspectos do problema em mãos |
| 3. Persistência para trabalhar com problemas desafiadores | 7. Reestruturar um problema em termos de outros conhecidos |
| 3. Confiança para lidar com a complexidade | 8. Avaliar pontos fortes/fracos de uma representação de dados/sistema representacional |
| 3. Representar ideas the forma computacionalmente significativa | 9. Gerar soluções algorítmicas |
| 3. Quebrar problemas grandes em problemas menores | 10. Reconhecer e lidar com a ambiguidade em algoritmos |

Fonte: Weintrop et al. (2016)

Tabela 4 – Facetas do pensamento computacional extraídos de material voltado para implementação do pensamento educacional nas escolas

| Atividade | Faceta | Prática Envolvida | Classificação taxonômica da prática |
|---|---|---|---|
| Projeto de construtor de montanha-russa  | Os alunos constroem modelos de montanhas-russas que podem ser executadas, gerando dados sobre a energia da montanha-russa | Obter insight / compreensão de simulações / modelos baseados em computador | Uso de um modelo computacional para entender um conceito<br><br>Construção de modelos computacionais. |
| | Os alunos registram medições de energia em quatro pontos da pista e armazenam os dados em uma tabela | Fazer medições efetivas a partir de uma simulação | Coleta de dados |
| Desenho de uma montanha-russa  Gráfico das energias cinéticas e potenciais da montanha-russa ao longo do tempo | São necessárias várias iterações para construir uma montanha russa de sucesso que termine a pista e não trave | Abordagem iterativa para a construção de uma solução | Uso de modelos computacionais para encontrar e testar uma solução<br><br>Investigação sistemática das causas de um problema (*troubleshooting*) e depuração (*debugging*) |
| | Traduzindo em gráficos de tela de energia potencial / cinética em forma de tabela | Mensurar a adequação de uma representação avaliando seus pontos fortes e fracos. | Manipulação de dados<br><br>Visualização de dados |

Fonte: Adaptado de Weintrop et al. (2016)

3. **Organização temática**

   A partir da análise feita até então, foi produzida uma lista com vinte e sete práticas, tematicamente organizada em cinco grandes campos: dados e informações, modelagem e simulação, resolução de problemas, pensamento sistêmico.

4. **Validação**

   A taxonomia criada, disponível até a execução dessa etapa, foi apresentada a professores da educação básica. Com esse material em mãos, colaborativamente, eles puderam desenhar atividades para suas turmas.

   No geral, de acordo com o autor, o *feedback* fornecidos por eles foram bastante positivos, embora algumas preocupações tenham sido evidenciadas em relação às seções da taxonomia com conceitos computacionais, tais como aplicação de lógica condicional e uso de recursão e iteração.

   A razão para esse receio era a percepção que tais tópicos estavam muito distantes do conteúdo trabalhado em sala de aula.

   Além de profissionais da educação básica, outros tantos *feedbacks* foram colhidos com pesquisadores de alguns setores da indústria, bioquímicos, físicos, engenheiro de materiais, astrofísicos, cientistas da computação e engenheiros biomédicos.

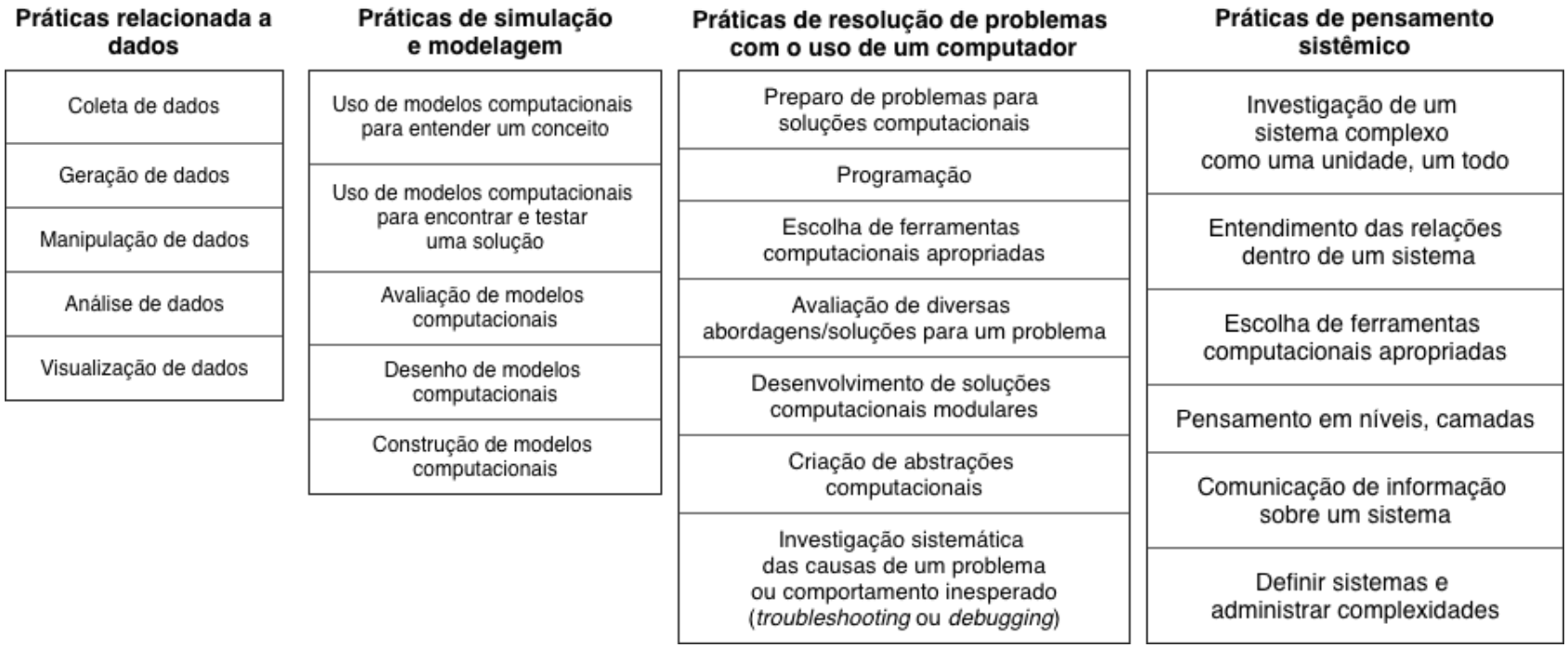
   Além da intenção de colher avaliações, o objetivo dessa rodada de entrevistas foi a obtenção de informações e *insights* sobre o como o pensamento computacional integra autênticos ambientes de pesquisas. Ao mesmo tempo, buscou-se observar as semelhanças e diferenças na forma como o uso computação se efetiva na rotina de trabalho em cada uma das áreas desses profissionais.

5. **Revisão**

   A partir dos *feedbacks* e sugestões feita pelos professores e pesquisadores, uma revisão da primeira versão da taxonomia foi feita.

   O resultado final desse processo pode ser visto na tabela disponível na figura 17. Uma discussão sobre cada um dos elementos dessa classificação é feita a seguir.

Figura 17 – Taxonomia para práticas do pensamento computacional




Fonte: Weintrop et al. (2016)

### 3.2.1 Práticas relacionadas a dados

Weintrop et al. (2016) sustenta que a capacidade de analisar e extrair significado de um grande volume de dados desestruturados tem sido uma das características definidoras da prática científica moderna. A maneira como esse empreendimento vem sendo conduzido tem sido impactados significativamente pelos avanços tecnológicos em curso.

Para ilustrar a relevância dessa prática, o autor destaca uma das entrevistas feitas durante a elaboração da taxonomia na qual um cientista de materiais, questionado sobre como era sua pesquisa, responde: "Em quase tudo, o que se tem são dados brutos". Em seguida ele prossegue explicando como ele define as perguntas a serem respondidas (problemas de pesquisa), e como dados são gerados, processados e organizados, para que então, finalmente, o uso de determinados softwares gerem as visualizações para suas conclusões.

Segundo o autor, abrigar algumas dessa práticas nas atividades escolares deve significar ajudar o alunos a perceberem que inerentemente dados não oferecem respostas imediatas, e que para alcançá-las "ordem deve ser imposta". Por esse motivo a introdução de noções básicas de estatística e probabilidade se faz necessária.

Cada um dos aspectos relevantes ao uso de dados são destrinchados a seguir.

1. **Coleta de dados**

   O papel que os computadores cumprem na coleta de dados guarda relação com a necessidade de automatizar protocolos de registro e armazenamento de dados gerados pela observação de um fenômeno.

   De acordo com Weintrop et al. (2016), o aluno que dominar esta prática será capaz de propor protocolos de coleta dados e formas de uso de ferramentas computacionais que automatizam essa tarefa.

2. **Geração de dados**

   Em astronomia, para a compreensão da evolução de uma galáxia, torna-se necessário a geração de dados a partir de simulações computacionais, dado que escala de tempo investigada (bilhões de anos) é incompatível com a dos pesquisadores.

   Esse exemplo ilustra um dos aspectos da pesquisa científica onde a necessidade de geração de dados impõe-se em razão de restrições circunstanciais.

   Como assinala o autor, o estudante que dominar esta prática será capaz de definir procedimentos computacionais e executar simulações que permitam a criação de dados, cuja utilização pode auxiliá-lo a aprofundar a compreensão em um tópico sob investigação.

3. **Manipulação de dados**

O uso de computadores torna possíveis a manipulação de dados desestruturados, de tal modo a permitir que significados possam ser extraídos.

Dentre as várias técnicas existentes de manipulação, o autor cita o ordenamento, a filtragem, a limpeza, a normalização e a união de conjunto de dados dissociados.

Como destaca, o aluno que dominar esta prática será capaz de, a partir do uso de ferramentas computacionais, remodelar um conjunto de dados de tal modo a facilitar seu trabalho de investigação.

4. **Análise de dados**

   Em face da abundância de dados existentes, o uso de ferramentas computacionais para análise de dados tem sido um das tarefas mais relevantes da prática científica.

   Dentre as várias estratégias existentes, o autor destaca a busca por padrões ou anomalias, a definição de regras para categorização de dados, e a identificação de tendências e correlações.

   Conforme sublinha, o aluno que dominar esta prática será capaz de fazer afirmações e extrair conclusões a partir da análise de um conjunto de dados.

5. **Visualização de dados**

   Outros aspectos relevantes e necessários da atividade científica são a comunicação e o compartilhamento de dados.

   Nesse sentido, o uso de ferramentas computacionais que tornam possível a projeção de dados, seja na forma de um gráfico simples ou de uma exposição dinâmica que permite a interação do usuário com a informação, desempenha papel preponderante.

   Conforme sustenta Weintrop et al. (2016), o aluno que dominar esta prática será capaz de usar com eficiência ferramentas de visualização que permitam a comunicação de informações e conclusões extraídas durante a análise.

### 3.2.2 Práticas de simulação e modelagem

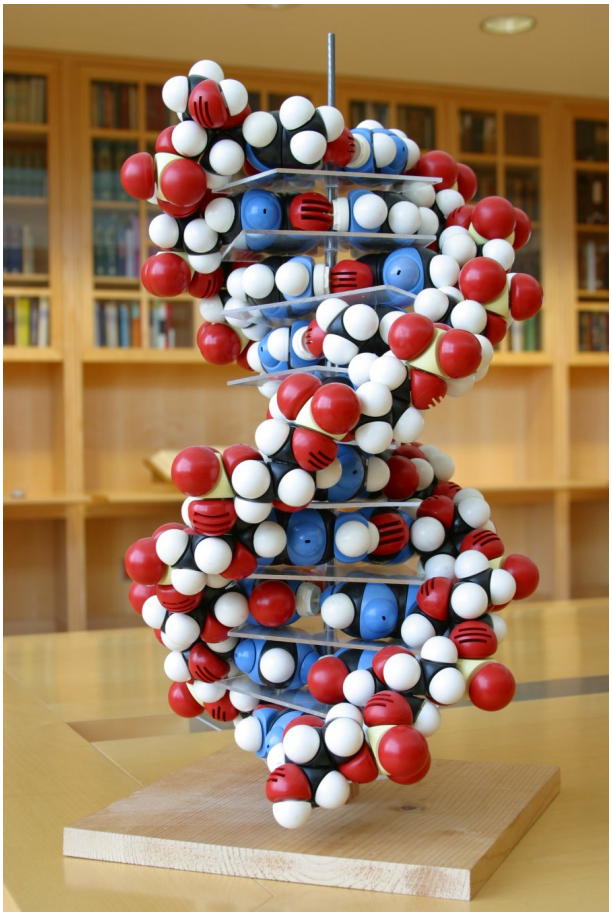
Outro conjunto práticas que compõe a taxonomia de Weintrop et al. (2016) são aquelas relacionadas ao exercício da representação de fenômenos, também presentes no uso e criação de modelos e simulacros computacionais.

Falando especificamente de modelos, poderíamos dizer que “todos estão errados, mas alguns deles são úteis” (BOX; DRAPER, 1987 apud WEINTROP et al., 2016), dado o caráter simplificador, mas representativo de alguns - proporcional à capacidade descritiva e preditiva que comportam.

Dentre as várias formas que um modelo pode assumir, o autor destaca os fluxogramas, os diagramas, as equações, as fórmulas químicas, os modelos físicos (como maquetes

ou representações tridimensionais de uma molécula), e por fim os modelos e simulações computacionais - representações não-estáticas de um fenômeno, reproduzidos por um computador (WEINTROP et al., 2016).

Figura 18 – Modelo físico de uma cadeia de DNA



O trabalho de criação de um modelo computacional envolve a identificação de abstrações apropriadas - regras matemáticas subjacentes ao fenômeno modelado, por exemplo - e seu refino iterativo, por meio de depuração (investigação e correção de comportamentos inesperados) e validação com elementos correspondentes no mundo real (WING, 2008).

De um ponto de vista pedagógico, a sua utilidade está na possibilidade de tornar a compreensão de um fenômeno mais clara e permitir a investigação de comportamentos cuja condições determinantes são difíceis, senão impossíveis de reproduzir.

O poder didático dessa ferramenta, contudo, não está apenas no seu uso, mas na possibilidade do estudante também criá-la, tornando possível a formulação de suas próprias interpretações da realidade.

Como ressalta Weintrop et al. (2016), os educadores reconhecem que tais práticas

de representação são indissociáveis da aprendizagem científica, mas elas são raremente explicitadas como parte da instrução. E é exatamente para contornar esse problema que novas formulações curriculares tem tornado claro a necessidade de estimular a criação, e não apenas o uso de modelos acabados.

Abaixo listamos as cinco práticas incluídas na taxonomia, referentes à simulação e modelagem.

1. **Uso de modelos computacionais para entender um conceito**

   Segundo o autor, o uso de modelos computacionais que demonstram ideias específicas ou simulam comportamento de fenômenos são poderosas ferramentas didáticas.

   Isso porque eles tendem a oferecer suporte a investigações sistemáticas que permitem o exercício do controle sobre parâmetros de comportamento, através de combinações e variações que, normalmente, são difíceis, senão impossíveis, de reproduzir sem o aparato computacional.

   Por isso, conclui que o estudante que dominar essa prática será capaz de aprofundar sua compreensão sobre um fenômeno ou conceito sob investigação, utilizando modelos computacionais que os reproduzem.

2. **Uso de modelos computacionais para encontrar e testar uma solução**

   Modelos computacionais podem ser usados para testar hipóteses e descobrir soluções para problemas.

   Como discutido anteriormente, eles viabilizam, por exemplo, a exploração de aspectos de fenômenos que seriam muito perigosos, caros, difíceis, ou simplesmente impossíveis de reproduzir em escala ordinária de tempo e de recursos.

   Entretanto, com poder computacional e modelos matemáticos adequados pode-se facilmente alcançar uma boa diversificação e combinação do "espaços de parâmetros" [3], e assim validar ou testar os limites de uma solução.

   Weintrop et al. (2016), sugere que o aluno que dominar esta prática será capaz de propor, testar e assim justificar o uso de uma solução, a partir da exploração de um modelo computacional, e aplicar adequadamente as informações extraídas nesse processo.

3. **Avaliação de modelos computacionais**

   A capacidade de avaliar a relação de um modelo computacional com a realidade representada é um dos aspectos aspectos essenciais para medirmos a sua adequabilidade, segundo Weintrop et al. (2016).

[3] conjunto de todos os arranjos possíveis dos parâmetros de um modelo matemático

O autor aponta algumas questões que precisam ser analisadas e respondidas quando se tem esse fim em mente. Dentre elas estão as seguintes:

a) Quais suposições que os criadores do modelo fizeram sobre o mundo e como elas afetam o seu comportamento?

b) Quais as camadas de abstração foram incorporadas ao modelo e como elas moldam a fidelidade da sua representação do fenômeno?

c) Quais são os aspectos do modelo que foram fielmente modelados e quais outros foram simplificados ou simplesmente ignorados?

Responder essas questões é tarefa essencial para compreensão dos limites de uma representação. Ao mesmo tempo, tê-las em contas é fundamental para que se possa validá-la ou não.

De acordo com o autor, o aluno que dominar esta prática "será capaz de articular as diferenças entre o modelo e fenômeno representado, o que inclui o levantamento de ameças a sua validade e a identificação das suposições que o sustentam" (WEINTROP et al., 2016).

4. **Desenho de modelos computacionais**

   O desenho de modelos computacionais podem ser norteados por algumas motivações. Dentre as já discutidas, podemos citar a necessidade de testar hipóteses ou comunicar ideas, e o interesse em compreender melhor um fenômeno,

   Durante esse processo somos levados à tomada de algumas decisões, que podem variar de acordo com o problema a ser resolvido. Dentre as mais recorrentes, Weintrop et al. (2016) cita a necessidade de definir os limites do sistema - explicitar aquilo que deve ser incorporado e/ou descartado - e conceitualizar as propriedades e comportamentos dos elementos incluídos. Ao final, deve ser assegurado que o produto resultante atenda aos objetivos que inicialmente motivaram a construção do modelo (WEINTROP et al., 2016).

   Portanto, os estudantes que exercitarem essa prática terão a oportunidade de desenhar modelos computacionais, descrevendo-os e decidindo quais dados serão produzidos, enquanto poderão cultivar o hábito de fazer avaliações críticas, a partir da análise dos limites da representação, das suposições nela incorporadas. Espera-se que essa reflexão faculte a eles o entendimento da extensão das conclusões que podem ser extraídas do modelo.

5. **Construção de modelos computacionais**

   Uma importante ferramenta da atividade de um pesquisador é sua capacidade de criar modelos computacionais ou ampliar outros já existentes, seja dando continuidade ou simplesmente complementando o trabalho realizado por seus pares.

Enquanto o desenho de um modelo, como já discutido, envolve a tomada de decisões essencialmente teóricas - tais como a seleção e descarte de elementos, bem como a descrição e conceptualização de comportamentos e propriedades - a sua construção, por outro lado, exige saber expressá-lo de forma "compreensível" a um computador.

Em muitos casos essa competência se expressa com o uso de uma linguagem de programação, mas em outros com *softwares* que provêm uma interface gráfica ou de linha de comando, tais como o *Wolfram Mathematica*[4] e o *Matlab*[5].

Weintrop et al. (2016), sugere que estudantes que dominarem essa prática serão capazes de implementar novos comportamento de modelos, seja por criação ou extensão, utilizando tanto uma linguagem de programação quanto *softwares* prontos, que oferecem suporte à modelagem.

### 3.2.3 Prática de resolução de problemas computacionais

Nesta etapa de elaboração da taxonomia Weintrop et al. (2016), concentra seus esforços na apresentação das práticas de resolução de problemas trazidas pela ciência da computação. Seguem destrinchadas cada uma delas.

1. **Preparo de problemas para admitir soluções computacionais**

   Normalmente, para que um problema admita solução computacional, faz-se necessário a sua reformulação de modo a mapeá-lo às capacidades de ferramentas existentes - sejam elas pacotes de software ou dispositivos físicos.

   Dentre as várias estratégias existentes que possibilitam esse reenquadramento, Weintrop et al. (2016) lista a possibilidade de decomposição de um problema em unidades menores, ou simplesmente a sua reedição em termos de outros, cujas soluções são conhecidas.

   O autor sugere que o estudante que dominar esta prática será capaz de empregar estratégias de reenquadramento e tornar mais eficiente a resolução de problemas nos quais estiver trabalhando.

2. **Programação**

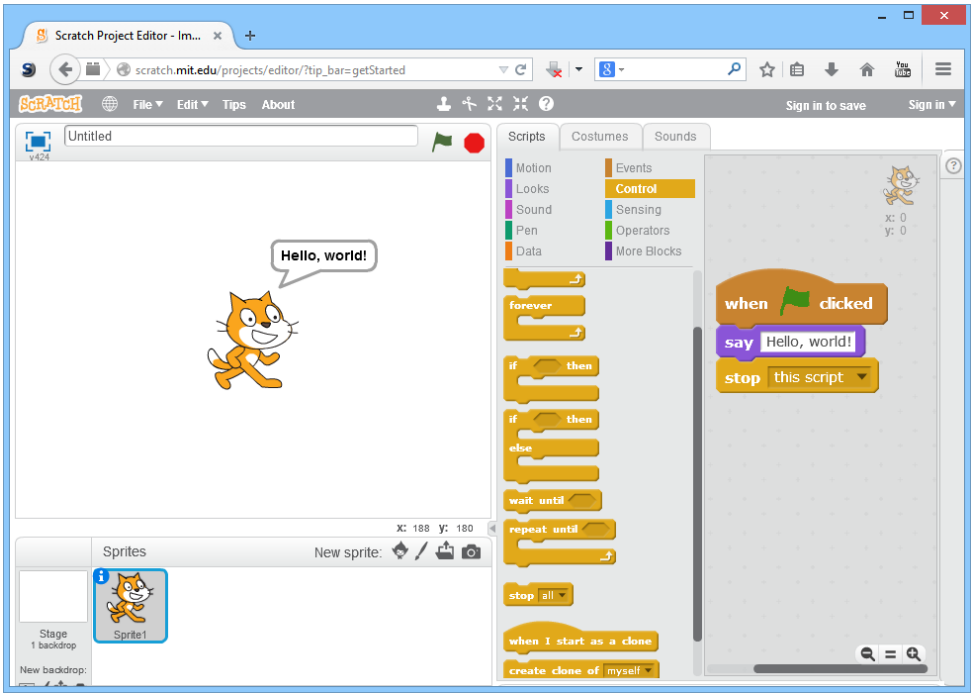
   O emprego da codificação tem se mostrado um poderosa ferramenta na resolução de problemas. Dentre as vários possibilidades que essa habilidade tem trazido para pesquisa científica, pode-se destacar a coleta e análise massiva de dados, a formatação e visualização de informações, bem como a construção e extensão de modelos e interfaces com ferramentas computacionais existentes (WEINTROP et al., 2016).

[4] http://www.wolfram.com/mathematica/
[5] https://www.mathworks.com/products/matlab.html

O ensino de programação, contudo, tem se mostrado um tarefa desafiadora em face da pouca ou nenhuma disponibilidade de programas de capacitação para professores e da dificuldade de aprendizado relacionada à sintaxe das linguagens de programação convencionais.

Para contornar estas dificuldades, tem sido desenvolvidas linguagens e plataformas de programação visual, cujo arranjo algorítmico e sintático das instruções dadas são estruturados por elementos gráficos e não textuais. Dos exemplos com esta finalidade, o mais proeminente é o *Scratch*, sobre o qual já falamos na seção 3.1. Mais detalhes a respeito e registros de experiências didáticas envolvendo o programa podem ser encontradas em Maloney et al. (2008) e Resnick (2012).

Figura 19 – Scratch - linguagem e plataforma de programação visual




Quanto às competências a serem exercitadas pelo estudante com o aprendizado de programação, de acordo com a taxonomia de Weintrop et al. (2016), espera-se que eles se tornem capazes de entender, modificar e criar programas e, com essas habilidades, persigam seus "próprios objetivos matemáticos e científicos".

3. **Escolha de ferramentas computacionais apropriadas**

   Um único problema pode ser resolvido de diversas formas e com o auxílio de diversas ferramentas. A destreza para identificar dentre um conjunto aquelas que melhor auxiliam nesse propósito é uma das principais competências necessárias em um projeto.

Na lista parâmetros que normalmente influenciam essa escolha incluem-se as funcionalidades oferecidas pelo software, as suas possibilidades de personalização, os tipos de dados que podem ser inseridos e produzidos, e para além do campo técnico, por exemplo, a maturidade e grau de atividade da sua comunidade de usuários (WEINTROP et al., 2016). Esse espaço de troca de informação é especialmente importante em situações em que a assistência se faz necessária mas a documentação disponível pelos desenvolvedores da ferramenta são insuficientes.

Weintrop et al. (2016) sugere que os alunos que adquirem essa *expertise* serão capazes de balancear as vantagens e desvantagens oferecidas por ferramenta computacional e, assim, tomar decisões justificáveis e informadas.

4. **Avaliação de diversas abordagens/soluções para resolver um problema**

   Uma outra faceta do processo de resolução de problemas, recorrente na prática científica, é necessidade de escolher uma solução dentre muitas possíveis.

   Quando há duas ou mais abordagens para um problema, faz-se necessário a avaliação da adequação de cada uma delas em termos de parâmetros tais como extensibilidade, durabilidade, flexibilidade, reusabilidade e custo, seja em recursos materiais ou em tempo.

   Weintrop et al. (2016) avalia que alunos que exercitarem essa prática poderão avaliar diversas abordagens ou soluções para um problema, tendo como referência de julgamento as restrições e requisitos do problema, bem como as funcionalidades das ferramentas disponíveis.

5. **Desenvolvimento de soluções computacionais modulares**

   Ordinariamente, a resolução de um problema complexo sequencia-se em um conjunto de etapas menores e assume forma na solução de outros menores.

   Cada uma dessas soluções podem ser desenhadas de modo intrincado, visando apenas o problema em questão, ou podem ser concebidas modularmente, de modo a serem reutilizáveis.

   Quando modulares, os elementos de um modelo ou ferramenta computacional podem ser desenvolvidos de maneira incremental e serem testados e depurados de maneira independente, aspecto que facilita sensivelmente sua manutenção.

   Quanto ao aspecto escolar, Weintrop et al. (2016) sustenta que os alunos que dominarem essa prática poderão desenvolver soluções cujos componentes atendem tanto as necessidades dos problemas em mãos quanto daqueles que, eventualmente, surgirem adiante.

6. **Criação de abstrações computacionais**

Como já discutido anteriormente, o trabalho de criação de uma abstração traduz-se na conceitualização de terminadas ideas ou processos, por meio do qual explicita-se determinados aspectos em detrimento de outros, entendidos como menos relevantes para descrição.

Como também já visto, a construção e o emprego dessa entidade conceitual desempenha papel fundamental na atividade matemática e científica, e toma forma na codificação de programas, na geração e visualização de dados, na definição do escopo e escala de um problema, ou na criação de modelos, concebidos para exploração e/ou melhor compreensão de um fenômeno (WEINTROP et al., 2016).

Tratando-se especificamente de abstrações computacionais, Weintrop et al. (2016) sugere alunos que adquirirem a capacitação para criá-las estarão aptos, ao mesmo, para identificá-las e empregá-las em função dos seus próprios objetivos matemáticos e científicos.

7. ***Troubleshooting* e *Debugging***

   O uso da expressão *troubleshooting* aplica-se à investigação sistemática e à compreensão do porquê algo não está se comportando como esperado. Em ciência da computação, esse processo é também conhecido como *debugging* (depuração, em tradução livre).

   Dentre as estratégias compreendidas pelo termo, incluem-se a definição clara do problema e o teste metódico do sistema sobre o qual recai o erro. Nos dois casos o que se objetiva é o seu isolamento e reprodução.

   Nas disciplinas STEM, o estudo do tema desempenha papel preponderante, dado que nelas o surgimento de erros e comportamentos inesperados são uma constante. Contudo, a forma como que eles são depurados varia, sendo possível, portanto, encontrar em cada uma delas um conjunto de técnicas e ferramentas específicas desenhadas com esse propósito.

   Weintrop et al. (2016) sugere que o exercício dessa prática tornará o aluno apto a identificar, isolar, reproduzir, e por fim, corrigir comportamentos e erros inesperados em problemas ou modelos com os quais estiver trabalhando.

### 3.2.4 Práticas de pensamento sistêmicos

Múltiplos problemas com quais lidamos atualmente envolvem um conjunto numeroso de componentes que se inter-conectam e inter-relacionam.

A habilidade de concebê-los sob essa perspectiva configura o que se denomina como "pensamento sistêmico", e se diferencia essencialmente das formas de análise tradicional, com as quais os comportamentos individuais das partes são tomadas como simples agregadores

do todo. Assim, surge a noção de que certas propriedades resultantes não podem ser explicadas sem que sejam considerados a atividade integral do sistema.

No contexto de uma sociedade em que a disponibilidade de dados é abundante - onde expressões como *big data* ganham força - emergem métodos de análise suportados pela premissa de que sistemas devem ser estudados, investigados, e entendidos no seu conjunto, e não em função das suas partes.

Essa concepção não é essencialmente nova. Esse fato é exemplificado por alguns modelos e conceitos científicos, existentes há décadas, cuja formulação tem a perspectiva sistêmica como referencial. Alguns bons exemplos incluem a seleção e dinâmica populacional em ecologia, o sistema respiratório humano, e a lei dos gases ideais, originários da física e química (WEINTROP et al., 2016).

Weintrop et al. (2016) sinaliza que embora os aspectos do pensamento computacional apresentados até aqui por sua taxonomia não estejam diretamente associados a noção de pensamento sistêmico, ferramentas computacionais constituem um meio poderoso para tornar essas ideias acessíveis a seus aprendizes.

# Referências

ABRAHAMSON, D.; WILENSKY, U. Problab goes to school: design, teaching, and learning of probability with multi-agent interactive computer models. 2005. Disponível em: <https://ccl.northwestern.edu/2005/Abr+Wil_CERME4.pdf>. Citado na página 19.

BARR, V.; STEPHENSON, C. Bringing computational thinking to K-12. *ACM Inroads*, v. 2, n. 1, p. 48, 2011. ISSN 21532184. Disponível em: <http://dl.acm.org/citation.cfm?doid=1929887.1929905>. Citado 4 vezes nas páginas 19, 23, 26 e 27.

BOX, G.; DRAPER, N. *Empirical model-building and response surfaces*. Wiley, 1987. (Wiley series in probability and mathematical statistics: Applied probability and statistics). ISBN 9780471810339. Disponível em: <https://books.google.com.br/books?id=QO2dDRufJEAC>. Citado na página 33.

BRENNAN, K.; RESNICK, M. New frameworks for studying and assessing the development of computational thinking. *annual American Educational Research Association meeting, Vancouver, BC, Canada*, p. 1–25, 2012. Disponível em: <http://web.media.mit.edu/{~}kbrennan/files/Brennan{_}Resnick{_}AERA201>. Citado na página 23.

DJORGOVSKI, S. Virtual Astronomy, Information Technology, and the New Scientific Methodology. *Seventh International Workshop on Computer Architecture for Machine Perception (CAMP'05)*, p. 125–132, 2005. Disponível em: <http://ieeexplore.ieee.org/document/1508175/>. Citado 3 vezes nas páginas 1, 2 e 5.

ESCOBAR, M. *Artificial intelligence: here's what you need to know to understand how machines learn*. 2017. Disponível em: <https://theconversation.com/artificial-intelligence-heres-what-you-need-to-know-to-understand-how-machines-learn-72004>. Acesso em: 10 de Setembro de 2018. Citado na página 2.

GOUGH, W. A. J. *Practical Arithmetic: In Four Books*. [S.l.: s.n.], 1813. Citado na página 7.

HATCH, R. A. *Ptolemy's planetary models*. 1998. Disponível em: <http://users.clas.ufl.edu/ufhatch/pages/03-Sci-Rev/SCI-REV-Home/resource-ref-read/chief-systems/08-0PTOL3-WSYS.html>. Acesso em: 11 de Setembro de 2018. Citado na página 3.

LEE, I. et al. Computational thinking for youth in practice. *ACM Inroads*, ACM, New York, NY, USA, v. 2, n. 1, p. 32–37, fev. 2011. ISSN 2153-2184. Disponível em: <http://doi.acm.org/10.1145/1929887.1929902>. Citado na página 19.

LOCKE, J. *An essay concerning human understanding*. New York, NY, USA: Oxford Univerty Press, 1690/1979. Citado na página 9.

MALONEY, J. H. et al. Programming by choice: urban youth learning programming with scratch. ACM, p. 367–371, 2008. Disponível em: <http://dblp.uni-trier.de/db/conf/sigcse/sigcse2008.html#MaloneyPKRR08>. Citado na página 38.

NATIONAL RESEARCH COUNCIL. *Report of a workshop on the scope and nature of computational thinking.* Washington, DC: The National Academies Press, 2010. Citado 2 vezes nas páginas 23 e 25.

OREMUS, W. *The Fascinating, Mostly Failed History of "Teaching Machines".* 2015. Disponível em: <http://www.slate.com/articles/slate_plus/technology/2015/10/the_history_of_learning_machines_from_sidney_presser_and_b_f_skinner_to.html>. Acesso em: 03 de Novembro de 2018. Citado na página 22.

PATHAK, J. et al. Model-free prediction of large spatiotemporally chaotic systems from data: A reservoir computing approach. *Phys. Rev. Lett.*, American Physical Society, v. 120, p. 024102, Jan 2018. Disponível em: <https://link.aps.org/doi/10.1103/PhysRevLett.120.024102>. Citado na página 2.

PIWEK, P. Introduction to Computational Thinking. 2016. Disponível em: <http://doer.col.org/handle/123456789/6196>. Citado 7 vezes nas páginas 6, 9, 10, 11, 12, 13 e 14.

PURTILL, C. *Apple, IBM, and Google don't care anymore if you went to college.* 2018. Disponível em: <https://qz.com/work/1367191/apple-ibm-and-google-dont-require-a-college-degree/>. Acesso em: 12 de Outubro de 2018. Citado na página 18.

RESNICK, M. *Let's teach kids to code.* 2012. Disponível em: <https://www.ted.com/talks/mitch_resnick_let_s_teach_kids_to_code?language=en>. Acesso em: 14 de Outubro de 2018. Citado 3 vezes nas páginas 19, 20 e 38.

SENGUPTA, P. et al. Integrating computational thinking with K-12 science education using agent-based computation: A theoretical framework. *Education and Information Technologies*, v. 18, n. 2, p. 351–380, 2013. ISSN 13602357. Citado 2 vezes nas páginas 10 e 18.

VERHAGE, J. *This Robot Said to Sell Facebook. Next Time It May Be Right.* 2017. Disponível em: <https://www.bloomberg.com/news/articles/2017-11-21/this-robot-said-to-sell-facebook-next-time-it-may-be-right>. Acesso em: 11 de Setembro de 2018. Citado na página 4.

VUTHA, A. *Could machine learning mean the end of understanding in science?* 2018. Disponível em: <https://theconversation.com/could-machine-learning-mean-the-end-of-understanding-in-science-98995>. Acesso em: 10 de Setembro de 2018. Citado 3 vezes nas páginas 2, 3 e 4.

WEINTROP, D. et al. Defining Computational Thinking for Mathematics and Science Classrooms. *Journal of Science Education and Technology*, Springer Netherlands, v. 25, n. 1, p. 127–147, 2016. ISSN 15731839. Citado 20 vezes nas páginas 1, 18, 19, 22, 23, 24, 25, 28, 29, 31, 32, 33, 34, 35, 36, 37, 38, 39, 40 e 41.

WILENSKY, U. *Netlogo.* 1999. Http://ccl.northwestern.edu/netlogo/. Citado na página 19.

WING, J. Computational Thinking. *Commun. ACM*, ACM, New York, NY, USA, v. 49, n. 3, p. 33–35, 2006. Citado 5 vezes nas páginas 1, 6, 8, 21 e 22.

WING, J. Computational thinking and thinking about computing. v. 366, p. 3717–25, 11 2008. Citado 2 vezes nas páginas 5 e 34.