

# O Pensamento Computacional como Recurso para Aprendizagem Científica

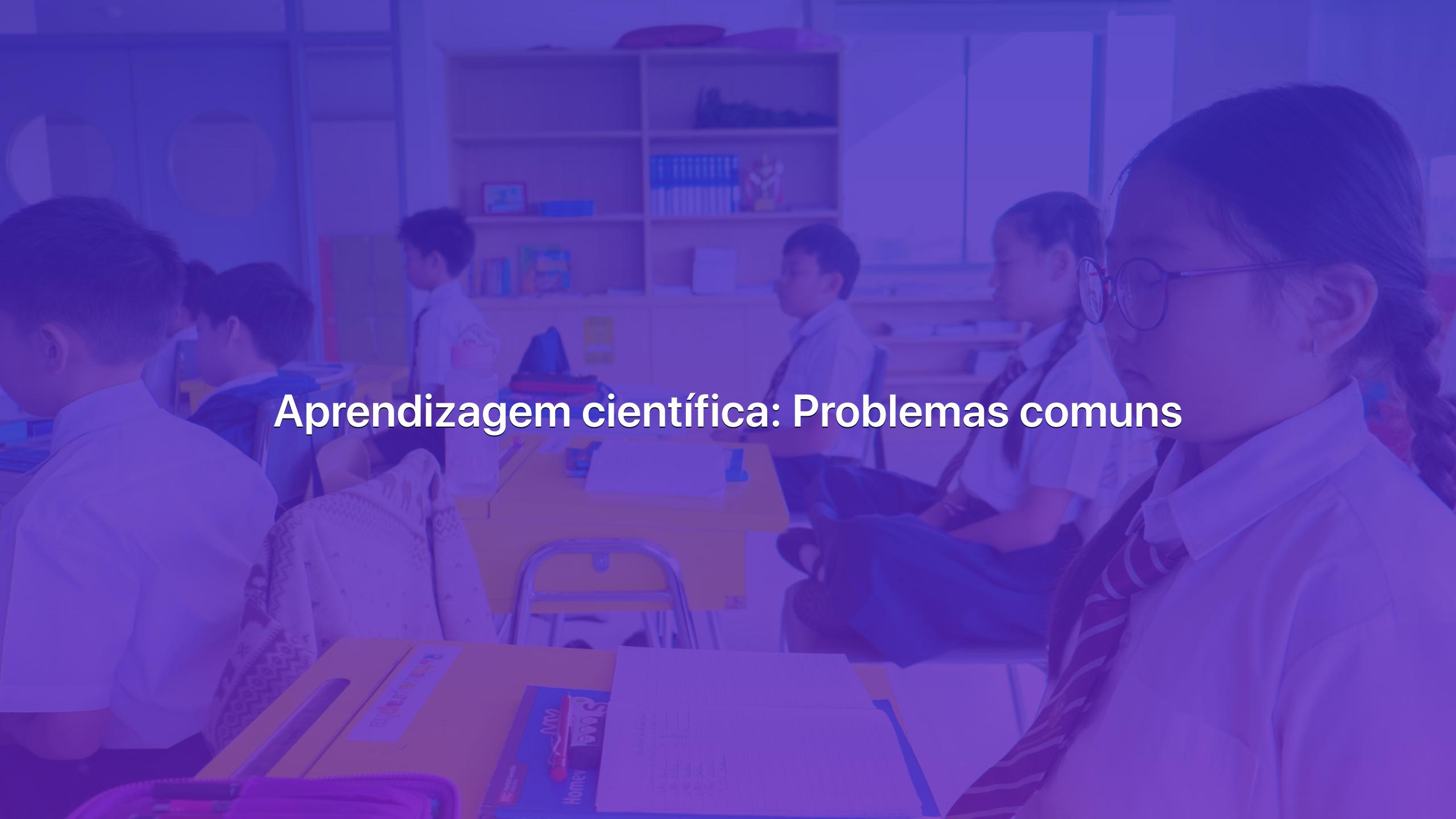
**Autor:** Hellon Canella Machado

**Orientador:** Dr. Hélio Salim de Amorim



# Tópicos

1. Enumerar problemas comuns à educação científica.
2. Apresentar pensamento computacional.
3. Demonstrar como esse conceito tem favorecido a aprendizagem científica.
4. Exemplo de atividade.

A photograph of a classroom. In the foreground, several students are seated at their desks, facing towards the front of the room. One student on the right side of the frame is wearing glasses and has long hair tied back. The desks are covered with various school supplies like notebooks and pens. In the background, there are wooden bookshelves filled with books and other classroom materials. The lighting is bright, typical of an indoor school environment.

# Aprendizagem científica: Problemas comuns

Segundo PISA 2015, apenas 30,6% dos alunos brasileiros alcançaram uma patamar de competência básico na área de ciências.

# Problemas recorrentes

Perceptíveis para professores veteranos

- 1. Dificuldade para transpor a contextos novos conceitos e estratégias de resolução de problemas aprendidos.**  
1. Quando o formato do problema muda, os alunos apresentam dificuldades para aplicar à nova situação os conceitos e algoritmos adquiridos.
  
- 2. Escassez de significado dos resultados obtidos.**  
2. Aplicação, **de modo acrítico**, dos algoritmos de um modelo de problema, visando quase sempre uma resposta numérica, única e "correta".
  
- 3. Ausência de motivação para aprendizagem de temas científicos.**  
3. A percepção de insignificância dos temas tratados em sala de aula leva à desmotivação

## Essência dos problemas

O modelo de ensino e avaliação vigente, por exemplo, desestimula a **aprendizagem significativa** ao privilegiar a reprodução de "**respostas corretas**", induzindo o estudante a **agir como um autômato**.

## Exemplo: um problema de física

Qual é velocidade após 32s de um objeto lançado para alto com uma velocidade de 20m/s?

A recorrência desse modelo de problema ilustra a preocupação reprodutiva do atual modelo de ensino, que colabora para a transmissão de uma imagem irrealista da atividade científica.

# Pensamento Computacional



# Computational Thinking

It represents a universally applicable attitude and skill set everyone, not just computer scientists, would be eager to learn and use.



**C**omputational thinking builds on the power and limits of computing processes, whether they are executed by a human or by a machine. Computational methods and models give us the courage to solve problems and design systems that no one of us would be capable of tackling alone. Computational thinking confronts the riddle of machine intelligence: What can humans do better than computers? and What can computers do better than humans? Most fundamentally it addresses the question: What is computable? Today, we know only parts of the answers to such questions.

Computational thinking is a fundamental skill for everyone, not just for computer scientists. To reading, writing, and arithmetic, we should add computational thinking to every child's analytical ability. Just as the printing press facilitated the spread of the three Rs, what is appropriately incestuous about this vision is that computing and computers facilitate the spread of computational thinking.

Computational thinking involves solving problems, designing systems, and understanding human behavior, by drawing on the concepts fundamental to computer science. Computational thinking includes a range of mental tools that reflect the breadth of the field of computer science.

Having to solve a particular problem, we might ask: How difficult is it to solve? and What's the best

cisely. Stating the difficulty of a problem accounts for the underlying power of the machine—the computing device that will run the solution. We must consider the machine's instruction set, its resource constraints, and its operating environment.

In solving a problem efficiently, we might further ask whether an approximate solution is good enough, whether we can use randomization to our advantage, and whether false positives or false negatives are allowed. Computational thinking is reformulating a seemingly difficult problem into one we know how to solve, perhaps by reduction, embedding, transformation, or simulation.

Computational thinking is thinking recursively. It is parallel processing. It is interpreting code as data and data as code. It is type checking as the generalization of dimensional analysis. It is recognizing both the virtues and the dangers of aliasing, or giving someone or something more than one name. It is recognizing both the cost and power of indirect addressing and procedure call. It is judging a program not just for correctness and efficiency but for aesthetics, and a system's design for simplicity and elegance.

Computational thinking is using abstraction and decomposition when attacking a large complex task or designing a large complex system. It is separation of concerns. It is choosing an appropriate representation for a problem or modeling the relevant aspects of a problem to make it tractable. It is using invariants to describe a system's behavior succinctly and declaratively. It is having the confidence we can

# Pensamento Computacional

Expressão presente na literatura desde da década de 80, mas que ganha popularidade com uma publicação, no ano de 2006, pela Professora Jeannette Wing, então professora da Universidade Carnegie Mellon.

"O pensamento computacional envolve **resolver problemas**, **projetar sistemas** e compreender comportamento, baseando-se em **conceitos fundamentais** à ciência da computação ."

— Jeannette Wing



**mas quais conceitos?**

# Conceitos fundamentais

Dos conceitos fundamentais à ciência da computação, três necessitam ser compreendidos para o entendimento do pensamento computacional.

São eles o **algoritmo**, o **problema computacional** e a **abstração**

algoritmo

# Algoritmo

Solução para um problema que satisfaz as seguintes condições:

01. ser uma sequencia de passos que leve à solução

02. ser um processo finito

03. resolver qualquer instância do problema

*A D D I T I O N.***Exemplo do sec. XVIII**

"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"

**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.

I. Place the numbers so that each figure 54327 A may stand directly underneath (or in the same 8062 B perpendicular row with) the figures of the 5041 C same value, that is: units under units, tens 67430 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.

*A D D I T I O N.*

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

"[...] 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;"

**GENERAL RULE.**

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

*54327 A  
8062 B  
5041 C*  
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 67430 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.

**A D D I T I O N.**

"[...] 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 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."

— John Gough - *Practical*

**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.

54327

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 8062 under tens, hundreds under hundreds, &c. 5041

67430

Then drawing a line under them; begin the Addition in 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 line (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.

**problema computacional**

# Problema Computacional

Conjunto de perguntas para qual um computador pode prover respostas.

Exemplo:

Encontre a razão entre dois inteiros A e B, onde B é diferente de zero.

Qualquer par de inteiros, em que B é diferente de 0, representa uma **instância** possível do problema.

Um problema computacional pode ter portanto um conjunto infinito de instâncias.

A definição do problema computacional, ou seja, de todas as instâncias de um problema é condição essencial para elaboração de um algoritmo.

O algoritmo - o conjunto de instruções a serem executadas pelo computador - é, por sua vez, o que torna possível a **automação da solução**.

abstração

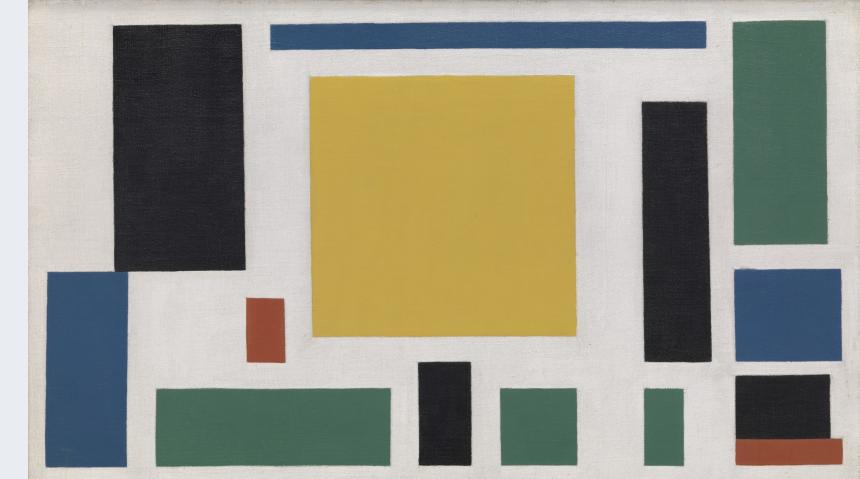
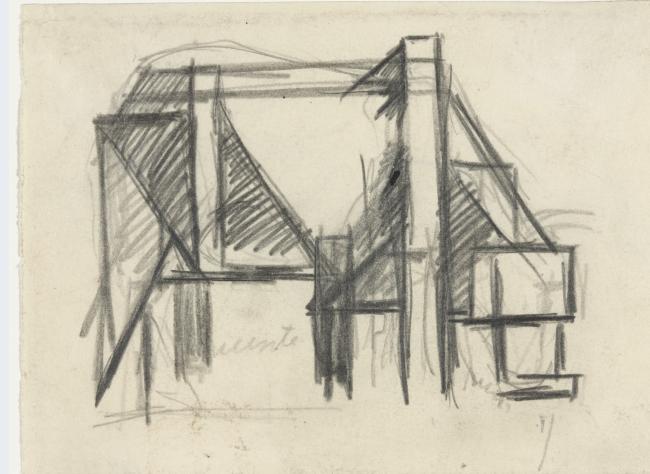
# Abstração

A abstração é, em essência, uma **redução** da realidade.

Uma **representação**.

Abstrair, nesse sentido, equivale à incorporar parte da realidade em um **modelo**, em desfavor de detalhes julgados como menos importantes.

# Um animal. Quatro modelos.



O grau de detalhamento de um modelo dependerá sempre das circunstâncias do problema.

Desprezar os efeitos do atrito na trajetória de um foguete é conveniente se estamos ensinando física básica, mas impróprio no contexto de um programa aeroespacial.

**abstração como cápsula**

# Abstração como cápsula

Em computação e engenharia, a construção de sistemas depende da criação módulos.

Nestes casos, o processo de abstração se caracteriza pelo **encapsulamento** de componentes responsáveis pela **implementação** das funcionalidades.

Busca-se deixar visível apenas a **interface**, os componentes que caracterizam o modelo e permitem a sua manipulação externa.

## Exemplo: planetário mecânico

À esquerda temos sua **interface**, superfície metálica responsável por externalizar os elementos que caracterizam o modelo, e esconder os detalhes da **implementação** - sistema de rodas dentadas, à direita.

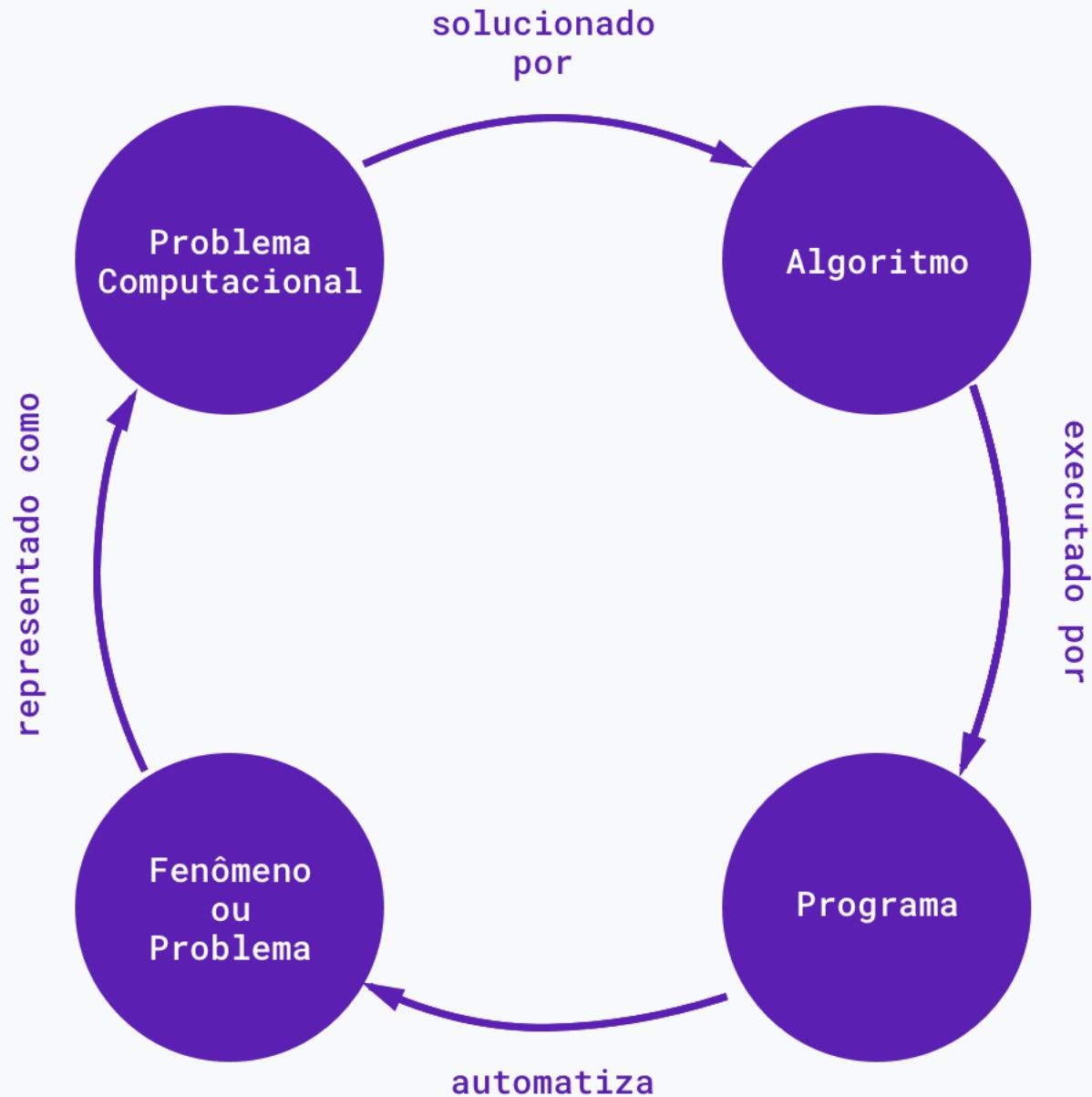


# **pensamento computacional: panorama**

# Panorama.

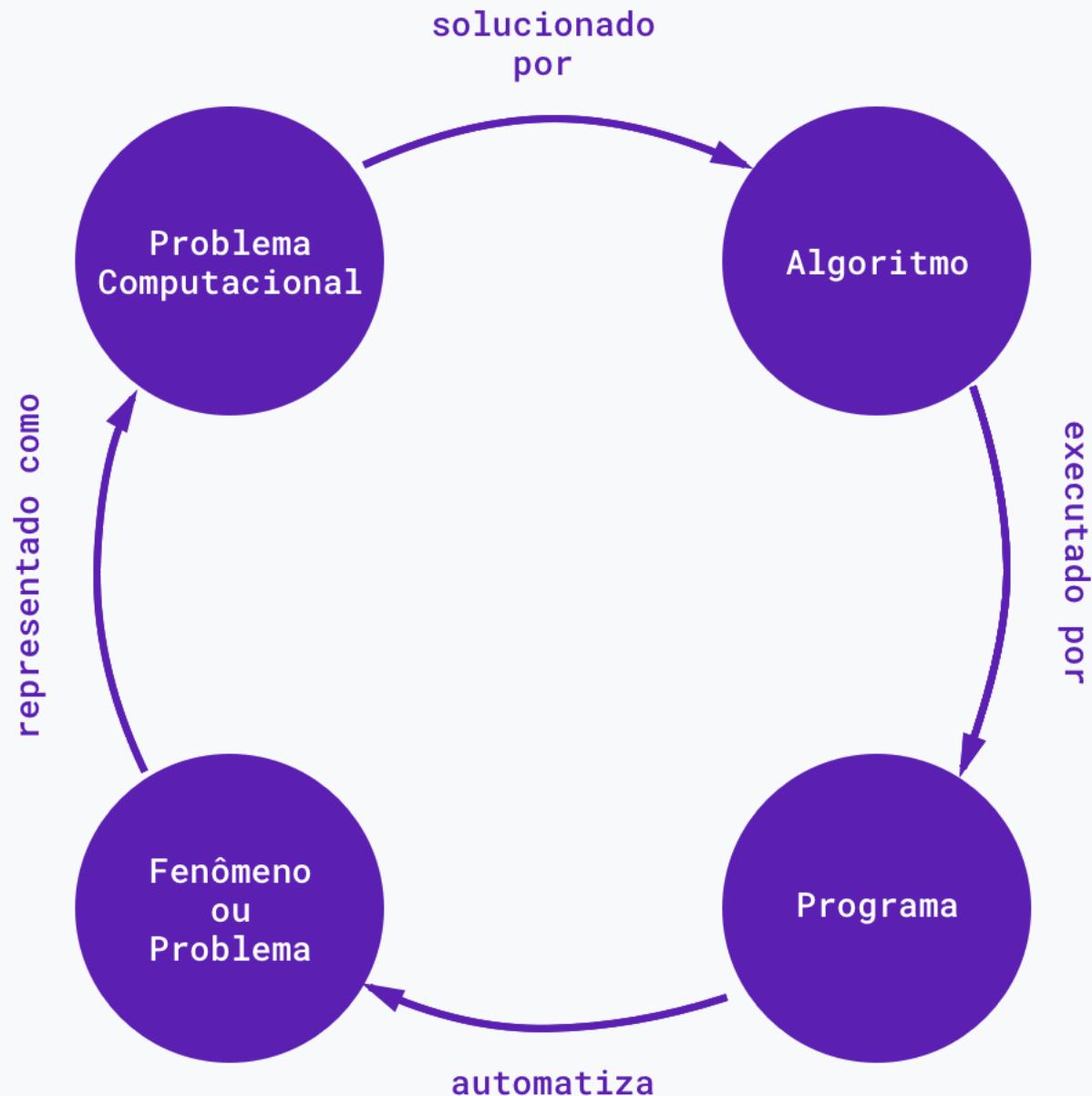
Problemas do mundo **real** necessitam ser expressos como **problemas computacionais**, para receberem tratamento computacional.

A resolução necessita então ser sequenciada - expressa em um **algoritmo**.



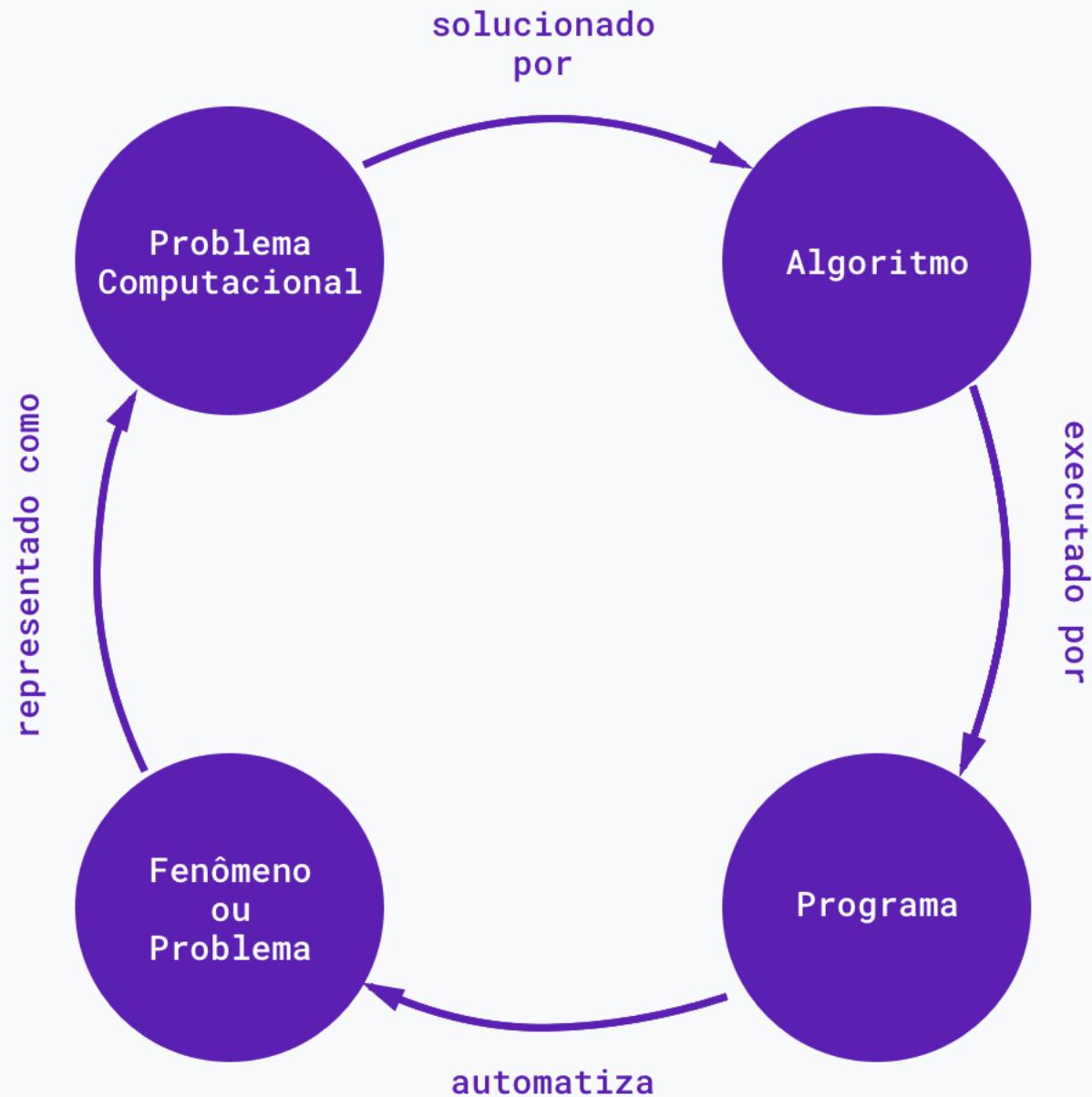
Problema computacional e algoritmo compõe assim uma representação, uma **abstração** do problema original.

A **automação** é feita com a ajuda de uma linguagem de programação ou mapeamento dos passos nos recursos de um software pronto.



A automação, por sua vez, permite que novos aspectos do problema sejam descobertos, o que pode levar a mais perguntas.

O ciclo é assim reiniciado.



---

## OBJETIVO

Demonstrar como o uso do pensamento computacional em **sala de aula** tem criado um **ambiente favorável** para a aprendizagem científica.

A young girl with blonde hair tied back, wearing a red sleeveless dress, is standing in a classroom. She is surrounded by numerous small, colorful paper fish scattered on the floor. The room has wooden chairs and desks, and various educational posters and drawings on the walls. The lighting is warm and natural.

# Implicações Educacionais

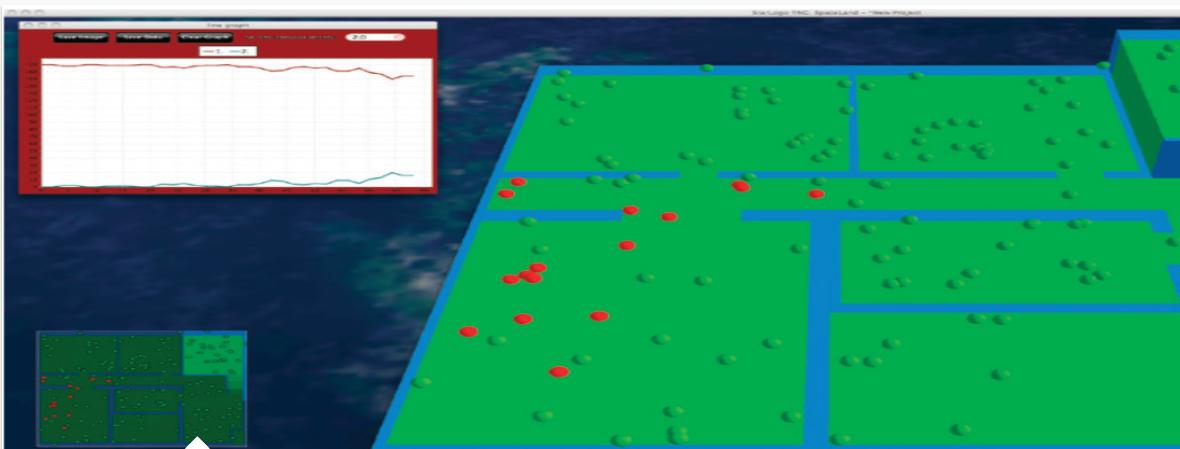
# Pensamento computacional na sala de aula



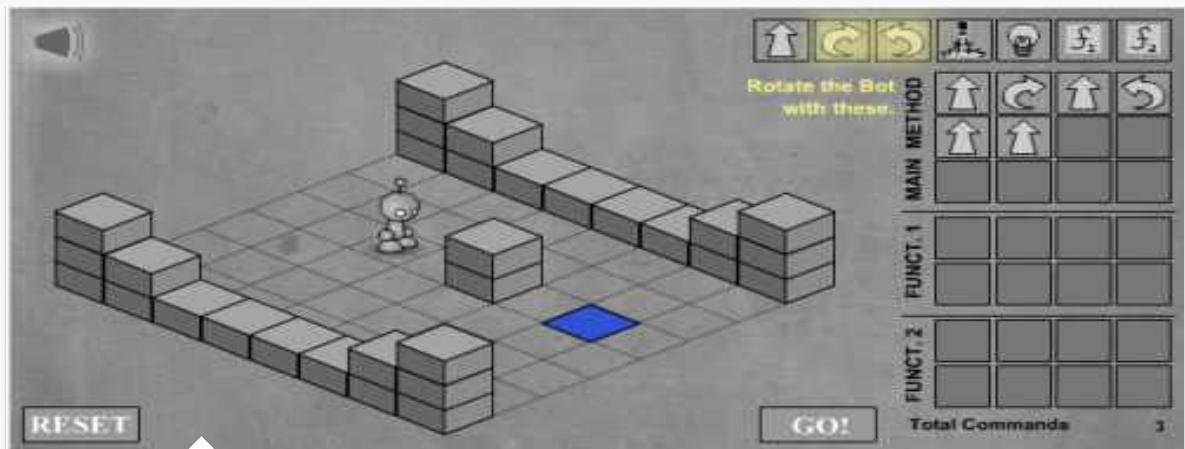
ENSINO DE PROGRAMAÇÃO



ROBÓTICA



MODELAGEM E SIMULAÇÃO



CRIAÇÃO DE JOGOS

# benefícios

# Benefícios, segundo a literatura

01. Alto índice de engajamento decorrente do caráter "mão na massa" das atividades.

02. Aumento do interesse para resolução de problemas abertos, e da percepção de que um problema pode ser resolvido de várias formas.

03. Aumento da persistência para a resolução de problemas difíceis.

04. Desenvolvimento de habilidades necessárias para o trabalho em grupo.

## Fonte:

1. LEE, I. et al. *Computational thinking for youth in practice*. ACM Inroads, v. 2, n. 1, p. 32, 2011.
2. BARR, V.; STEPHENSON, C. *Bringing computational thinking to K-12*. ACM Inroads, v. 2, n. 1

**motivações adicionais**



Carro autônomo da Waymo - subsidiária do Google

## Ubiquidade da computação

A computação está tomando todos os domínios da vida cotidiana.

Carros autônomos e robôs investidores que leem notícias compõem uma pequena amostra da dominância que a computação exerce atualmente.

No mundo do trabalho, um sem-número de atividades vêm sendo substituídos por robôs.

**Model-Free Prediction of Large Spatiotemporally Chaotic Systems from Data Using a Reservoir Computing Approach**

Aideep Pathak,<sup>1,2,\*</sup> Brian Hunt,<sup>3,4</sup> Michelle Girvan,<sup>1,3,2</sup> Zhixin Lu,<sup>1,3</sup> and Edward Ott<sup>1,2,†</sup>  
*<sup>1</sup>Department of Electrical and Computer Engineering, University of Maryland, College Park, Maryland 20742, USA*

*<sup>2</sup>Department of Physics, University of Maryland, College Park, Maryland 20742, USA*

*<sup>3</sup>Department of Mathematics, University of Maryland, College Park, Maryland 20742, USA*

*<sup>4</sup>Department of Electrical and Computer Engineering, University of Maryland, College Park, Maryland 20742, USA*

Received 24 July 2017; revised manuscript received 23 November 2017; published 12 January 2018

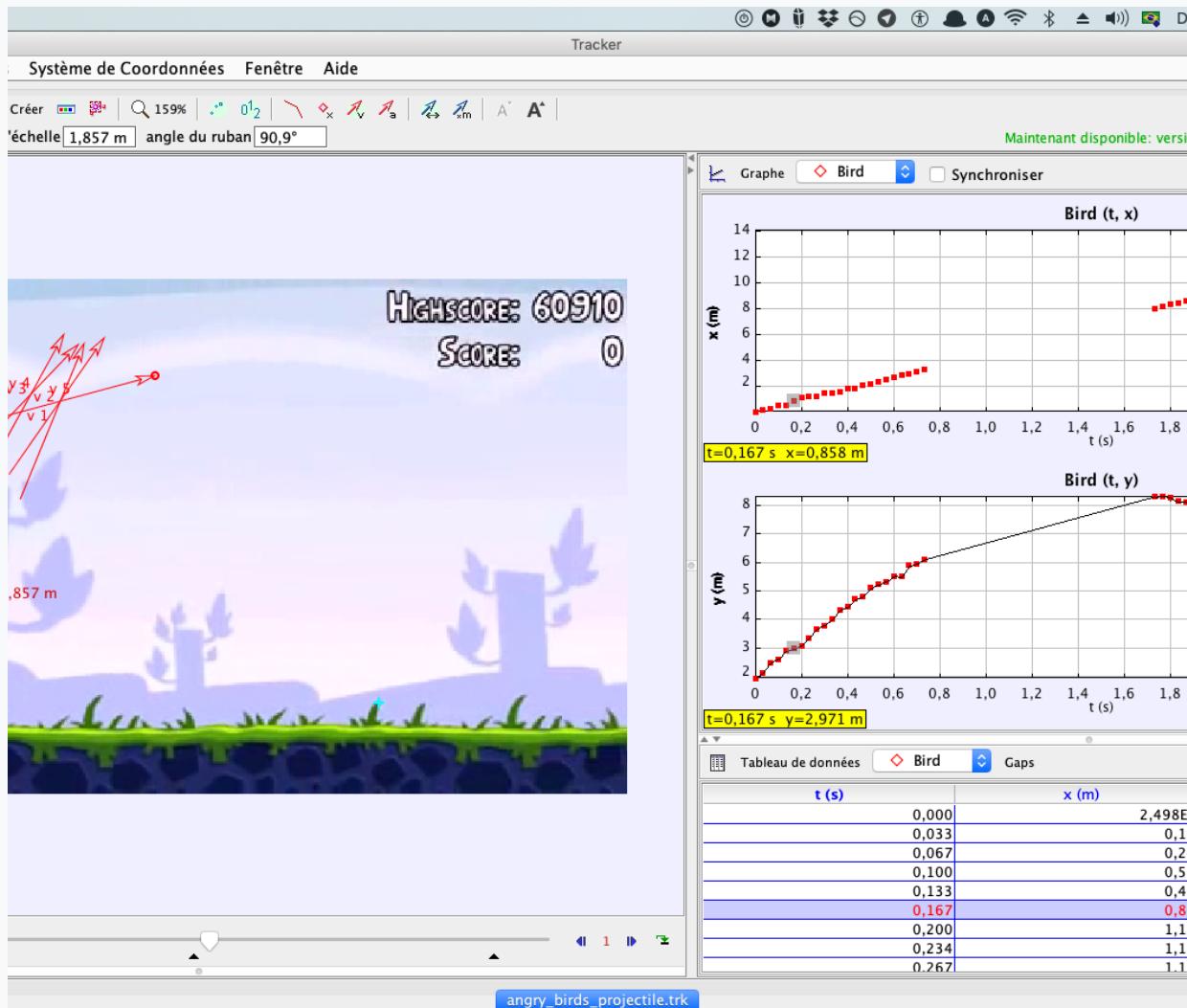
We demonstrate the effectiveness of using machine learning for model-free prediction of spatiotemporally chaotic systems of arbitrarily large spatial extent and attractor dimension purely from observation of the system's past evolution. We present a parallel scheme with an example implementation based on a reservoir computing paradigm and demonstrate the scalability of our scheme using the Kuramoto-Sivashinsky equation as an example of a spatiotemporally chaotic system.

[10.1103/PhysRevLett.120.024102](https://doi.org/10.1103/PhysRevLett.120.024102)

## Relevância para a prática científica

Na academia, o uso de inteligência artificial tem permitido a análise de um volume astronômico de dados.

Esse recurso têm permitido a predição da evolução de sistemas complexos, mesmo na ausência de modelos, como indica o artigo ao lado, publicado em 2018.



## Diálogo com centros de interesse

"Motivar é mudar as prioridades de uma pessoa" — Guy Claxton

O uso do computador viabiliza atividades com **conteúdo scientificamente relevante**, de baixíssimo custo, que dialogam com os **centros de interesse** dos estudantes.

Tomar como ponto de partida suas preferências é uma estratégia essencial para atrair interesse, engajar e permitir que a **aprendizagem significativa** se desenvolva.



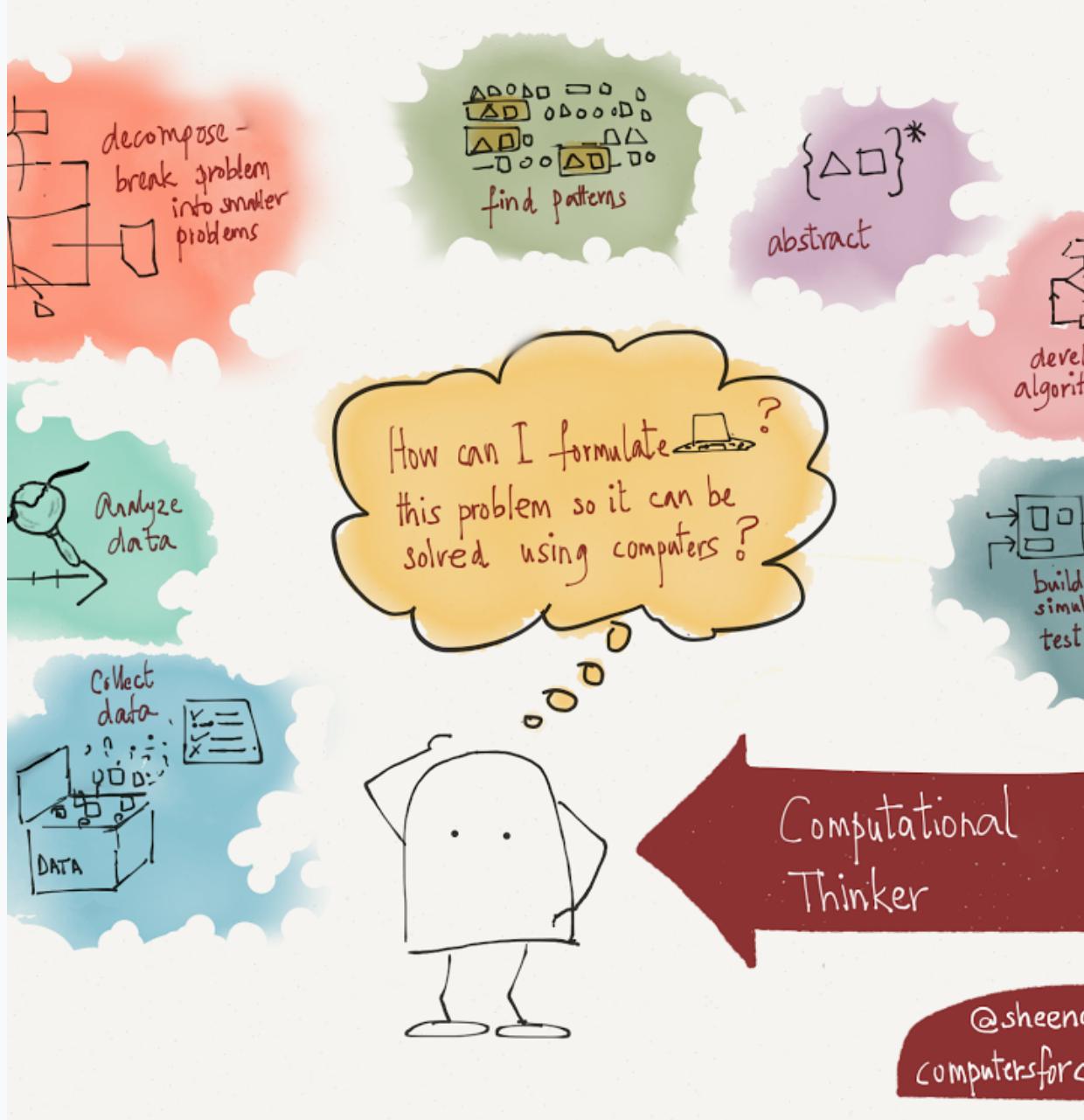
# Inversão da relação com a tecnologia

Nativos digitais apenas consomem tecnologia.

Apesar serem exímios usuários das novas tecnologias, as novas gerações não estão familiarizadas com a ideia de produzi-las.

Dotá-los com essa capacidade criativa permitirá o desenvolvimento de diversas competências, tais como:

1. Pensar em termos sistêmicos — “o todo não é soma das partes”
2. Estruturar as etapas de um projeto
3. Experimentar novas ideias - **prototipagem**
4. Decompor um problema em partes menores - **abstração**



## Contexto excelente

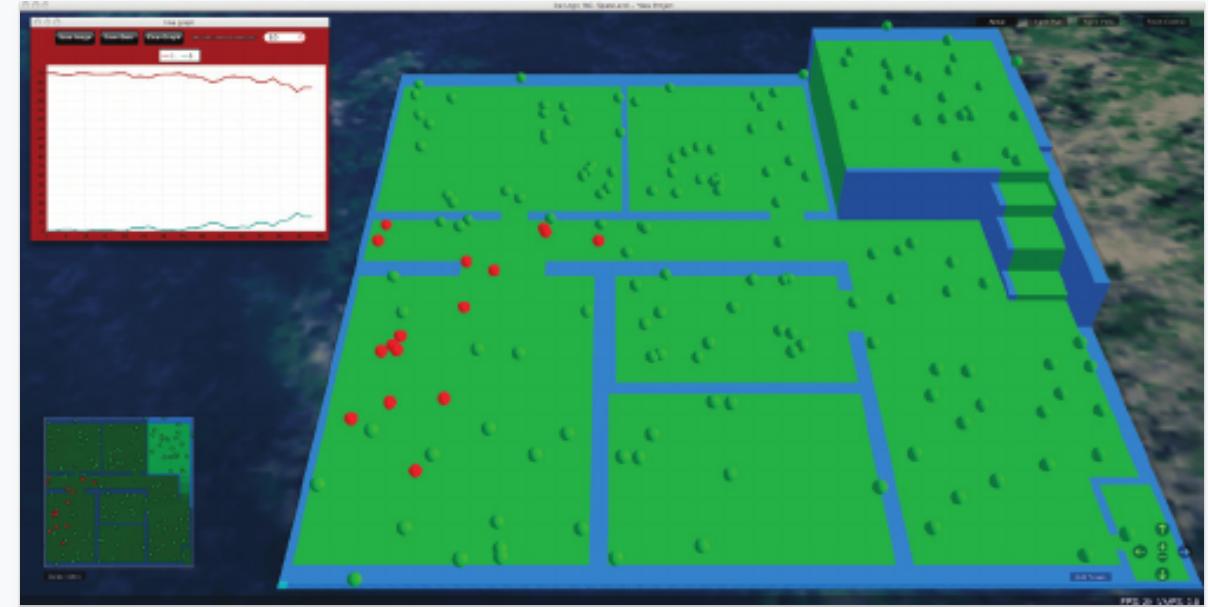
Pensar computacionalmente requer competências próprias da atividade científica.

Atividades com esse propósito permitem que estudantes decomponham problemas em unidades menores, e também, elaborem, avaliem e refinem modelos.

Ao tratar problemas de forma abertas, essa abordagem desconstrói a ideia que aprender física, por exemplo, é o mesmo que "acertar uma questão".

um exemplo

# Gripe suína na escola





**Fonte:** LEE, I. et al. *Computational thinking for youth in practice*. ACM Inroads, v. 2, n. 1, p. 32, 2011.

