

UNIVERSITÀ DEGLI STUDI DEL SANNIO

DIPARTIMENTO SCIENZE E TECNOLOGIE (DST)

CORSO DI LAUREA IN BIOTECNOLOGIE SETTORE CHIMICO-FARMACEUTICO

TESI DI LAUREA TRIENNALE

SVILUPPO DEL PENSIERO COMPUTAZIONALE NELLO STUDIO DELLE BIOTECNOLOGIE E DELLE STEM

RELATORE:

FRANCESCO NAPOLITANO

CANDIDATA:

MARIA CHIARA ARMENISE

MATRICOLA: 171005761

Anno accademico 2025/2026

Sommario

Abstract 3

Introduzione 5

Plasticità neuronale 7

Nozioni di logica e Computational thinking 14

Logica computazionale nelle STEM e la sua applicazione nelle biotecnologie 30

Insegnamento universitario 42

L'informatica nella scuola italiana 46

Tecniche di insegnamento del pensiero computazionale nelle diverse fasce d'età 50

Conclusioni 67

Abstract

Neuroplasticity is the nervous system's ability to alter its structure and function in response to experience and learning. This serves as the starting point for this study. The work aims to investigate how computational thinking, combined with basic logic concepts, can influence cognitive development and academic pathways in STEM disciplines, with a particular focus on biotechnology. After reviewing neuroimaging studies on brain changes associated with training, the paper examines applications of computational logic within biotechnology, such as feedback-based genetic circuits and bioprocessing. Given the evident importance of computer science and bioinformatics, the study explores how to effectively integrate them into STEM curricula and biotechnology programs; regarding the latter in Italian universities, each institution maintains its own syllabus, which can sometimes be relatively limited in scope. The paper then analyzes the role of computer science in the Italian school system, spanning from primary to upper secondary education and the teaching techniques used to foster computational thinking across different age groups, including "unplugged" and "plugged" methods, scaffolding, and the use of specific software. The literature suggests that the impact of computational thinking is generally positive or negligible, with a more pronounced effect observed during childhood and adolescence in various studies. Furthermore, the analysis highlights that computer science is an increasingly vital skill for the sciences, underscoring the need for its greater integration into the Italian school curriculum.

Introduzione

Fin da piccoli impariamo a sviluppare nozioni di logica: alcune di queste ci vengono insegnate, mentre altre derivano dalle conclusioni che traiamo dalle nostre esperienze; tuttavia, ci sono altre nozioni di logica che raramente ci vengono introdotte, fra cui nozioni teoriche appartenenti all'informatica; queste ci possono aiutare sia nello sviluppo cerebrale, sia allo studio di determinate materie dove la logica è fondamentale. Tra queste discipline rientrano quelle appartenenti all'ambito STEM, che comprendono le discipline scientifiche, tecnologiche, ingegneristiche e matematiche, incluse le biotecnologie che verranno prese più in considerazione. In questo elaborato vedremo anzitutto un minimo di plasticità neuronale, in modo da comprendere la motivazione di questo elaborato, quali sono le nozioni di logica più rilevati insieme al CT (computational thinking), quanto siano fondamentali per lo studio in generale nelle altre materie STEM e nelle biotecnologie in modo particolare parlando propriamente di come vengono utilizzate in questo settore. Faremo anche riferimento agli argomenti studiati nell'informatica e nella bioinformatica in questo determinato percorso di studi in Italia, quali sono le probabili prospettive nelle scuole elementari e medie, oltre alla presenza nelle scuole superiori. Infine, analizzeremo quale è l'influenza del pensiero computazionale nelle diverse fasce d'età osservando anche alcune tecniche di insegnamento utilizzate per la sua comprensione.

Plasticità neuronale

La neuroplasticità può essere definita come la capacità del sistema nervoso di modificare la propria struttura e il proprio funzionamento in risposta all'esperienza, agli stimoli ambientali e all'apprendimento. Essa costituisce uno dei principali meccanismi attraverso i quali il cervello si adatta alle richieste dell'ambiente e acquisisce nuove competenze. In tale prospettiva, l'apprendimento non rappresenta esclusivamente un cambiamento comportamentale, ma è associato a modificazioni dell'organizzazione e dell'efficienza dei circuiti neurali.

Un ruolo centrale è svolto dalla plasticità sinaptica, ovvero la capacità delle connessioni tra neuroni di modificare stabilmente la propria efficacia in funzione dell'esperienza. L'attività ripetuta, la pratica, la motivazione e le condizioni ambientali possono favorire tali modificazioni, determinando il rafforzamento o l'indebolimento delle connessioni sinaptiche. Ne consegue che l'apprendimento efficace dipende, almeno in parte, dalla ripetizione sistematica dell'attività, dalla sua intensità e dal coinvolgimento attivo dell'individuo. Il principio secondo cui l'allenamento di una determinata funzione può potenziarla evidenzia il rapporto diretto tra esperienza e riorganizzazione cerebrale.

La neuroplasticità dipendente dall'esperienza assume particolare rilevanza durante l'infanzia e l'adolescenza, periodi caratterizzati da una marcata capacità di adattamento del cervello. In questa fase, i processi di maturazione cerebrale si intrecciano con quelli determinati dall'esperienza, consentendo all'ambiente, all'educazione e all'allenamento di influenzare significativamente lo sviluppo dei circuiti neurali. L'apprendimento ripetuto può contribuire alla modificazione delle connessioni sinaptiche, dei dendriti e degli assoni, nonché dei processi di mielinizzazione. Tuttavia, i cambiamenti legati all'apprendimento devono essere distinti dai processi di maturazione fisiologica, che avvengono parallelamente durante lo sviluppo.

Le evidenze ottenute mediante tecniche di risonanza magnetica hanno permesso di osservare modificazioni cerebrali associate all'allenamento. Gli studi longitudinali, attraverso il confronto delle caratteristiche cerebrali prima e dopo un periodo di apprendimento, hanno fornito indicazioni particolarmente rilevanti sull'esistenza di cambiamenti neuroplastici associati all'esperienza. Tali modificazioni possono riguardare la struttura della sostanza grigia e bianca, la microstruttura delle connessioni e la connettività funzionale tra differenti regioni cerebrali. È importante, tuttavia, sottolineare che le misure ottenute mediante neuroimaging costituiscono indicatori indiretti dei processi biologici

sottostanti e non consentono di identificare univocamente il meccanismo cellulare responsabile del cambiamento osservato.

L'apprendimento di una nuova abilità costituisce un processo dinamico e progressivo. Le fasi iniziali, caratterizzate da un rapido miglioramento della prestazione, possono coinvolgere meccanismi e regioni cerebrali differenti rispetto alle fasi successive, nelle quali l'abilità viene progressivamente consolidata e automatizzata. Con la pratica, può verificarsi una riorganizzazione dell'attività cerebrale: alcune regioni coinvolte nel controllo attentivo e cognitivo possono diventare meno necessarie, mentre aumenta il contributo delle aree maggiormente specializzate nell'esecuzione dell'abilità acquisita. La neuroplasticità deve pertanto essere considerata come un processo dinamico, la cui manifestazione dipende dalla natura dell'attività, dalla durata dell'allenamento, dal livello di competenza raggiunto e dal momento in cui viene osservata.

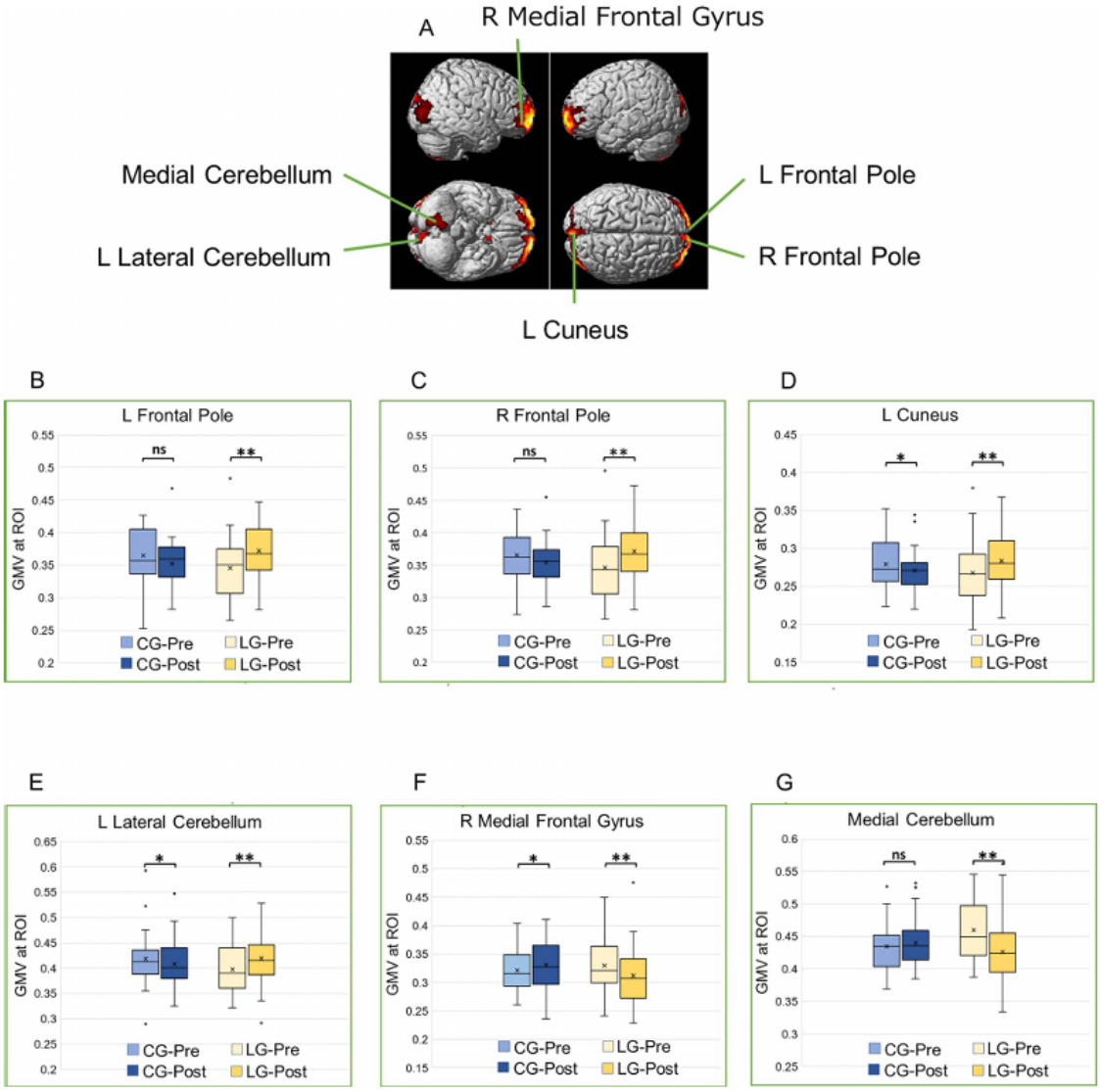
Un esempio particolarmente significativo del rapporto tra apprendimento e neuroplasticità è rappresentato dall'apprendimento della programmazione informatica, includendo il pensiero computazionale. Tale attività non richiede esclusivamente l'acquisizione di conoscenze specifiche relative ai linguaggi di programmazione, ma coinvolge competenze cognitive più generali, quali il pensiero logico, l'astrazione, la pianificazione, la risoluzione dei problemi, il ragionamento, la correzione degli errori e la perseveranza nel raggiungimento degli obiettivi. L'apprendimento della programmazione costituisce pertanto un modello utile per studiare come l'acquisizione di una competenza complessa possa modificare il funzionamento e la struttura cerebrale.

Uno studio condotto su studenti universitari privi di precedenti esperienze di programmazione ha evidenziato, dopo un corso di 15 settimane, modificazioni significative del volume della sostanza grigia in otto regioni cerebrali, comprendenti i poli frontali, il giro frontale mediale, il cuneo, il pallido e alcune regioni cerebellari. Tali variazioni non sono state semplicemente considerate come conseguenze generiche dell'attività universitaria, poiché il gruppo che aveva seguito il corso di programmazione è stato confrontato con un gruppo di controllo impegnato in altri percorsi formativi.

Particolarmente rilevante è risultata la relazione tra i cambiamenti neuroplastici e le prestazioni ottenute nell'apprendimento. Le modificazioni del volume della sostanza grigia nel polo frontale destro sono risultate positivamente associate alla qualità del prodotto finale realizzato dagli studenti. Tale regione è coinvolta in processi cognitivi complessi, tra cui la pianificazione, il controllo metacognitivo, la perseveranza orientata al

raggiungimento di obiettivi e il monitoraggio delle proprie azioni. La realizzazione di un progetto di programmazione richiede infatti di definire obiettivi, pianificare strategie, affrontare problemi, individuare e correggere errori attraverso il debugging e mantenere l'impegno per un periodo prolungato.

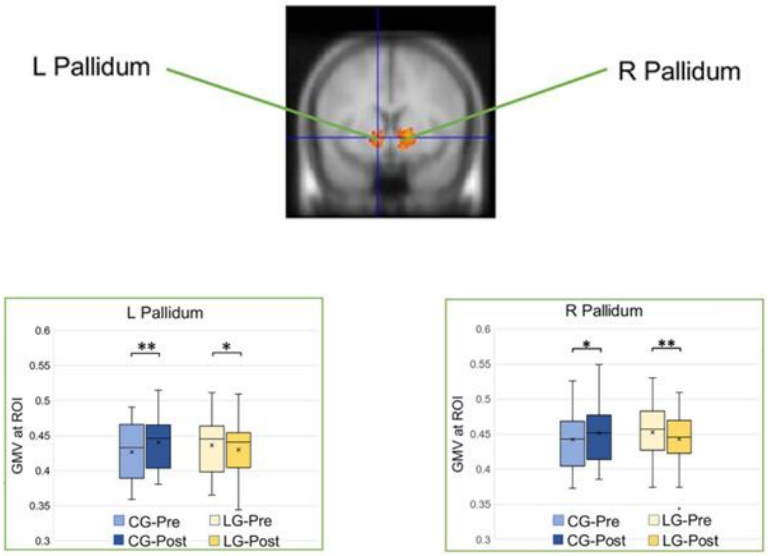
.



Analisi longitudinale del volume di materia grigia (GMV) corticale e cerebellare prima e dopo l'apprendimento della programmazione. A) Interazioni tempo-gruppo nell'analisi longitudinale di morfometria basata sui voxel (VBM). B–G) I box plot mostrano il GMV nella sede in cui è stata osservata l'interazione. B) Polo frontale sinistro; t(35)=−5,52; P<0,001. C) Polo frontale destro; t(35)=−6,13; P<0,001. D) Cuneo sinistro; t(35)=−5,28; P<0,001. E) Cervelletto laterale sinistro (lobulo posteriore e tuber); t(35)=−4,42; P<0,001. F) Giro frontale mediale destro; t(35)=4,87; P<0,001. G) Cervelletto mediale (uvula e tonsilla); t(35)=5,34; P<0,001. Note: CG/Pre, CG/Post, LG/Pre e LG/Post in ciascun box plot rappresentano, rispettivamente, i valori pre-apprendimento del gruppo di controllo (CG), post-apprendimento del CG, pre-apprendimento del gruppo di apprendimento (LG) e post-apprendimento del gruppo di apprendimento; ∗∗: P<0,01; ∗: P<0,05. Hongo T., Yakou T., Yoshinaga K., Kano T., Miyazaki M., Hanakawa T. (2022). Structural neuroplasticity in computer programming beginners. Oxford Academic 33, 5375–5381 https://doi.org/10.1093/cercor/bhac425

I risultati suggeriscono pertanto che la neuroplasticità del polo frontale possa essere associata allo sviluppo di capacità cognitive necessarie per affrontare problemi complessi e perseverare nel loro superamento. Analogamente, le modificazioni osservate nel giro frontale mediale destro sono risultate correlate alle prestazioni nei test di programmazione. Questa regione è associata a funzioni esecutive, ragionamento, monitoraggio dei processi cognitivi e applicazione di regole. Durante l'apprendimento della programmazione, tali funzioni risultano fondamentali per comprendere le istruzioni, recuperare le conoscenze precedentemente acquisite e applicarle alla risoluzione di problemi nuovi.

Anche le modificazioni del pallido bilaterale hanno mostrato una relazione con i risultati dei test. Questa struttura partecipa ai circuiti coinvolti nella motivazione e nell'apprendimento per rinforzo. L'acquisizione di una competenza complessa, infatti, non dipende esclusivamente dalle capacità cognitive, ma anche dalla motivazione a raggiungere un determinato risultato e dalla capacità di utilizzare il feedback derivante dall'esperienza per correggere progressivamente la propria prestazione.



Risultati dell'analisi longitudinale della sostanza grigia cerebrale. A) Interazioni Tempo × Gruppo nell'analisi longitudinale di morfometria basata sui voxel (VBM). B, C) I box plot in B e C mostrano il volume della sostanza grigia (GMV) nelle sedi in cui è stata osservata l'interazione. B) Pallido sinistro; t(35)=2,04; P=0,049. C) Pallido destro; t(35)=3,93; P<0,001. Note: CG/Pre, CG/Post, LG/Pre e LG/Post in ciascun box plot rappresentano, rispettivamente, i valori pre-apprendimento del gruppo di controllo (CG), post-apprendimento del CG, pre-apprendimento del gruppo di apprendimento (LG) e post-apprendimento del gruppo di apprendimento; ∗∗: P<0,01; ∗: P<0,05. Hongo T., Yakou T., Yoshinaga K., Kano T., Miyazaki M., Hanakawa T. (2022). Structural neuroplasticity in computer programming beginners. Oxford Academic 33, 5375–5381 https://doi.org/10.1093/cercor/bhac425

Questi risultati evidenziano come l'apprendimento di una competenza complessa non coinvolga una singola area cerebrale, ma una rete di regioni caratterizzate da funzioni differenti e complementari. Alcune strutture contribuiscono all'elaborazione delle informazioni e al ragionamento, altre alla memoria di lavoro, alla motivazione, alla pianificazione e al controllo metacognitivo. L'apprendimento determina quindi una progressiva riorganizzazione delle risorse neurali necessarie per acquisire e consolidare la nuova abilità.

È inoltre significativo che, nello studio sulla programmazione, non siano state osservate modificazioni strutturali nelle principali aree cerebrali associate al linguaggio, nonostante precedenti studi di neuroimaging avessero evidenziato l'attivazione di tali regioni durante la comprensione del codice. Una possibile interpretazione è che, nei soggetti principianti, l'elaborazione del codice non sia ancora sufficientemente automatizzata o integrata nei sistemi cognitivi linguistici. Ciò suggerisce che la neuroplasticità associata all'apprendimento possa variare in funzione del livello di competenza, modificandosi progressivamente durante il passaggio da principiante a esperto.

L'effetto dell'esperienza sul cervello è stato documentato anche attraverso numerosi studi di allenamento cognitivo e comportamentale condotti su bambini e adolescenti. Una revisione sistematica comprendente 71 studi ha rilevato che nell'87% dei casi erano presenti modificazioni significative dell'attivazione cerebrale, della struttura, della microstruttura o della connettività cerebrale in seguito a differenti forme di allenamento. Inoltre, in una parte consistente degli studi, i cambiamenti neurali risultavano associati al miglioramento delle prestazioni comportamentali. Tali evidenze sostengono l'ipotesi secondo cui la pratica intenzionale e ripetuta possa determinare modificazioni misurabili a livello cerebrale.

Il rapporto tra cambiamento neurale e miglioramento della prestazione, tuttavia, non deve essere interpretato in termini esclusivamente lineari. I cambiamenti cerebrali e comportamentali possono presentare tempi di sviluppo differenti: una modificazione neurale può precedere il miglioramento osservabile della prestazione, oppure il consolidamento di una nuova abilità può richiedere un periodo più lungo rispetto alle modificazioni iniziali dei circuiti cerebrali. Inoltre, l'allenamento può determinare contemporaneamente modificazioni cognitive, comportamentali e fisiologiche che influenzano le misure ottenute mediante neuroimaging.

Nel complesso, le evidenze disponibili indicano che la neuroplasticità costituisce una componente fondamentale dei processi di apprendimento. L'acquisizione di nuove conoscenze e competenze è accompagnata da modificazioni funzionali e, in determinate condizioni, strutturali del cervello. La pratica ripetuta, l'intensità dell'allenamento, la motivazione, il feedback e il coinvolgimento attivo dell'individuo rappresentano fattori rilevanti nel favorire tali adattamenti. Durante lo sviluppo, questa capacità risulta particolarmente significativa, poiché l'elevata plasticità del cervello giovane consente all'esperienza e all'ambiente di contribuire in maniera rilevante alla formazione e al consolidamento dei circuiti neurali.

Pertanto, l'apprendimento può essere considerato non soltanto come l'acquisizione di nuove informazioni o abilità, ma come un processo di riorganizzazione progressiva del sistema nervoso. La neuroplasticità fornisce infatti il substrato biologico attraverso il quale l'esperienza modifica il cervello, mentre la pratica consente di consolidare tali modificazioni e di trasformarle in competenze più efficienti e stabili. In questa prospettiva, comprendere i meccanismi neuroplastici dell'apprendimento rappresenta un elemento fondamentale per sviluppare strategie educative e programmi di allenamento sempre più efficaci, soprattutto durante le fasi dello sviluppo caratterizzate da una maggiore plasticità cerebrale.

Nozioni di logica e Computational thinking

Per comprendere il significato del pensiero computazionale (Computational Thinking, CT) e il suo ruolo nelle discipline STEM e nelle biotecnologie, è utile partire dalle nozioni fondamentali di logica, poiché il ragionamento computazionale si basa sulla capacità di analizzare informazioni, stabilire relazioni tra condizioni, organizzare procedure e prendere decisioni secondo regole precise. La logica costituisce quindi uno dei presupposti concettuali per comprendere il modo in cui i computer elaborano le informazioni e, più in generale, per sviluppare un approccio strutturato alla risoluzione dei problemi. Prima di analizzare in che modo la logica influenzi il funzionamento dei computer, è opportuno introdurre alcune nozioni fondamentali di logica. Queste conoscenze sono necessarie per sviluppare ragionamenti corretti e per comprendere quando una determinata condizione può essere considerata valida.

Per comprendere il funzionamento della logica utilizzata dai computer può essere utile introdurre anche il concetto di sistema di numerazione in base due, ovvero il sistema binario. Mentre nella vita quotidiana siamo abituati a utilizzare il sistema decimale, basato su dieci simboli, da 0 a 9, i computer utilizzano il sistema binario, costituito esclusivamente da due simboli: 0 e 1. Questo sistema può risultare, almeno concettualmente, più semplice da comprendere, poiché prevede soltanto due valori possibili. La comprensione del sistema binario è importante non soltanto per avvicinarsi al modo in cui il computer rappresenta ed elabora le informazioni, ma anche per comprendere i principi alla base della logica booleana, sulla quale si fondano numerosi componenti dei sistemi informatici.

Per introdurre il concetto di proposizione logica, possiamo considerare due affermazioni:

- Il computer ragiona in binario → A;
- Il computer ragiona in decimale →B.

A partire da queste proposizioni è possibile costruire diverse espressioni logiche:

- Il computer ragiona in binario → A;
- Il computer ragiona in decimale → B;
- Il computer non ragiona in decimale → NOT B;
- Il computer ragiona in binario e non in decimale → A AND NOT B;
- Il computer ragiona in binario o in decimale → A OR B;
- Il computer ragiona in binario e in decimale → A AND B.

Le proposizioni possono assumere un valore di verità, ovvero possono essere vere o false. Nella logica utilizzata nei sistemi informatici, tali valori possono essere rappresentati

rispettivamente con 1 e 0. Nel nostro esempio, A è vera e quindi può essere rappresentata con 1, o con V, mentre B è falsa e può essere rappresentata con 0, o con F. Attraverso questa rappresentazione è possibile determinare il valore di verità delle diverse espressioni ottenute combinando A e B. Questo principio costituisce la base del funzionamento dei circuiti logici, che utilizzano specifici operatori logici. Tra i principali troviamo AND, OR e NOT, corrispondenti rispettivamente alle operazioni “e”, “o” e “non”.

L'operatore OR merita una particolare attenzione, poiché può essere interpretato in senso inclusivo oppure esclusivo. Nell'OR inclusivo, l'espressione è vera quando almeno una delle due proposizioni è vera; pertanto, il risultato è vero anche quando entrambe le proposizioni sono vere. Nell'OR esclusivo, indicato come XOR, invece, l'espressione è vera soltanto quando una delle due proposizioni è vera e l'altra è falsa.

La differenza può essere evidenziata attraverso i seguenti esempi:

- Il computer ragiona in binario o in decimale → OR inclusivo;
- Il computer ragiona in binario o in decimale, ma non in entrambi → OR esclusivo (XOR).

Nell'utilizzo ordinario dell'operatore OR si fa generalmente riferimento all'OR inclusivo, mentre l'OR esclusivo viene utilizzato quando è necessario specificare che una sola delle due condizioni può essere vera.

Sulla base degli operatori logici è possibile costruire le tavole di verità, strumenti che consentono di rappresentare sistematicamente tutte le possibili combinazioni degli ingressi e il corrispondente risultato dell'operazione logica. I valori possono essere indicati mediante V e F, rispettivamente vero e falso, oppure mediante 1 e 0.

Una volta introdotti questi concetti, è possibile comprendere il funzionamento dei circuiti logici. Un circuito logico opera sulla base di valori binari ed è costituito da uno o più ingressi, o input, e da una o più uscite, o output. Gli ingressi vengono collegati attraverso specifiche porte logiche, che realizzano le operazioni corrispondenti agli operatori logici, come AND, OR e NOT.

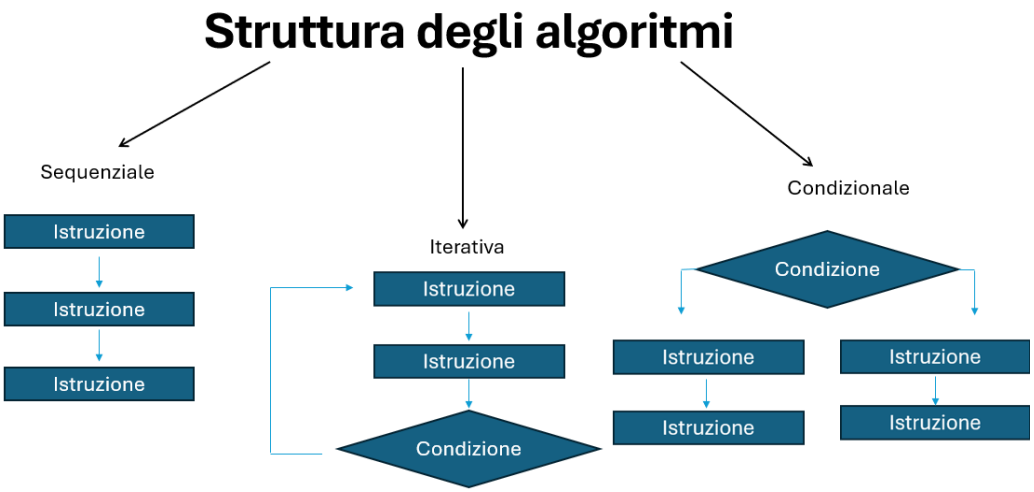
Fornendo determinati valori in ingresso e collegando correttamente le porte logiche, il circuito produce un determinato valore in uscita. Anche in questo caso l'output può essere rappresentato mediante 0 e 1, che possono essere interpretati, in modo semplificato, come due stati differenti, ad esempio “spento” e “acceso”. Anche per i circuiti logici è possibile utilizzare delle tabelle per rappresentare le combinazioni degli input e i relativi output.

Il passaggio dalla logica agli algoritmi rappresenta un ulteriore livello di organizzazione del ragionamento. Se la logica permette di stabilire se una determinata condizione è vera o falsa e di combinare diverse condizioni attraverso operatori specifici, l'algoritmo permette di organizzare una serie di operazioni in modo ordinato per raggiungere un determinato risultato. Un algoritmo può essere definito come un insieme finito di operazioni elementari, comprensibili ed eseguibili da un determinato esecutore, organizzate secondo un ordine preciso al fine di realizzare un determinato compito e, più in generale, risolvere un problema.

All'interno di questa definizione è possibile individuare alcune caratteristiche fondamentali degli algoritmi. Innanzitutto, l'atomicità, secondo la quale le operazioni devono essere costituite da passi elementari, non ulteriormente scomponibili nell'ambito considerato. È inoltre necessaria la finità, poiché un algoritmo deve essere costituito da un numero finito di passi. A questa caratteristica è collegato il principio di terminazione, secondo il quale l'esecuzione dell'algoritmo deve concludersi dopo un numero finito di operazioni. Un'altra caratteristica fondamentale è la non ambiguità: ogni istruzione deve essere sufficientemente precisa da poter essere interpretata nello stesso modo dall'esecutore. Infine, le operazioni devono essere effettive, ovvero concretamente eseguibili, e devono permettere di ottenere un risultato determinato secondo le specifiche del problema. I singoli passi che compongono un algoritmo vengono definiti istruzioni. Un'istruzione rappresenta, quindi, un'operazione elementare che può essere eseguita dall'esecutore.

Gli algoritmi possono essere organizzati principalmente secondo tre strutture fondamentali:

- struttura sequenziale, nella quale le istruzioni vengono eseguite una dopo l'altra secondo un ordine prestabilito;
- struttura condizionale, nella quale l'esecuzione di determinate istruzioni dipende dal verificarsi di una specifica condizione;
- struttura iterativa, nella quale una o più istruzioni vengono ripetute secondo una determinata condizione o per un numero prestabilito di volte.



Le strutture degli algoritmi si possono dividere in tre tipi principali: sequenziale, iterativa e condizionale. Qualsiasi algoritmo ha all'interno queste strutture che possono concatenarsi fra loro.

La struttura condizionale rappresenta in particolare un collegamento diretto con la logica: per stabilire quale parte dell'algoritmo debba essere eseguita, è infatti necessario valutare una o più proposizioni o condizioni logiche. In questo senso, gli operatori AND, OR e NOT e i valori vero/falso costituiscono strumenti concettuali fondamentali per descrivere le decisioni che un algoritmo deve prendere.

Le tre strutture possono essere combinate e concatenate tra loro in modi differenti, consentendo di descrivere algoritmi anche complessi. Per rappresentare graficamente un algoritmo possono essere utilizzati i diagrammi a blocchi, o flowchart. Un diagramma a blocchi rappresenta il flusso delle operazioni necessarie per realizzare la trasformazione dei dati iniziali nei risultati previsti dall'algoritmo. Ogni istruzione viene rappresentata mediante un blocco elementare, la cui forma varia in funzione del tipo di operazione rappresentata. I diversi blocchi vengono collegati tra loro attraverso frecce, che indicano la direzione del flusso di esecuzione. In generale, ogni blocco presenta frecce entranti e uscenti secondo la propria funzione. Fanno eccezione il blocco di inizio, che presenta una freccia uscente, e il blocco di fine, che presenta una freccia entrante.

Affinché un diagramma a blocchi sia strutturalmente valido, devono essere rispettate alcune condizioni fondamentali:

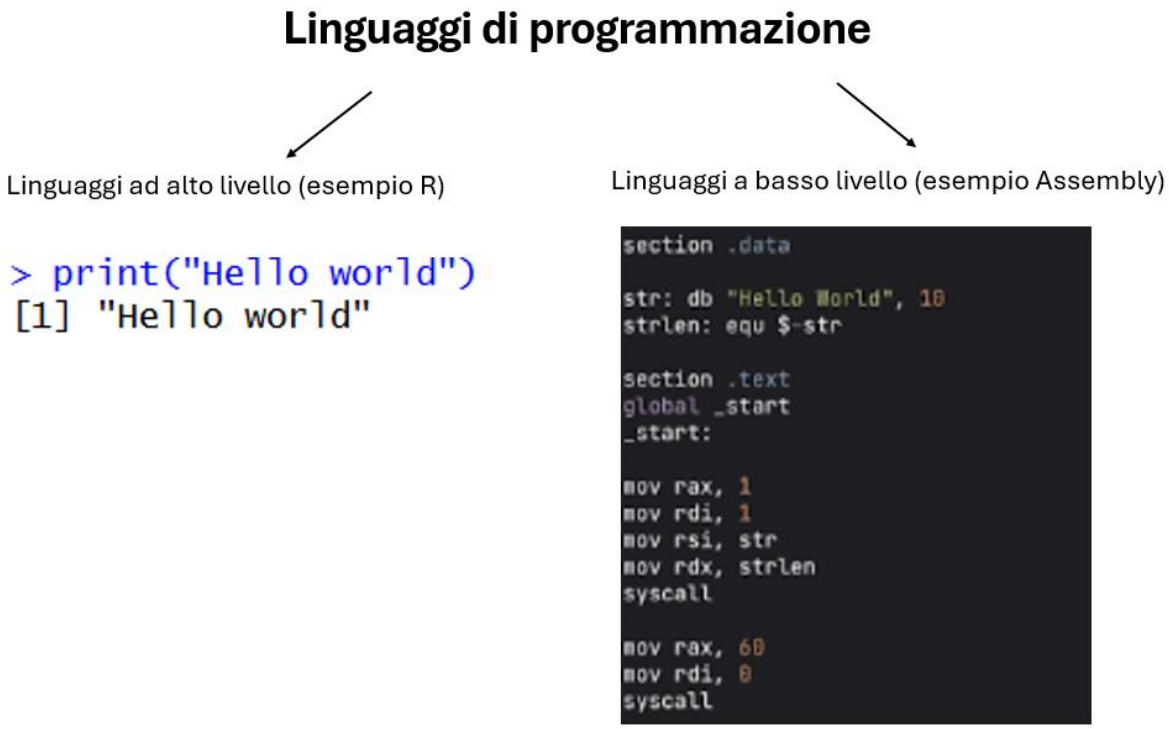
- ciascun blocco di azione o di lettura/scrittura deve avere una sola freccia entrante e una sola freccia uscente;

- ciascun blocco di controllo deve avere una sola freccia entrante e due frecce uscenti, corrispondenti ai possibili esiti della condizione;
- ciascuna freccia deve entrare in un blocco oppure innestarsi in un'altra freccia;
- ciascun blocco deve essere raggiungibile a partire dal blocco iniziale;
- ciascun blocco deve poter raggiungere il blocco finale.

Gli algoritmi costituiscono inoltre la base della programmazione: attraverso i linguaggi di programmazione è infatti possibile descrivere le procedure che il computer deve eseguire. La programmazione può quindi essere considerata uno degli strumenti attraverso cui un procedimento logico e algoritmico viene trasformato in istruzioni eseguibili da un sistema informatico. I linguaggi di programmazione possono essere distinti, in modo generale, in linguaggi ad alto livello e linguaggi a basso livello.

I linguaggi ad alto livello presentano una sintassi relativamente vicina al linguaggio umano e risultano quindi più semplici da comprendere e utilizzare da parte del programmatore. Un esempio è Python, ma appartiene a questa categoria anche R, un linguaggio particolarmente utilizzato nell'ambito dell'analisi statistica e dell'elaborazione dei dati, compresi alcuni contesti legati alle biotecnologie.

I linguaggi a basso livello, invece, sono caratterizzati da istruzioni più vicine a quelle direttamente comprensibili ed eseguibili dall'hardware. Risultano pertanto generalmente più complessi per il programmatore e richiedono un numero maggiore di istruzioni per descrivere operazioni che, nei linguaggi ad alto livello, possono essere espresse in maniera più sintetica. Tra i principali esempi di linguaggi a basso livello troviamo il linguaggio macchina e l'assembly. Il linguaggio macchina utilizza istruzioni espresse direttamente attraverso codici binari eseguibili dal processore, mentre l'assembly utilizza una rappresentazione simbolica di tali istruzioni, mantenendo comunque uno stretto rapporto con l'architettura e con l'hardware del computer.



Esempi di linguaggi di programmazione ad alto e a basso livello. Nel primo caso basta un'istruzione, nel secondo caso ci vogliono più istruzioni per ottenere lo stesso risultato. I due linguaggi usati vengono usati generalmente a scopi differenti, perciò si deve usare il linguaggio di programmazione più adatto.

La successione logica che va dalle proposizioni e dai valori di verità agli operatori logici, dai circuiti agli algoritmi e dagli algoritmi alla programmazione permette quindi di comprendere un aspetto fondamentale del pensiero computazionale: il computer non "risolve" un problema in modo generico, ma esegue procedure formalizzate attraverso istruzioni, condizioni e operazioni definite in modo sufficientemente preciso.

Negli ultimi anni, il pensiero computazionale (Computational Thinking, CT) ha assunto un ruolo sempre più importante nell'educazione, dalla scuola dell'infanzia fino all'istruzione superiore. La sua diffusione è legata alla crescente necessità di sviluppare negli studenti competenze utili non solo nell'informatica, ma anche nella risoluzione di problemi complessi e nell'apprendimento di diverse discipline.

Il concetto di pensiero computazionale è stato introdotto inizialmente da Seymour Papert (matematico e informatico), che collegò la programmazione allo sviluppo di un particolare modo di ragionare. A partire da queste prime riflessioni, il concetto si è progressivamente ampliato e oggi non esiste una definizione unica e universalmente condivisa di CT. Inizialmente, infatti, esso era strettamente associato alla programmazione e all'informatica;

successivamente è stato interpretato in modo più ampio, come un insieme di strategie cognitive e pratiche utilizzabili per affrontare problemi anche al di fuori dell'ambito informatico.

In termini generali, il pensiero computazionale può essere definito come un processo di pensiero e di risoluzione dei problemi che consiste nel selezionare e applicare strumenti, strategie e pratiche adeguate per affrontare un problema in modo efficace ed efficiente.

Il collegamento con le nozioni di logica e algoritmi introdotte precedentemente è diretto. Pensare in modo computazionale significa infatti analizzare un problema, comprenderne la struttura, individuare le condizioni rilevanti, suddividerlo in parti più semplici, selezionare le informazioni necessarie, stabilire relazioni logiche tra esse, elaborare una procedura ordinata per arrivare alla soluzione e verificare e, se necessario, correggere il risultato.

Il CT non coincide quindi con la semplice capacità di programmare. La programmazione rappresenta infatti uno degli strumenti attraverso cui il pensiero computazionale può essere sviluppato e applicato, ma il CT comprende competenze più generali e trasferibili a diversi contesti. Per questo motivo viene considerato una competenza trasversale, che può essere utilizzata nella vita quotidiana e in numerosi ambiti disciplinari, dalle STEM alle scienze sociali e alle discipline umanistiche. In questo senso, la logica costituisce uno degli elementi di base del CT, mentre l'algoritmo rappresenta uno dei modi attraverso cui il ragionamento può essere formalizzato. Tuttavia, il pensiero computazionale va oltre la semplice costruzione di algoritmi: comprende un insieme più ampio di pratiche finalizzate alla comprensione, rappresentazione, analisi e risoluzione dei problemi.

Nel corso degli anni sono state proposte diverse definizioni e classificazioni delle competenze che costituiscono il CT. Alcuni studiosi lo hanno descritto attraverso concetti propri dell'informatica e della programmazione, come sequenze, cicli, istruzioni condizionali, debugging e riutilizzo del codice. Altri hanno invece posto maggiore attenzione alle abilità generali di problem-solving.

Tra le principali pratiche associate al CT troviamo:

- scomposizione, cioè suddividere un problema complesso in problemi più semplici e gestibili;
- astrazione, ovvero concentrarsi sugli elementi essenziali di un problema eliminando le informazioni non necessarie;
- formulazione e riformulazione del problema, per renderlo affrontabile attraverso un approccio computazionale;

- pensiero algoritmico, cioè progettare una sequenza ordinata di passaggi necessari per raggiungere una soluzione;
- gestione e analisi dei dati, attraverso la raccolta, organizzazione, rappresentazione e interpretazione delle informazioni;
- automazione, ossia utilizzare strumenti computazionali per eseguire procedure in modo efficiente;
- modellazione e simulazione, per rappresentare e studiare sistemi, fenomeni o situazioni complesse;
- debugging, cioè individuare, analizzare e correggere errori;
- iterazione, attraverso la ripetizione e il miglioramento progressivo delle procedure;
- generalizzazione, che permette di trasferire una soluzione o una strategia a problemi e contesti differenti.

Anche in questo caso è possibile osservare il rapporto con le nozioni logiche e algoritmiche precedentemente introdotte. La scomposizione permette di trasformare un problema complesso in una serie di sottoproblemi; l'astrazione consente di selezionare le informazioni realmente rilevanti; il pensiero algoritmico organizza le operazioni in una sequenza precisa; le condizioni logiche permettono di stabilire percorsi differenti nell'esecuzione; l'iterazione richiama la struttura dei cicli; il debugging introduce un processo sistematico di verifica e correzione; infine, la generalizzazione permette di utilizzare una soluzione già elaborata in situazioni differenti.

Il CT presenta quindi una natura multidimensionale. Comprende innanzitutto una componente cognitiva, costituita dalle conoscenze e dalle abilità necessarie per analizzare e risolvere problemi. Tuttavia, il pensiero computazionale comprende anche una componente non cognitiva, legata agli atteggiamenti e alle disposizioni che permettono di affrontare efficacemente situazioni problematiche. Tra queste rientrano la sicurezza nell'affrontare problemi complessi, la perseveranza di fronte alle difficoltà, la tolleranza dell'ambiguità, la capacità di gestire problemi aperti e la disponibilità a comunicare e collaborare con gli altri.

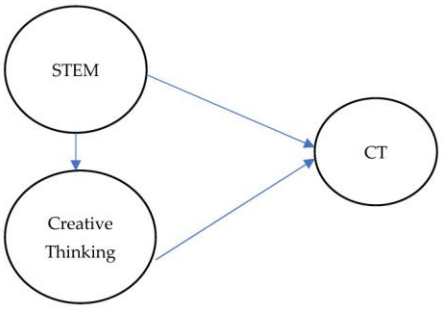
Il valore del CT risiede dunque nella sua trasferibilità. Sviluppare queste competenze non significa soltanto acquisire conoscenze informatiche, ma imparare un modo strutturato di affrontare i problemi che può essere applicato a numerosi contesti. Il pensiero computazionale può quindi contribuire allo sviluppo del pensiero analitico, della creatività, del pensiero critico, della capacità di gestione dei dati e della collaborazione.

La crescente importanza del CT ha portato alla sua integrazione nell'educazione STEM, dando origine all'approccio CT-STEM. Esso consiste nell'integrazione delle pratiche del pensiero computazionale con i contenuti e i contesti di apprendimento delle discipline STEM, con l'obiettivo di sviluppare contemporaneamente le competenze di CT e la comprensione dei contenuti disciplinari. L'integrazione tra CT e STEM risponde anche alla necessità di preparare gli studenti ai nuovi ambiti interdisciplinari presenti nel mondo professionale. Il CT-STEM persegue quindi un duplice obiettivo: da un lato sviluppare le competenze di pensiero computazionale, dall'altro migliorare l'apprendimento dei contenuti STEM.

Possiamo individuare principalmente due modalità di integrazione. La prima considera il CT come una competenza trasversale e utilizza le discipline STEM come contesto significativo nel quale applicare strategie di problem-solving. Gli studenti imparano così a formulare problemi in termini computazionali, elaborare dati, automatizzare procedure, costruire modelli e simulazioni, valutare le soluzioni e generalizzarle. La seconda modalità integra invece specifiche pratiche computazionali direttamente nell'apprendimento delle singole discipline. Attività come gestione dei dati, modellizzazione, simulazione e programmazione possono aiutare gli studenti a rappresentare fenomeni complessi, formulare previsioni, verificare ipotesi e comprendere meglio sistemi e meccanismi scientifici. In questo caso, il CT contribuisce soprattutto ad approfondire i contenuti disciplinari e a sviluppare forme di pensiero computazionale specifiche per una determinata materia.

Per applicare il CT-STEM sono state sperimentate diverse strategie didattiche, tra cui l'apprendimento basato su problemi e progetti, la progettazione di giochi, la modellazione e simulazione computazionale, la robotica e le attività collaborative. Un ruolo importante è svolto anche dallo scaffolding, utilizzato per ridurre il carico cognitivo e sostenere l'apprendimento degli studenti. Esso può assumere la forma di guida diretta, supporto tra pari, risorse didattiche o interventi adattivi. Le strategie collaborative, inoltre, favoriscono lo sviluppo delle capacità di comunicazione e cooperazione.

La valutazione degli interventi CT-STEM riguarda sia la sfera cognitiva, comprendente conoscenze, abilità di CT e STEM, pensiero critico e creatività, sia la sfera non cognitiva, che include autoefficacia, interesse, atteggiamenti e capacità di collaborazione.

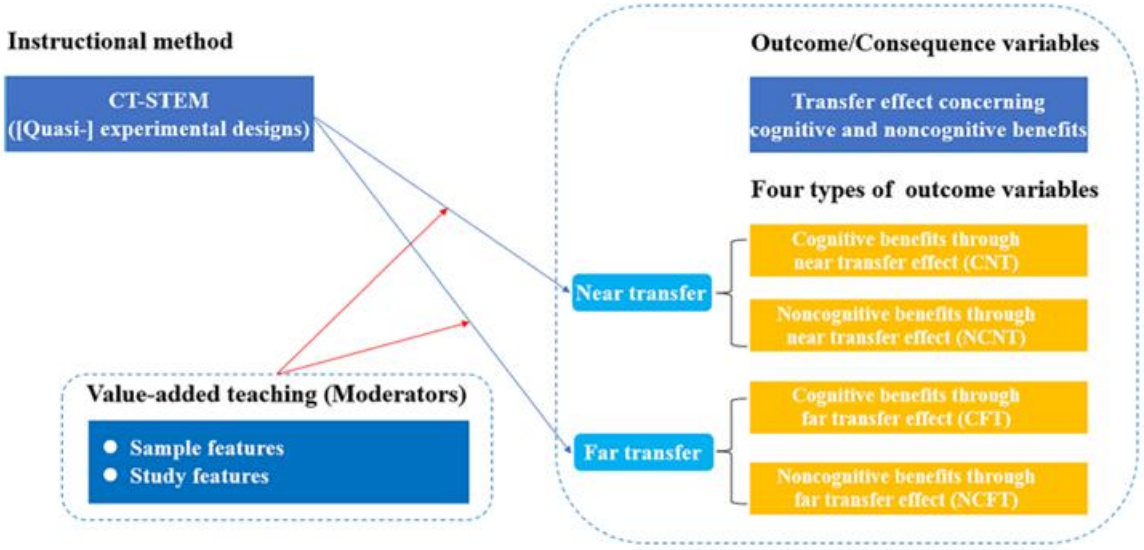


Modello teorico. Suherman, S., Vidákovich, T., Mujib, M., Hidayatulloh, H., Andari, T., & Susanti, V. D. (2025). The Role of STEM Teaching in Education: An Empirical Study to Enhance Creativity and Computational Thinking. Journal of Intelligence, 13(7), 88. https://doi.org/10.3390/jintelligence13070088

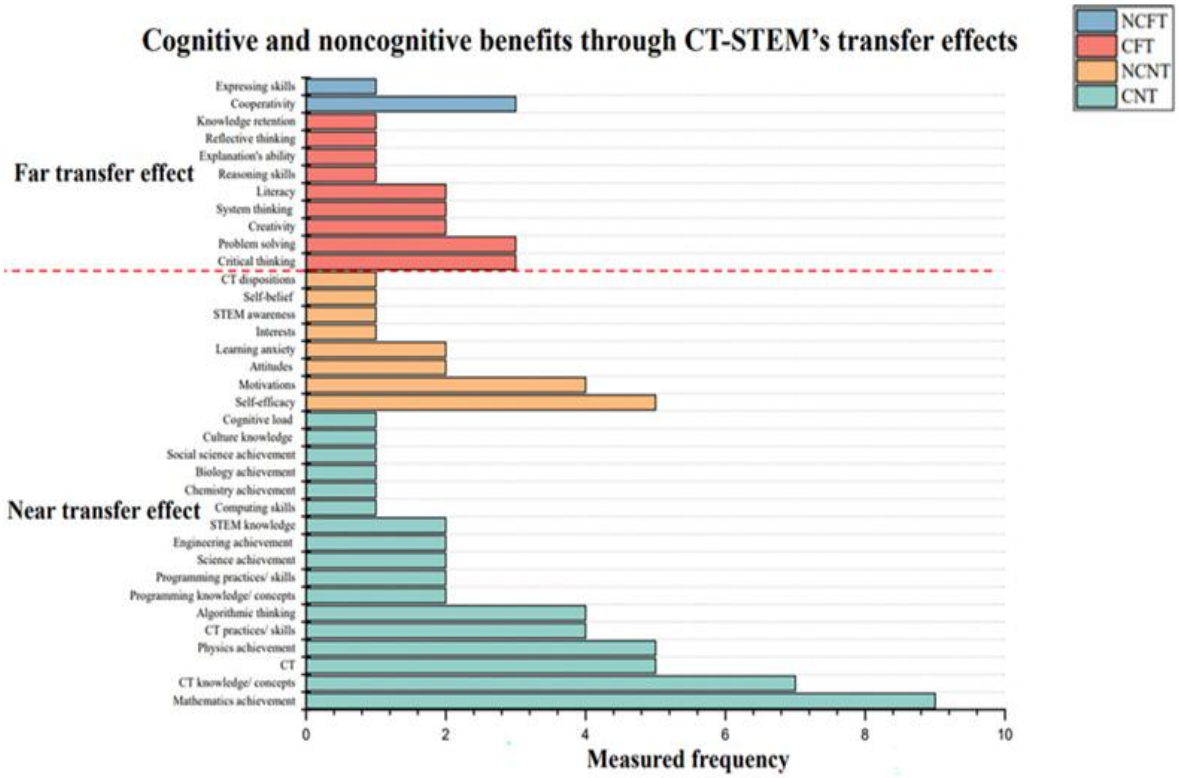
La meta-analisi presa in considerazione ha analizzato i benefici del CT-STEM distinguendo tra trasferimento prossimo (near-transfer), cioè l'applicazione delle competenze a contesti simili a quelli di apprendimento, e trasferimento remoto (far-transfer), che riguarda l'applicazione delle competenze in contesti più lontani da quello originario.

Sono stati inoltre distinti quattro tipi di benefici:

- benefici cognitivi derivanti da trasferimento prossimo;
- benefici non cognitivi derivanti da trasferimento prossimo;
- benefici cognitivi derivanti da trasferimento remoto;
- benefici non cognitivi derivanti da trasferimento remoto.



Conceptual framework of the present meta-analysis. Li, Z., Oon, P.T. (2024) The transfer effect of computational thinking (CT)-STEM: a systematic literature review and meta-analysis. International Journal of STEM Education 11, 44. https://doi.org/10.1186/s40594-024-00498-z



The measured frequency of documented benefits in selected publications. Li, Z., Oon, P.T. (2024) The transfer effect of computational thinking (CT)-STEM: a systematic literature review and meta-analysis. International Journal of STEM Education 11, 44. https://doi.org/10.1186/s40594-024-00498-z

I risultati mostrano effetti positivi e di entità moderata sia sui benefici cognitivi sia su quelli non cognitivi. In particolare, gli effetti del trasferimento prossimo risultano generalmente più forti di quelli del trasferimento remoto, mentre i benefici cognitivi mostrano dimensioni dell'effetto superiori rispetto a quelli non cognitivi.

Tra i fattori che possono influenzare l'efficacia del CT-STEM vengono considerati il livello di istruzione, la dimensione del campione o della classe, il disegno dello studio, la disciplina coinvolta, la strategia didattica e la durata dell'intervento. Si evidenzia che gli effetti sono stati più marcati negli studenti della scuola dell'obbligo rispetto agli universitari, suggerendo l'importanza di introdurre il CT precocemente. Le classi più piccole, inferiori a 50 studenti, mostrano effetti di trasferimento maggiori. Le attività basate su modellazione computazionale, simulazione e apprendimento incarnato hanno prodotto risultati particolarmente positivi. Gli interventi con una durata compresa tra una settimana e un mese sono risultati generalmente più efficaci, mentre durate più lunghe sembrano associate a una riduzione dell'effetto, probabilmente anche a causa di un

possibile calo di interesse e motivazione. Nel trasferimento remoto, il livello di istruzione e la dimensione del campione sono risultati moderatori significativi.

L'approccio CT-STEM permette di collegare diverse competenze ai contenuti disciplinari, favorendo sia lo sviluppo del pensiero computazionale sia una maggiore comprensione delle discipline STEM. In particolare, la modellazione computazionale rappresenta un punto di accesso privilegiato per integrare il CT nella didattica, poiché consente agli studenti di rappresentare e simulare fenomeni complessi, formulare previsioni e verificare ipotesi.

Pertanto, l'integrazione del CT dovrebbe essere promossa a diversi livelli scolastici e possibilmente fin dalle prime fasi dell'istruzione, prestando attenzione alla dimensione delle classi, alla durata degli interventi e alla scelta di strategie didattiche attive, soprattutto modellizzazione e simulazione.

Il collegamento tra logica, algoritmi, pensiero computazionale e discipline STEM assume un'importanza particolare nell'ambito delle biotecnologie, dove la complessità dei sistemi biologici e la quantità crescente di dati prodotti dalle tecnologie moderne richiedono l'impiego di strumenti e strategie computazionali.

L’informatica ha infatti assunto un ruolo sempre più centrale nell’ambito della biologia applicata e delle biotecnologie, analogamente a quanto avvenuto in numerosi altri settori della società contemporanea. L’integrazione tra metodologie computazionali e discipline biologiche ha dato origine a un insieme articolato di approcci e applicazioni che costituiscono una componente fondamentale della ricerca biotecnologica moderna.

In questo contesto, la biologia computazionale e, in particolare, la bioinformatica rappresentano alcune delle espressioni più significative dell’interazione tra informatica e scienze della vita, soprattutto per quanto riguarda la gestione, l’analisi e l’interpretazione delle ingenti quantità di dati generate dalle tecnologie “omiche”.

Il collegamento con il pensiero computazionale è evidente: di fronte a grandi quantità di dati biologici è necessario innanzitutto scomporre il problema, individuare quali informazioni siano rilevanti, organizzarle e rappresentarle in modo appropriato. Successivamente è possibile applicare procedure algoritmiche per analizzare i dati, identificare regolarità e pattern, formulare modelli e verificare ipotesi. L'astrazione permette di concentrarsi sulle caratteristiche biologiche più significative, mentre l'automazione consente di eseguire in modo efficiente procedure che, quando applicate a grandi quantità di dati, sarebbero difficilmente gestibili manualmente.

L’evoluzione delle tecnologie di sequenziamento, comprese quelle comunemente definite di nuova generazione o di seconda generazione, ha ulteriormente ampliato le possibilità di applicazione dei metodi computazionali alla ricerca biologica. Un esempio particolarmente significativo è rappresentato dalla genomica delle colture, ambito caratterizzato da una notevole complessità sia dal punto di vista biologico sia da quello computazionale.

In tale prospettiva, l’integrazione di dati provenienti da differenti discipline omiche con quelli ottenuti mediante tecnologie di sequenziamento di nuova generazione consente di identificare e caratterizzare elementi genetici associati a organismi geneticamente modificati presenti nelle filiere alimentari e mangimistiche. Questi approcci assumono una rilevanza particolare anche in relazione alla crescente necessità di sviluppare strumenti scientifici e computazionali adeguati al monitoraggio e alla caratterizzazione dei prodotti biotecnologici.

Il lavoro con questi dati mostra concretamente come le pratiche del CT possano essere applicate alla ricerca biologica. La raccolta e l'organizzazione delle informazioni richiamano la gestione dei dati; l'individuazione di caratteristiche ricorrenti richiama il riconoscimento di pattern; la selezione delle informazioni rilevanti implica un processo di astrazione; l'analisi mediante procedure automatizzate richiede la progettazione e l'esecuzione di algoritmi; la verifica dei risultati e la correzione di eventuali errori costituiscono forme di debugging e iterazione.

Un ulteriore elemento caratterizzante della ricerca contemporanea nelle biotecnologie è rappresentato dall’impiego del cloud computing e del Web computing per affrontare problemi biologici complessi. Tali tecnologie possono essere utilizzate sia come infrastrutture per l’esecuzione di processi di calcolo parallelo e distribuito, sia come strumenti finalizzati a rendere disponibili flussi di lavoro bioinformatici a ricercatori e professionisti che non dispongono direttamente di risorse di calcolo ad alte prestazioni. In questo quadro si colloca anche FoldIt, uno dei primi esempi di applicazione del calcolo decentralizzato allo studio delle strutture proteiche e delle relazioni tra struttura e funzione, con importanti ricadute nel campo della biologia computazionale e delle biotecnologie.

L'impiego dell'informatica nelle scienze della vita, tuttavia, non si limita all'analisi dei dati prodotti dalle discipline omiche. Gli approcci computazionali vengono infatti applicati alla modellizzazione dei sistemi biologici a differenti livelli di organizzazione, dalle interazioni tra singole molecole fino alla descrizione dei processi metabolici a livello dell'intero organismo.

La modellizzazione rappresenta un ulteriore collegamento particolarmente importante con il pensiero computazionale. Un sistema biologico complesso può essere scomposto nelle sue componenti principali e successivamente rappresentato attraverso un modello che considera le relazioni più rilevanti tra tali componenti. Il modello può quindi essere utilizzato per formulare ipotesi, effettuare simulazioni, produrre previsioni e confrontare i risultati ottenuti con dati sperimentali.

In questo senso, modellizzazione e simulazione rappresentano applicazioni dirette del CT: permettono infatti di trasformare un fenomeno complesso in una rappresentazione formalizzata che può essere analizzata attraverso procedure computazionali.

Parallelamente, lo sviluppo di browser Web e di nuovi standard HTML ha reso possibile la visualizzazione e l'esplorazione di strutture biomolecolari senza la necessità di ricorrere necessariamente a software specialistici. Queste tecnologie costituiscono strumenti abilitanti anche per attività quali la progettazione razionale di farmaci, nelle quali l'interpretazione computazionale delle strutture molecolari assume un ruolo rilevante.

L'evoluzione di tali applicazioni può essere interpretata anche alla luce del paradigma del computational thinking, inteso come un approccio alla risoluzione dei problemi basato sulla loro scomposizione, sull'identificazione di pattern e relazioni, sull'astrazione delle informazioni rilevanti e sulla definizione di procedure e algoritmi capaci di automatizzare o supportare i processi di analisi.

Applicato alle biotecnologie, il computational thinking favorisce il passaggio da una concezione dell'informatica come semplice strumento di elaborazione dei dati a una prospettiva nella quale il ragionamento computazionale diventa parte integrante della progettazione della ricerca biologica. La complessità dei sistemi biologici e la crescente disponibilità di dati richiedono infatti competenze capaci di integrare conoscenze biologiche, capacità di formalizzazione dei problemi e strategie computazionali. Il ricercatore deve essere in grado non soltanto di utilizzare un software già esistente, ma anche di comprendere come un problema biologico possa essere tradotto in termini analizzabili computazionalmente.

In questa prospettiva, la portata dell'informatica e della modellizzazione nelle biotecnologie risulta estremamente ampia e non può essere ricondotta esclusivamente alla bioinformatica tradizionale. Gli strumenti computazionali trovano applicazione nella progettazione e nell'interpretazione di esperimenti complessi di biologia dei sistemi, nello studio del destino e della differenziazione delle cellule staminali,

nell'immunosequenziamento e nella caratterizzazione del repertorio del sistema immunitario adattativo, nonché in numerosi altri ambiti della biologia applicata.

L'integrazione del CT nelle discipline STEM permette di applicare queste strategie a problemi scientifici autentici, favorendo contemporaneamente lo sviluppo delle competenze computazionali e la comprensione dei contenuti disciplinari. In particolare, la modellizzazione e la simulazione permettono di rappresentare sistemi complessi, formulare previsioni e verificare ipotesi.

Nelle biotecnologie, questo approccio assume un'importanza ancora maggiore. La quantità e la complessità dei dati biologici, la necessità di integrare differenti fonti di informazione e la possibilità di costruire modelli dei sistemi biologici richiedono infatti un'interazione sempre più stretta tra competenze biologiche, biotecnologiche e computazionali.

Il progressivo ampliamento di queste applicazioni evidenzia quindi una trasformazione significativa del rapporto tra informatica, biologia e biotecnologie. Gli approcci computazionali non rappresentano più esclusivamente strumenti destinati all'elaborazione di dati biologici, ma partecipano sempre più direttamente alla formulazione delle domande di ricerca, alla progettazione sperimentale, alla costruzione di modelli predittivi e all'interpretazione dei fenomeni biologici.

L'integrazione tra logica, algoritmi, programmazione, pensiero computazionale, discipline STEM e biotecnologie mostra quindi come il ragionamento computazionale non debba essere considerato soltanto una competenza informatica, ma come un metodo generale per affrontare problemi complessi. Dalla semplice valutazione di una condizione logica fino alla costruzione di modelli biologici e all'analisi di grandi quantità di dati genomici, il medesimo principio di fondo rimane quello di rendere un problema comprensibile, strutturabile, analizzabile e, quando possibile, risolvibile attraverso procedure sistematiche.

Logica computazionale nelle STEM e la sua applicazione nelle biotecnologie

Il pensiero computazionale (computational thinking, CT) è fondamentale per molte, se non per tutte, le professioni, in particolare nei settori della scienza, della tecnologia, dell'ingegneria e della matematica (STEM). Queste discipline si avvalgono infatti di strumenti computazionali per la modellazione e la simulazione, per l'analisi e la visualizzazione dei dati, per la creazione di programmi informatici finalizzati alla risoluzione di problemi e per la comprensione dei sistemi come aggregati di parti. Tutte queste attività sono caratteristiche del pensiero computazionale, del quale si può dunque dedurre il carattere intrinseco alle pratiche STEM.

Pur in assenza di un consenso unanime sulla sua definizione, la letteratura concorda sul fatto che il CT implichi scomposizione, astrazione, riconoscimento di schemi (pattern) e pensiero algoritmico. Dopo aver acceso il dibattito sulla sua importanza nel 2006, dieci anni più tardi Jeannette Wing (professoressa di informatica e attuale vicepresidente esecutivo per la ricerca nella Columbia University) lo ha definito come:

«… i processi di pensiero coinvolti nella formulazione di problemi e delle relative soluzioni, in modo tale che queste ultime siano rappresentate in una forma efficacemente eseguibile da un agente in grado di elaborare informazioni».

I partecipanti ai progetti STEM sono fortemente motivati alla loro realizzazione. Una caratteristica distintiva di tali progetti è l'orientamento tecnico, insieme all'integrazione di diverse discipline didattiche e aree scientifiche. In questo contesto, l'analisi intelligente dei dati, e in particolare l'analisi intelligente delle immagini, consente di automatizzare numerose attività nei settori industriale, scientifico, dei trasporti e in altri ambiti. Nei sistemi di trasporto, ad esempio, essa viene impiegata per riconoscere i veicoli e i loro componenti, per la diagnostica tecnica, per il monitoraggio della posizione e della velocità, nonché a fini didattici. Sussiste attualmente l'esigenza di sviluppare sistemi informatici per analizzare lo stato del materiale rotabile ferroviario, ovvero di locomotive e vagoni.

A tal fine esistono oggi numerosi metodi di segmentazione, tra cui il metodo di crescita delle regioni (region-growing) e il metodo dello spartiacque (watershed). Quest'ultimo si rivela particolarmente efficace nella segmentazione di immagini caratterizzate da contorni ben definiti: secondo tale metodo, i contorni vengono dapprima evidenziati nell'immagine e successivamente si individuano i confini dei segmenti tenendo conto di essi.

I principi già descritti (in particolare la scomposizione dei sistemi in elementi, l’astrazione e il ragionamento algoritmico) trovano un’applicazione rilevante nella biologia sintetica. In questo ambito, le cellule possono essere progettate come sistemi in grado di elaborare

informazioni, grazie a circuiti genetici che realizzano operazioni riconducibili alla logica digitale o analogica.

La biologia sintetica si propone di progettare, ingegnerizzare e riprogettare sistemi biologici per impieghi concreti, tra cui la bioproduzione, il biosensing e la bioterapia. Il processo segue generalmente il ciclo di progettazione, realizzazione e verifica (design-build-test). Sulla base di concetti mutuati dall'informatica e dall'elettronica, sono stati sviluppati circuiti genetici sintetici capaci di dirigere il flusso informativo nei sistemi biologici. Tali circuiti ricorrono soprattutto a due modalità di calcolo: quella digitale, fondata sulla logica booleana, e quella analogica, basata su livelli di segnale graduati. La progettazione si fonda su principi quali modularità, ortogonalità, prevedibilità e affidabilità. Se i primi circuiti erano relativamente semplici, anche per la limitata disponibilità di componenti modulari e ortogonali, negli ultimi anni il repertorio di elementi utilizzabili si è ampliato sensibilmente. Parallelamente, lo sviluppo di tecniche ad alto rendimento per l'assemblaggio e la caratterizzazione del DNA ha reso possibili la prototipazione rapida, la valutazione sistematica *in situ* e l'automazione della progettazione e dell'assemblaggio. Tra le direzioni di ricerca figurano inoltre la realizzazione di circuiti dotati di memoria e il calcolo distribuito attraverso consorzi cellulari.

Per mantenersi vitali e riprodursi, le cellule devono riconoscere numerosi segnali provenienti dall'ambiente esterno o generati al loro interno. In risposta a tali stimoli, attivano specifiche funzioni di uscita (*output*), che possono comprendere cambiamenti nella morfologia cellulare oppure la sintesi di proteine e piccole molecole. Queste risposte possono essere impiegate per obiettivi applicativi, come la produzione di composti di valore o l'individuazione di tossine pericolose presenti nell'ambiente. Poiché molte di queste proprietà hanno un'utilità concreta, la possibilità di modificarle e controllarle rappresenta un'opportunità importante.

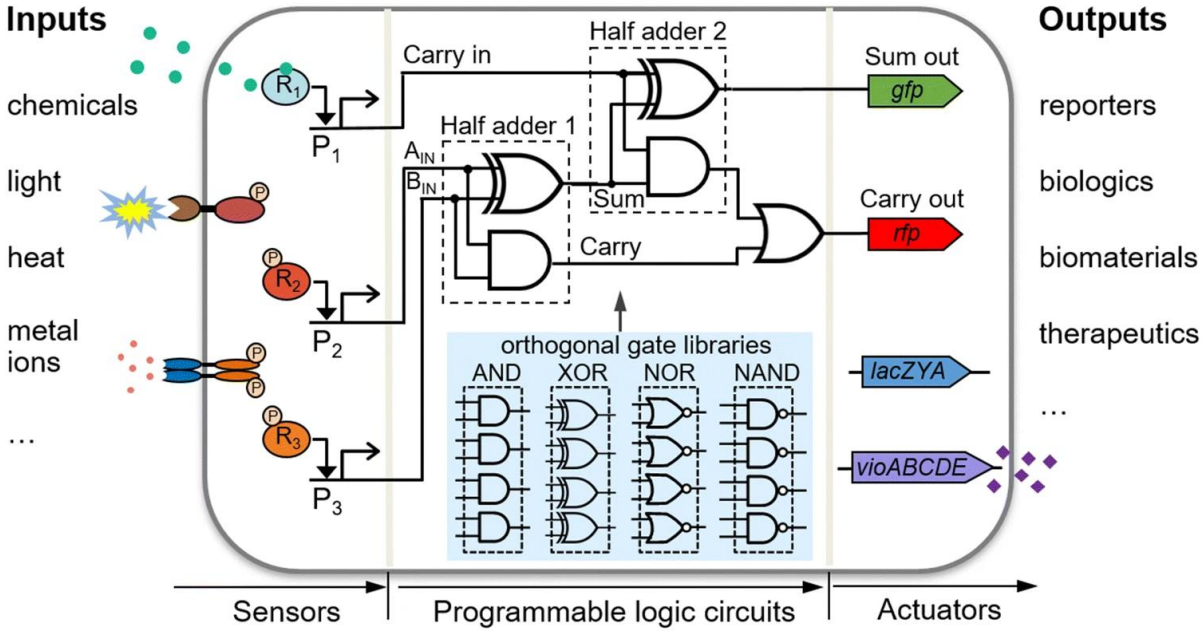
In passato, i flussi di lavoro biotecnologici sono stati spesso organizzati attorno a soluzioni progettate su misura per singole esigenze. Un simile approccio tende a richiedere procedure di laboratorio laboriose e difficilmente trasferibili da un problema a un altro, limitando anche il riutilizzo delle conoscenze acquisite. Sebbene non esista una definizione universalmente condivisa di biologia sintetica, le definizioni presenti in letteratura convergono su molti aspetti. Una formulazione la descrive come la progettazione e l'ingegnerizzazione di componenti, dispositivi e sistemi biologici innovativi, insieme alla riprogettazione di sistemi naturali già esistenti. In senso più ampio, può essere intesa come

un approccio razionale alle biotecnologie, ispirato all'ingegneria e orientato a rendere la progettazione dei sistemi biologici più semplice, rapida e prevedibile. Tra i suoi principi fondamentali rientrano standardizzazione, modularità, caratterizzazione e ortogonalità. In questa prospettiva, sono state sviluppate raccolte organizzate di parti biologiche ed è stato adottato in diversi contesti scientifici, industriali e istituzionali. Il ciclo ingegneristico di progettazione, realizzazione, verifica e apprendimento è chiamato DBTL (design-build-test-learn).

Il calcolo cellulare costituisce uno degli ambiti centrali della biologia sintetica: consiste nell'utilizzare le vie di risposta già presenti nelle cellule per realizzare dispositivi biologici capaci di produrre risultati programmabili e prevedibili in funzione di determinati segnali in ingresso (*input*). Una strategia di progettazione prevede l'impiego di circuiti genetici in grado di regolare il comportamento cellulare. Gli approcci principali sono di tipo digitale, fondato su stati binari, oppure analogico. Sono possibili anche altre forme di calcolo basate sul DNA, che utilizzano processi quali piastrellatura, ibridazione, autoassemblaggio o ricombinazione.

I componenti biologici digitali presentano caratteristiche simili a quelle delle porte logiche elettroniche, nelle quali l'informazione è rappresentata da stati binari distinti. Un esempio in elettronica è il sommatore completo a un bit, formato da cinque porte logiche disposte su tre livelli e dotato di tre ingressi e due uscite. In tali sistemi i livelli alti e bassi del segnale sono nettamente separati e corrispondono rispettivamente agli stati ON (acceso) e OFF (spento). Di conseguenza, una variazione contenuta dell'ingresso può provocare un cambiamento brusco dell'uscita quando viene superata una determinata soglia, associata a un coefficiente di Hill superiore a 1, che indica nel legame di un ligando a una proteina o a un enzima una cooperatività negativa. Questi circuiti vengono spesso descritti attraverso la logica booleana, che riconduce i valori a VERO (1) e FALSO (0). Le operazioni, come la funzione AND possono essere rappresentate mediante simboli circuitali derivati dall'elettronica o tabelle di verità, citati anche in un capitolo precedente. Per funzionare correttamente all'interno di un sistema più ampio, i componenti dovrebbero interagire con gli elementi a monte e a valle secondo modalità prevedibili, mostrando quindi modularità. Si desidera, inoltre, che il segnale in uscita sia stabile, poco influenzato dal rumore, ossia privo dalle fluttuazioni casuali indesiderate, e caratterizzato da un elevato rapporto ON:OFF. Queste proprietà contribuiscono a evitare il deterioramento del segnale lungo il circuito. La logica digitale è particolarmente adatta ai processi decisionali, per esempio

quelli coinvolti nel differenziamento cellulare naturale o nell'apoptosi: la transizione netta tra gli stati favorisce decisioni affidabili e l'integrazione di più segnali, mentre la relativa resistenza al rumore migliora la robustezza. Sono state realizzate porte logiche genetiche binarie, tra cui AND, OR e NOR, successivamente combinate in circuiti più articolati, come una porta AND con quattro ingressi.



Calcolo cellulare programmabile con capacità di elaborazione del segnale scalabile. Per ottenere un controllo del comportamento cellulare su larga scala, è necessaria una libreria ampliata di blocchi di regolazione genetica ortogonali e versatili, unitamente ai relativi principi di interconnessione. Ad esempio, un programma genetico che implementa un sommatore completo a 1 bit (full adder) per numeri binari — dotato di 3 ingressi e 2 uscite — può essere realizzato assemblando 5 porte logiche modulari, organizzate su 3 livelli e selezionate da librerie di porte ortogonali ben caratterizzate. Tali circuiti genetici possono essere integrati con sensori genetici di ingresso e attuatori di uscita modulari, consentendo processi decisionali complessi per una varietà di applicazioni di interesse umano. Xiang, Y., Dalchau, N. & Wang, B. (2018). Scaling up genetic circuit design for cellular computing: advances and prospects. Natural Computing 17, 833–853 https://doi.org/10.1007/s11047-018-9715-9

Le risposte analogiche, diversamente da quelle digitali, producono un segnale continuo che cambia in modo graduale in relazione all'intensità dell'ingresso. Per essere adeguate al calcolo analogico, le funzioni di trasferimento dovrebbero essere ben definite e capaci di operare su un ampio intervallo di input, mantenendo al contempo affidabilità, modularità e un livello contenuto di rumore. Questo approccio può utilizzare le risorse cellulari in modo efficiente ed è utile per generare risposte autonome e localizzate a variazioni nel tempo, per i processi di autoregolazione e per l'interazione con il metabolismo dell'organismo ospite.

Di contro, i circuiti analogici sono proprio più esposti agli effetti del rumore e alle perturbazioni transitorie del segnale in ingresso.

Le applicazioni potenziali dei circuiti genetici sono state illustrate mediante diverse dimostrazioni di principio. Tra queste figurano biosensori in grado di rilevare l'arsenico e un circuito classificatore basato su una porta NOR a tre ingressi, progettato per eliminare cellule tumorali. I consorzi cellulari, costituiti da almeno due popolazioni che cooperano per svolgere compiti complessi, sono stati inoltre impiegati per realizzare un convertitore analogico-digitale capace di rilevare odori attraverso recettori olfattivi. In altri casi, microrganismi commensali ingegnerizzati sono stati utilizzati per introdurre funzionalità di memoria.

Negli ultimi anni sono stati sviluppati circuiti genetici di dimensioni maggiori, resi possibili dal miglioramento delle tecniche di caratterizzazione e dall'ampliamento della libreria di componenti funzionali. L'impiego di componenti basati sull'RNA ha consentito di costruire un circuito articolato con dodici ingressi. In cellule di mammifero è stato inoltre dimostrato un circuito di logica booleana con sei ingressi e un'uscita, definito mediante una tabella di verità. Un altro progetto ha integrato quattro ingressi di selezione e due ingressi dati: i primi stabiliscono quale delle sedici funzioni logiche possibili a due variabili debba essere eseguita, permettendo di modificare in tempo reale la logica della cellula. La standardizzazione dei promotori e delle ricombinasi ha consentito di ottenere 113 circuiti, 109 dei quali funzionanti; al momento della pubblicazione, questa collezione rappresentava la più ampia serie di circuiti logici distinti e funzionali realizzati in cellule di mammifero. Un ulteriore esempio è costituito da un sommatore completo a un bit, realizzato in consorzi di cellule di mammifero mediante l'integrazione di 22 porte logiche distinte, distribuite in nove tipi cellulari specializzati all'interno di un ambiente tridimensionale complesso. Nonostante questi progressi, i circuiti genetici su larga scala restano poco comuni perché richiedono una grande quantità di tempo e di lavoro per poter completare ciascun ciclo DBTL.

La costruzione di circuiti estesi richiede una raccolta ampia di porte logiche accuratamente caratterizzate, ortogonali e modulari, che possano fungere da elementi costitutivi del sistema. Diversi strumenti biochimici, adattati a nuove funzioni, hanno già dimostrato di consentire un certo controllo sullo stato cellulare. Molti agiscono sul dogma centrale della biologia, intervenendo sull'espressione genica e sul flusso dell'informazione dal DNA all'RNA e, successivamente, alle proteine. In base al livello su cui operano, possono essere

distinti in tre categorie principali: regolazione della trascrizione del DNA, regolazione della traduzione dell'RNA messaggero (mRNA) e modulazione delle interazioni tra proteine.

La regolazione mediata dall'RNA comporta generalmente un onere metabolico inferiore per la cellula ospite. Può essere ottenuta, per esempio, mediante il riconoscimento di coppie di sequenze bersaglio destinate al taglio oppure attraverso la formazione di strutture secondarie dell'RNA. Anche le interazioni proteina-proteina costituiscono una risorsa per la progettazione di circuiti: quelle presenti nel sistema di regolazione genica hrp, associato alla risposta di ipersensibilità e patogenicità, possono essere sfruttate per realizzare porte logiche genetiche flessibili e dotate di più ingressi. Gli inteini, sequenze amminoacidiche capaci di autoescindersi e di favorire la riunione delle porzioni proteiche rimanenti mediante un legame chimico, possono contribuire all'integrazione dei segnali, sia nella loro forma naturale sia quando vengono suddivisi artificialmente. È inoltre possibile fondere proteine diverse per combinare in un singolo elemento proprietà differenti.

L'ampliamento dei componenti disponibili è stato favorito da numerosi strumenti sperimentali e computazionali. Le tecniche di genome mining, per esempio, hanno permesso di ottenere una libreria contenente sedici repressori ortogonali appartenenti alla famiglia TetR, caratterizzati da una marcata attività repressiva, quattro polimerasi T7 e venti fattori sigma. Sono stati inoltre caratterizzati 582 terminatori. Un approccio di ingegneria proteica, basato sull'integrazione di metodi bioinformatici e mutagenesi sito-diretta, ha consentito di generare differenti specificità di legame al DNA all'interno della famiglia delle proteine recettrici dell'AMP ciclico (CRP). In prospettiva, componenti ortogonali in numero ancora maggiore potrebbero derivare da proteine di legame al DNA progettate in modo prevedibile e da strumenti fondati sulle strutture secondarie dell'RNA.

Oltre ai componenti che esercitano un controllo diretto, sono disponibili diversi parametri regolatori che permettono di modulare la dinamica del circuito, migliorando la risposta agli ingressi previsti e la generazione delle uscite desiderate. La cooperatività, per esempio, è stata incrementata attraverso l'impiego di domini di oligomerizzazione. Nei circuiti digitali, l'introduzione di feedback positivi e di cascate di segnalazione ha contribuito a migliorare il rapporto ON/OFF, che può risultare limitato dall'espressione genica basale non desiderata presente anche in assenza di attivatori o in presenza di repressori, questa è detta leaky,

Nel ciclo DBTL, dopo la costruzione del circuito si passa alla fase di test, durante la quale se ne misurano quantitativamente le proprietà e le dinamiche. I risultati ottenuti orientano le successive fasi di progettazione e permettono di individuare e correggere eventuali malfunzionamenti, secondo un processo analogo al debugging informatico. La possibilità di caratterizzare sistemi su larga scala dipende soprattutto dall'esecuzione simultanea di molti esperimenti e dalla raccolta di misure accurate su diversi parametri dei campioni, così da ridurre il tempo necessario a completare il ciclo di sviluppo.

La caratterizzazione dei circuiti si è basata tradizionalmente su geni reporter fluorescenti, tra cui quelli che codificano la proteina fluorescente verde (GFP) e la proteina fluorescente rossa (RFP). L'intensità della fluorescenza consente di quantificare l'espressione genica mediante lettori di piastre o microscopi a fluorescenza. Le stesse proteine possono essere impiegate per valutare sequenze regolatrici oppure fuse ad altre proteine, in modo da studiarne localizzazione e interazioni attraverso il trasferimento di energia per risonanza di Förster (FRET). La citometria a flusso ha ulteriormente ampliato queste possibilità, consentendo di misurare contemporaneamente, su singole cellule trascinate in un flusso liquido e attraversate da più fasci laser, parametri quali dimensioni, granularità e fluorescenza.

Le tecniche ad alto rendimento (high-throughput) consentono di condurre numerosi esperimenti in parallelo e di raccogliere molti più dati in tempi ridotti. A queste si affiancano le analisi su scala globale del genoma e del proteoma, che forniscono una descrizione complessiva dello stato cellulare, difficilmente ottenibile in modo pratico attraverso la sola misurazione dell'espressione di singoli geni.

L'automazione della progettazione può accelerare in modo significativo lo sviluppo di sistemi biologici. I software dedicati possono mettere a disposizione dei progettisti le conoscenze disponibili sui componenti, combinarli in circuiti, pianificare gli esperimenti e archiviare e visualizzare i dati raccolti; in prospettiva, possono anche prevedere il comportamento del sistema. Durante la costruzione, un programma potrebbe definire le operazioni necessarie per assemblare fisicamente la sequenza di DNA a partire dai materiali di partenza e fornire il relativo protocollo sperimentale. Per la fase di verifica, potrebbe invece supportare la pianificazione degli esperimenti e la simulazione di un modello computazionale. In questo modo sarebbe possibile ottenere indicazioni preliminari sul comportamento del circuito, limitando il ricorso a prove costose e lunghe e individuando gli esperimenti più informativi per le decisioni successive.

Tra i software per la progettazione automatizzata dei circuiti genetici, Cello consente di definire circuiti destinati a svolgere specifiche funzioni logiche. La descrizione funzionale viene espressa mediante il linguaggio di programmazione logica Verilog, mentre un file di vincoli stabiliti dall'utente specifica i componenti e gli organismi da utilizzare, delimitando lo spazio delle soluzioni possibili. Il sistema rielabora quindi il circuito per adattarlo a una libreria di porte NOR e NOT basate sulla repressione genica. Le prestazioni vengono poi simulate considerando elementi quali il carico del circuito, la variabilità della popolazione, la crescita cellulare e la connettività legata al flusso della RNA polimerasi (RNAP).

Una difficoltà rilevante per i software di progettazione automatizzata consiste nel costruire modelli matematici capaci di prevedere con affidabilità il comportamento effettivo dei circuiti.

Un'ulteriore criticità deriva dal carattere discreto e casuale delle reazioni chimiche. Nei circuiti in cui è rilevante soltanto distinguere uno stato genico attivo da uno inattivo, senza considerare i livelli intermedi di espressione, anche modelli booleani più semplici possono risultare efficaci.

La memoria cellulare si fonda sulla capacità di conservare nel tempo modificazioni indotte da segnali temporanei, di norma di natura genetica. In questo modo diventa possibile realizzare una logica sequenziale, che amplia le possibilità della logica combinatoria tradizionale. Tra i primi dispositivi studiati figura l'interruttore bistabile, capace di mantenere due stati stabili di espressione genica. L'informazione memorizzata può essere recuperata mediante sequenziamento; in alternativa, si possono progettare elementi genetici (per esempio promotori e sequenze codificanti) che vengono invertiti per portarli dentro o fuori dall'orientamento corretto per la trascrizione, integrando così una componente sequenziale nel circuito. Utilizzando ricombinasi collocate ai lati di elementi come promotori e terminatori, sono state implementate sedici funzioni booleane a due ingressi, con una memoria stabile per novanta generazioni cellulari. In uno studio successivo è stata raggiunta una capacità di memorizzazione pari a 1 byte.

Il sistema CRISPR (Clustered Regularly Interspaced Short Palindromic Repeats) e le proteine associate Cas consentono, in natura, di conservare nel genoma frammenti di DNA provenienti da organismi invasori, contribuendo alla risposta immunitaria. Lo stesso principio è stato adattato alla registrazione di informazioni: associando determinati colori a sequenze specifiche e distribuendo il sistema in un'ampia popolazione cellulare, è stato realizzato un dispositivo elementare per registrare immagini, con una capacità di 2,6

kilobyte e dati recuperabili tramite sequenziamento. Strategie analoghe hanno permesso di registrare la presenza di metaboliti specifici inducendo, in loro presenza, la produzione di un DNA trigger. Altri approcci si sono basati sulla modifica della sequenza di plasmidi reporter e sulla misurazione del rapporto tra plasmidi modificati e non modificati. È stata inoltre sfruttata la metilazione del DNA come indicatore di eventi: nello stato OFF, una proteina zinc finger impedisce il processo; in seguito a induzione termica, la metilazione viene avviata e ostacola il legame della proteina, rendendo possibile un'espressione persistente.

L'ingegneria metabolica mira a intervenire in modo mirato sul metabolismo cellulare per aumentarne la capacità produttiva. Considerata la complessità delle reti metaboliche, i circuiti genetici a feedback rappresentano strumenti promettenti per contenere l'eterogeneità metabolica, sostenere la cooperazione tra microrganismi e migliorare la produttività cellulare. Un approccio consiste nell'introdurre meccanismi di ricompensa e penalizzazione che favoriscano le cellule più efficienti e contrastino i cosiddetti cheater, ossia le cellule che beneficiano delle risorse comuni senza contribuire adeguatamente alla produzione. L'utilizzo di fattori di trascrizione sensibili a specifici metaboliti consente di riconfigurare l'espressione genica e di collegare l'attività dei circuiti di feedback alla fitness di crescita, rendendo possibile un controllo dinamico della popolazione. In quest'ottica, i circuiti genetici a feedback e le interazioni microbiche potrebbero assumere un ruolo centrale negli sviluppi futuri del settore. Le strategie di progettazione comprendono il controllo del carico metabolico per limitare lo stress cellulare, il monitoraggio della qualità della popolazione per contenere le cellule parassite, l'impiego della dipendenza dal prodotto per avvantaggiare le cellule produttive e l'uso di una doppia regolazione dinamica per separare la crescita dalla sintesi del prodotto. Nel loro insieme, queste strategie possono migliorare le prestazioni della comunità cellulare. La codifica di funzioni decisionali e la riprogrammazione del comportamento collettivo potrebbero rendere possibile lo sviluppo di fabbriche cellulari più robuste e di sistemi di bioproduzione intelligente. Diversi meccanismi regolatori e circuiti genetico-metabolici potrebbero inoltre essere sfruttati per ottenere funzioni metaboliche autonome o capaci di adattarsi alle condizioni, con possibili applicazioni biotecnologiche e mediche. L'adozione di questi principi potrebbe infine affiancare le strategie tradizionali di ingegneria dei bioprocessi e contribuire a ridurre i costi della fermentazione industriale.

I progressi dell'ingegneria genetica hanno permesso di regolare con maggiore precisione l'espressione dei geni e di riprogrammare il metabolismo, ottenendo fabbriche cellulari progettate su misura e caratterizzate da migliori valori di resa e produttività, indicati dall'indice TYP. Attraverso l'introduzione di vie metaboliche eterologhe o la modifica di quelle endogene, è possibile convertire materie prime a basso costo in numerosi prodotti: sostanze chimiche di base, nuovi materiali, biocarburanti avanzati, nutraceutici e farmaci. Tra le strategie più consolidate rientrano la sovraespressione degli enzimi che catalizzano le fasi limitanti della velocità di reazione, l'eliminazione delle vie metaboliche concorrenti, la gestione dell'ATP e il riciclo del NADPH e di altri cofattori.

Se una modifica non apporta un vantaggio in termini di fitness, essa può imporre alla cellula un costo energetico supplementare e ridurre la resa della via metabolica. Poiché i processi evolutivi tendono a favorire la proliferazione cellulare, gli interessi della cellula possono entrare in conflitto con quelli dell'ingegnere, che mira a incrementare la produzione di metaboliti. Ne può derivare uno squilibrio metabolico: il maggiore flusso di precursori, ottenuto potenziando vie eterologhe, potrebbe non essere sostenuto dalle reazioni a valle; l'accumulo o la carenza di intermedi potrebbero compromettere la vitalità e la produttività; inoltre, la sintesi di proteine eterologhe può consumare risorse energetiche e attivare risposte di stress. Un ulteriore limite è legato al passaggio dalla sperimentazione di laboratorio alla produzione industriale. I ceppi, spesso sviluppati e valutati in condizioni controllate, devono invece operare in terreni di coltura complessi e in bioreattori caratterizzati da miscelazione non uniforme. In tali condizioni, possono mostrare difficoltà di adattamento all'ambiente variabile e di mantenimento dell'omeostasi metabolica.

Durante l'impiego dei ceppi sottoposti a ingegneria metabolica possono emergere mutazioni indesiderate, che generano sottopopolazioni in competizione per risorse limitate, come energia e substrato. Questa eterogeneità fenotipica tende a compromettere la produttività complessiva. Se le varianti meno produttive riescono a sfuggire ai meccanismi di controllo e a proliferare durante l'aumento di scala della fermentazione, possono arrivare a dominare la popolazione e determinare un fenotipo sostanzialmente improduttivo.

Una possibile contromisura consiste nella progettazione di circuiti genetici a feedback capaci di attribuire al ceppo iperproduttore un vantaggio competitivo in termini di crescita. La maggiore fitness potrebbe funzionare come un meccanismo di ricompensa e penalizzazione, promuovendo la proliferazione delle cellule ad alta produttività e limitando

quella dei cheater, ossia delle sottopopolazioni degenerative caratterizzate da un indice TYP ridotto. In tal modo si rafforzerebbe la cooperazione microbica e si favorirebbe il prevalere dei ceppi iperproduttori su quelli meno produttivi. Dal punto di vista evolutivo, questa strategia mira a sostenere il beneficio collettivo della popolazione, anziché il solo vantaggio delle singole cellule. Per realizzarla, è necessario integrare circuiti di feedback nelle cellule impiegate come fabbriche biologiche, così che possano regolare autonomamente l'espressione della via metabolica in risposta all'eterogeneità interna e alle sollecitazioni ambientali. In linea di principio, un sistema composto da sensore, trasduttore e attuatore può rilevare il carico metabolico, attivare specifiche risposte trascrizionali e compensare la riduzione dell'attività metabolica. L'implementazione di tale strategia richiede biosensori basati su fattori di trascrizione e circuiti capaci di tradurre il segnale in decisioni cellulari, così da favorire la crescita delle popolazioni più produttive e contenere quelle a bassa resa. La rassegna esamina i progressi recenti nella progettazione di circuiti genetici a feedback che conferiscono un vantaggio selettivo alle cellule, con l'obiettivo di promuovere sistemi di bioproduzione intelligente caratterizzati da prestazioni TYP migliori.

Insegnamento universitario

Nelle università italiane la bioinformatica ha assunto un ruolo progressivamente più rilevante all'interno dei percorsi di formazione STEM, in particolare nelle scienze della vita, nella medicina, nell'informatica e nell'ingegneria. La sua integrazione nei curricula universitari risulta tuttavia caratterizzata da una significativa eterogeneità, sia per quanto riguarda il carico didattico sia per i contenuti affrontati. In assenza di linee guida nazionali che definiscano un insieme uniforme di competenze bioinformatiche minime, i contenuti dei singoli insegnamenti sono spesso determinati dal docente e dal contesto formativo nel quale il corso è inserito. La Società Italiana di Bioinformatica (BITS), attraverso il gruppo Training & Teaching, ha condotto nel 2021 un'indagine, con l'obiettivo di descrivere la situazione della didattica della bioinformatica e di elaborare successivamente indicazioni per la progettazione dei relativi insegnamenti.

L'indagine ha evidenziato una prevalenza degli insegnamenti di bioinformatica nei percorsi afferenti alle scienze della vita e alla medicina (68%), rispetto a quelli appartenenti all'area informatica e ingegneristica (32%). Inoltre, il 76% degli insegnamenti analizzati era collocato all'interno di corsi di laurea magistrale, mentre i corsi triennali rappresentavano una quota minoritaria. Il carico didattico risultava variabile, da 2 a 12 ECTS, con una mediana di 6 ECTS, dei quali circa 1 ECTS era mediamente destinato ad attività pratiche.

Anche i contenuti risultano fortemente influenzati dal percorso di studi nel quale la bioinformatica viene insegnata. Nei percorsi di area biologica e biomedica vengono trattati più frequentemente database biologici e genomici, alberi filogenetici, sequenziamento, genomica, trascrittomica, bioinformatica strutturale e biologia dei sistemi. Nei percorsi informatici e ingegneristici assumono invece maggiore rilevanza gli algoritmi per la bioinformatica, la teoria dei grafi, il clustering, il machine learning e la programmazione. Nei corsi triennali, in particolare, risultano relativamente più frequenti i fondamenti di informatica, gli algoritmi di allineamento e la programmazione in linguaggi quali Python, R, Perl o C.

Questa differenziazione riflette la natura intrinsecamente interdisciplinare della bioinformatica. L'obiettivo formativo non è infatti identico nei diversi ambiti: nei percorsi biologici e biomedici l'attenzione è maggiormente rivolta alla capacità di utilizzare strumenti computazionali per ricavare e interpretare informazioni biologiche a partire dai dati; nei percorsi informatici e ingegneristici è invece maggiormente orientata alla progettazione e allo sviluppo di metodi e strategie computazionali per l'analisi dei dati biologici.

Alla luce dei risultati della propria indagine, la BITS ha elaborato specifiche raccomandazioni per l'insegnamento della bioinformatica. Per i corsi di laurea triennale viene indicato come adeguato un insegnamento di circa 6 CFU, mentre per la laurea magistrale viene raccomandato un carico di almeno 8 CFU, sconsigliando fortemente valori inferiori a 6 CFU. La BITS suggerisce inoltre di aumentare progressivamente la componente pratica: fino a circa 2 CFU di attività pratiche nella laurea triennale e 3 CFU nella magistrale. L'obiettivo è evitare che la bioinformatica venga trasmessa esclusivamente in forma teorica o attraverso l'utilizzo passivo di software, favorendo invece attività laboratoriali nelle quali lo studente possa costruire e comprendere direttamente le procedure di analisi.

Un elemento centrale delle raccomandazioni BITS è l'interdisciplinarità. Nei percorsi di scienze della vita e medicina, la bioinformatica dovrebbe essere accompagnata da solide basi di matematica, statistica e informatica, comprese nozioni fondamentali di programmazione, algoritmi, sistemi Unix/Linux e utilizzo della riga di comando. Nei percorsi informatici e ingegneristici, al contrario, dovrebbe essere rafforzata la preparazione biologica attraverso conoscenze di biologia cellulare e molecolare e biochimica. In questo modo, la formazione dovrebbe consentire agli studenti di comprendere non soltanto come utilizzare uno strumento bioinformatico, ma anche perché impiegarlo, quali siano i suoi presupposti e come interpretarne criticamente i risultati.

La BITS raccomanda inoltre di distribuire la formazione bioinformatica lungo entrambi i livelli della formazione universitaria, evitando di concentrare tutti i contenuti in un unico insegnamento, generalmente collocato nella laurea magistrale. Nella prospettiva proposta, la laurea triennale dovrebbe fornire i fondamenti della bioinformatica, mentre la laurea magistrale dovrebbe approfondirne le applicazioni contemporanee, quali le scienze omiche, la bioinformatica strutturale e la biologia dei sistemi. Particolare importanza viene attribuita alla didattica hands-on: gli studenti dovrebbero costruire piccoli workflow, utilizzare direttamente gli strumenti e comprenderne i passaggi, evitando un approccio di tipo black box. L'attività didattica dovrebbe inoltre essere collegata a esempi di ricerca, così da sviluppare non soltanto capacità operative, ma anche la capacità di valutare criticamente i risultati prodotti.

All'interno di questo quadro generale, la situazione delle Biotecnologie presenta alcune peculiarità. La struttura dei corsi di laurea italiani è infatti fortemente dipendente dalle scelte dei singoli atenei. L'informatica può essere presente come insegnamento autonomo,

come parte di un insegnamento più ampio di informatica e statistica, oppure attraverso insegnamenti di bioinformatica. Di conseguenza, la presenza dell'informatica nel percorso formativo non implica necessariamente una formazione strutturata alla programmazione.

Nei corsi di Biotecnologie nei quali sono presenti insegnamenti di informatica o bioinformatica, i contenuti possono infatti variare considerevolmente. In alcuni casi l'informatica è affrontata principalmente come strumento per l'elaborazione e l'analisi dei dati, mentre in altri vengono introdotti elementi più propriamente computazionali. Tra gli argomenti che possono essere incontrati rientrano i database biologici, l'analisi di sequenze, la genomica e la trascrittomica, gli algoritmi di allineamento, l'analisi statistica dei dati e l'utilizzo di strumenti bioinformatici. La programmazione vera e propria costituisce invece una componente più specifica: quando viene prevista, i linguaggi maggiormente utilizzati sono Python e R, generalmente applicati all'analisi dei dati biologici e alle procedure bioinformatiche.

Questa distinzione è importante perché la semplice presenza di un insegnamento denominato "Informatica" o "Bioinformatica" non permette di dedurre automaticamente che lo studente acquisisca competenze di programmazione. Un corso può infatti concentrarsi sull'utilizzo di software e banche dati senza prevedere lo sviluppo di codice, mentre altri insegnamenti introducono esplicitamente linguaggi di programmazione, algoritmi ed esercitazioni pratiche. La stessa BITS, nelle proprie raccomandazioni per i percorsi biologici, considera tuttavia le conoscenze di base di programmazione una componente importante della formazione bioinformatica, insieme a matematica, statistica, informatica, Unix/Linux e riga di comando.

Nel complesso, emerge quindi una situazione caratterizzata da una crescente presenza dell'informatica e della bioinformatica nelle discipline STEM e, in particolare, nelle scienze della vita, ma anche da una notevole differenziazione dei livelli formativi. Nelle Biotecnologie l'esposizione all'informatica può essere relativamente limitata e orientata prevalentemente all'utilizzo di strumenti per l'analisi dei dati, mentre una formazione più sistematica in programmazione e bioinformatica tende a essere maggiormente sviluppata nei percorsi magistrali specialistici. Le raccomandazioni BITS delineano invece un modello nel quale le competenze computazionali dovrebbero essere introdotte già nella formazione triennale e successivamente approfondite nella magistrale, attraverso una combinazione di basi teoriche, programmazione, attività pratiche e applicazioni a problemi biologici reali.

L’informatica nella scuola italiana

Le Linee guida per le discipline STEM, e le più recenti Indicazioni nazionali vogliono rafforzare l'insegnamento delle competenze digitali e informatiche in tutti i cicli scolastici.
Nella scuola primaria l'informatica viene introdotta gradualmente attraverso attività concrete, giochi, esperienze laboratoriali e, quando opportuno, attività "unplugged", cioè senza l'utilizzo diretto di dispositivi elettronici. L'obiettivo principale è avvicinare gli alunni ai concetti fondamentali dell'informatica attraverso situazioni semplici e vicine alla loro esperienza.
Gli alunni imparano a riconoscere gli algoritmi nelle attività quotidiane, a scomporre un problema in parti più semplici e a comprendere le basi della programmazione. Possono inoltre creare semplici contenuti digitali, utilizzare applicazioni informatiche e iniziare a comprendere la differenza tra dati e informazioni.
In questa fase l'informatica non consiste principalmente nell'imparare a usare uno specifico programma, ma nel costruire le prime capacità di ragionamento logico, problem solving e uso responsabile della tecnologia.
Nella scuola secondaria di primo grado gli studenti passano da un utilizzo prevalentemente operativo della tecnologia a una comprensione più consapevole del suo funzionamento.
Particolare attenzione verrà dedicata agli algoritmi, alla programmazione, alle variabili, ai dati e alla struttura dei programmi. Gli studenti possono progettare e modificare semplici programmi, individuare errori e comprendere il funzionamento di Internet, dei dispositivi informatici e delle reti.
Nelle scuole superiori l'informatica raggiunge un livello maggiore di astrazione e complessità. Gli studenti approfondiscono la progettazione e l'analisi degli algoritmi, la programmazione, le strutture dei dati e la modellazione di problemi reali.
Nel biennio vengono introdotti anche concetti più avanzati, come la valutazione dell'efficienza degli algoritmi, la programmazione modulare, le strutture dati e l'utilizzo di linguaggi di programmazione testuali. Le attività possono comprendere la raccolta e l'analisi di dati attraverso sensori, la realizzazione di applicazioni e la simulazione di sistemi reali.
Il percorso informatico previsto per i diversi livelli scolastici segue quindi una progressione:
Scuola primaria: scoperta dei concetti fondamentali, pensiero logico, primi algoritmi, coding e uso responsabile degli strumenti digitali.

Scuola secondaria di primo grado: approfondimento di algoritmi, dati, programmazione, Internet e sicurezza digitale.
Scuola secondaria di secondo grado: progettazione di algoritmi, programmazione più avanzata, strutture dati, analisi dei problemi, progettazione di sistemi e riflessione sugli aspetti etici e sociali della tecnologia.
La funzione dell'informatica è anche quella di sviluppare capacità trasversali come pensiero critico, precisione, creatività, capacità di analisi, astrazione e problem solving.
Attualmente, quindi, l'indirizzo ministeriale va verso una presenza dell'informatica fin dalla scuola primaria e lungo tutto il percorso scolastico, con un progressivo passaggio dall'uso consapevole della tecnologia alla comprensione dei suoi principi e alla programmazione. Le Linee guida STEM e le nuove Indicazioni nazionali rafforzano questo orientamento, insieme all'educazione alla cittadinanza digitale e all'attenzione verso temi recenti come l'intelligenza artificiale, la privacy, la sicurezza informatica e l'impatto ambientale delle tecnologie.
È però importante distinguere tra ciò che è previsto dai documenti ministeriali e ciò che viene effettivamente realizzato nelle singole scuole. Le competenze indicate costituiscono infatti un quadro di riferimento per la progettazione didattica, mentre la quantità e la qualità delle attività informatiche possono variare in base all'organizzazione della scuola, alla disponibilità di laboratori e dispositivi, alla formazione dei docenti e alle scelte dei singoli istituti.
In sintesi, la direzione attuale è quella di considerare l'informatica una competenza di base per tutti gli studenti, non limitata agli indirizzi tecnici o scientifici. Si parte dalla scuola primaria con il pensiero computazionale e la cittadinanza digitale, si prosegue nella scuola media con algoritmi e programmazione e si arriva alle superiori alla progettazione, alla modellazione dei problemi e a una maggiore consapevolezza dell'impatto delle tecnologie sulla società. Nonostante però questi programmi è anche da sottolineare che alla scuola primaria e alla scuola secondaria di primo grado non è presente un vero e proprio orario dedicato all'informatica, ma farà parte di una nuova materia chiamata educazione civica. Per quanto riguarda le scuole superiori varia molto dall'indirizzo, nel liceo scientifico sono previste due ore settimanali per tutti gli anni. Normalmente gli istituti tecnici hanno delle ore dedicate, specialmente negli istituti tecnici con indirizzo informatico dove l'informatica è il cuore dell'indirizzo di studi; per quanto riguarda le biotecnologie, gli istituti tecnici dedicati includono delle ore di laboratorio e delle ore dedicate alla teoria,

solitamente queste però si svolgono solo il primo anno, nonostante ciò, l'informatica negli anni successivi non scompare del tutto ma viene integrata in altre materie.

L'informatica è oggi prevista come competenza formativa lungo tutto il percorso scolastico, ma non dispone di un monte ore autonomo e uniforme per tutti gli studenti. Nella scuola primaria viene prevalentemente integrata nelle diverse discipline e nelle attività interdisciplinari; nella secondaria di primo grado è affidata soprattutto alla Tecnologia; nella scuola secondaria di secondo grado diventa una disciplina autonoma soltanto in alcuni indirizzi, come il liceo scientifico delle Scienze Applicate e, in modo molto più approfondito, gli istituti tecnici a indirizzo Informatica e Telecomunicazioni.

Il riconoscimento della rilevanza del CT al di fuori dell'Italia si è tradotto anche in scelte di politica scolastica. Possiamo citare per esempio l'Inghilterra (parte del Regno Unito) che è stata uno dei primi paesi a includerlo come parte obbligatoria del programma scolastico, già nel 2014, ed entro il 2018 tutti i 50 stati degli Stati Uniti lo avevano adottato nei curricula come politica ufficiale. In paesi come il Ghana e la Nigeria esiste l'impegno a introdurlo come componente obbligatoria, ma ciò non è ancora diventato una politica ufficiale. In India l'informatica (computer science, CS) è diventata materia obbligatoria.

Sebbene il Sudafrica si sia impegnato a introdurre il pensiero computazionale nei programmi scolastici, non ha ancora avviato con successo la fase pilota dell'iniziativa, il cui inizio era previsto per il 2021 in 1.000 scuole. È importante sottolineare le notevoli difficoltà esistenti in termini di infrastrutture e di competenze degli insegnanti. Si ritiene che le università, in quanto attori chiave nella preparazione degli studenti al mondo del lavoro, abbiano un ruolo fondamentale da svolgere nell'affrontare tali sfide.

È interessante notare che globalmente si stia iniziando a dare importanza a questa materia, nonostante ci sia ancora strada da fare.

Tecniche di insegnamento del pensiero computazionale nelle diverse fasce d’età

La bioinformatica, intesa come l'uso di risorse computazionali per classificare ingenti quantità di dati grezzi ed estrarre informazioni significative dai dataset, riveste un ruolo sempre più centrale nella pratica scientifica quotidiana. La promozione di una società scientificamente informata richiede pertanto un potenziamento delle risorse curricolari e didattiche nelle scuole di ogni ordine e grado, fondato su iniziative validate dalla ricerca in didattica delle scienze. Parallelamente, il pensiero computazionale (Computational Thinking, CT) si è affermato come competenza essenziale della società digitale contemporanea. Il costrutto, introdotto da Papert (come già accennato prima) nel 1987 per indicare i processi cognitivi associati alla risoluzione di problemi mediante un sistema computazionale, è stato successivamente ampliato da Wing (2006), che lo ha definito come l'insieme dei processi di risoluzione di problemi, progettazione di sistemi e comprensione del comportamento umano fondati sui principi e sui metodi dell'informatica (come anche spiegato precedentemente). In questa accezione, il CT designa un insieme di strumenti mentali coinvolti nell'espressione di soluzioni, sotto forma di passaggi computazionali o algoritmi, eseguibili da una macchina o da un agente informatico, e si articola in cinque fasi ricorsive: l'analisi dello spazio del problema, la scomposizione in sottoproblemi più semplici, la rappresentazione adeguata dei dati, lo sviluppo di una sequenza algoritmica e la verifica/correzione dell'implementazione. Il CT incarna così una forma di ragionamento analitico che ricorda da vicino il ragionamento matematico, quello ingegneristico e quello scientifico, valorizzando al contempo elementi quali la fiducia in sé, la perseveranza, la tolleranza verso l'incertezza e la comunicazione: le nazioni sviluppate lo considerano il fondamento della futura società tecnologica, mentre i paesi in via di sviluppo vi individuano uno strumento per colmare i divari educativi.

Sul piano neuroevolutivo, l'apprendimento e l'acquisizione di queste competenze si inseriscono nei più ampi processi di plasticità cerebrale che accompagnano lo sviluppo. Un utile quadro teorico distingue i processi "experience-expectant", in cui il sistema neurale si è evoluto per attendere specifici input ambientali entro finestre temporali definite dello sviluppo, dai processi "experience-dependent", che riflettono cambiamenti neurali duraturi legati a esperienze la cui natura e la cui collocazione temporale variano considerevolmente da individuo a individuo. È entro questa cornice teorica che si colloca la rassegna che segue, organizzata secondo l'età dei campioni studiati — dalla prima infanzia alla tarda adolescenza — con attenzione sia alle modalità didattiche adottate in ciascun intervento, sia ai risultati ottenuti.

Uno studio condotto con tecnica fNIRS ha indagato come lattanti di 9 mesi apprendano quali caratteristiche, tra quelle concorrenti in un ambiente visivo disordinato, siano informative per l'attenzione successiva in contesti nuovi. I risultati hanno mostrato che i neonati acquisiscono rapidamente una conoscenza top-down delle caratteristiche visive rilevanti attraverso l'apprendimento di regole astratte, e sono in grado di generalizzare tali regole per modulare l'attenzione visiva verso le caratteristiche apprese anche in una situazione nuova. I lattanti che hanno mostrato una migliore capacità di generalizzazione hanno fatto registrare anche una maggiore connettività tra corteccia prefrontale e corteccia visiva, a indicare un legame precoce tra la maturazione dei circuiti fronto-visivi e l'apprendimento di regole astratte.

Nella fascia d'età compresa tra i tre e i sei anni, il pensiero computazionale non si limita a porre le basi per il ragionamento logico e la risoluzione di problemi, ma favorisce anche disposizioni quali curiosità, creatività e perseveranza. Una rassegna della letteratura dedicata a questa fascia d'età ha organizzato le evidenze disponibili lungo tre dimensioni: le esigenze educative che motivano l'integrazione del CT, gli strumenti e le strategie pedagogiche adottati in classe, e gli esiti di apprendimento osservati.

Sul piano delle esigenze educative, il 28,6% degli studi esaminati ha utilizzato il pensiero computazionale per affrontare obiettivi legati allo sviluppo di competenze cognitive, sociali e comunicative, con l'obiettivo di potenziare la risoluzione di problemi matematici, le abilità spaziali, le funzioni esecutive, l'acquisizione del linguaggio e le interazioni sociali; gli studi hanno inoltre riconosciuto la necessità di adattare le attività di programmazione e robotica alle capacità cognitive e fisiche dei bambini in età prescolare, mantenendo alta la motivazione. Sul piano didattico, la maggior parte degli studi ha privilegiato l'uso di strumenti software quali kit di robotica e applicazioni di programmazione, spesso trascurando l'opportunità di integrare il CT da prospettive più ampie, specifiche per l'insegnante e per l'esperienza degli studenti. Gli strumenti impiegati sono riconducibili a tre contesti principali: ambienti di programmazione visuale (Scratch, Alice, Game Maker, MakeCode, Code.org), che utilizzano interfacce a blocchi per costruire sequenze logiche senza richiedere la padronanza della sintassi di un linguaggio di programmazione, e sono ampiamente diffusi nella scuola primaria grazie alla loro accessibilità; kit di robotica e dispositivi tangibili, che combinano componenti fisici e strumenti di programmazione, favorendo l'apprendimento attivo, la progettazione iterativa e la risoluzione collaborativa dei problemi; e simulazioni basate sul web. Programmi come

Scratch Jr., in particolare, offrono un ambiente di programmazione visuale che stimola il pensiero simbolico e l'espressione creativa senza richiedere competenze di lettura, scrittura o digitazione, consentendo ai bambini di sperimentare strutture algoritmiche attraverso la narrazione e l'animazione e di sviluppare così sia la padronanza computazionale sia la competenza narrativa.

Sul piano degli esiti, i contributi analizzati mostrano come il pensiero computazionale potenzi competenze fondamentali nella prima infanzia, tra cui il problem solving, l'astrazione e la sequenziazione logica, favorendo al contempo lo sviluppo sociale e cognitivo, e in particolare la comprensione numerica e la risoluzione di problemi matematici attraverso sequenze logiche e schemi ripetitivi. La rassegna sottolinea come il potenziale educativo di questi strumenti non risieda nella loro sofisticatezza tecnologica, bensì nel modo in cui vengono integrati, quali mediatori pedagogici, in pratiche adeguate allo sviluppo: se sostenuti da uno scaffolding intenzionale, essi contribuiscono non solo all'acquisizione di competenze di programmazione, ma anche allo sviluppo di capacità più ampie quali la perseveranza, la collaborazione, l'astrazione e la creatività. Restano tuttavia aperte due sfide rilevanti: la tendenza a privilegiare strumenti digitali come robot e applicazioni a scapito di approcci più idonei allo sviluppo dei bambini piccoli, come le attività "unplugged" o basate sul gioco; e una persistente mancanza di chiarezza concettuale nella distinzione tra abilità cognitive (astrazione, sequenziazione) e disposizioni all'apprendimento, che rischia di generare discrepanze tra le iniziative didattiche e le caratteristiche evolutive dei bambini piccoli.

Un filone di ricerca ha esaminato le differenze individuali nei meccanismi neurali dell'apprendimento in bambini con disturbo dello spettro autistico (ASD) e nei loro coetanei con sviluppo tipico, con particolare attenzione alle abilità numeriche tra bambini e adolescenti. Nei bambini con sviluppo tipico è emersa una correlazione negativa tra abilità numeriche e attivazione funzionale in un network di regioni associate alla cognizione numerica, mentre nei bambini con ASD si è osservato l'effetto opposto; questi ultimi necessitavano inoltre di una maggiore quantità di informazioni per giungere a una decisione, e tale soglia decisionale più elevata ha moderato la relazione tra le differenze individuali nelle abilità numeriche e il coinvolgimento dei sistemi di controllo prefrontali. Un ulteriore studio, condotto nell'ambito di un breve programma di tutoraggio in matematica, ha mostrato che, sebbene in media i bambini migliorassero le prestazioni grazie all'esercizio su specifici problemi, i ritmi di apprendimento variavano notevolmente

da un bambino all'altro: chi apprendeva più rapidamente mostrava prestazioni migliori sia sui problemi esercitati sia su problemi nuovi, e a livello neurale presentava una maggiore sovrapposizione delle rappresentazioni neurali dei due tipi di problema nelle regioni temporali mediali, frontali e temporali, accompagnata da una connettività ridotta e da una maggiore segregazione tra le aree implicate nella risoluzione di problemi matematici.

Sul versante della lettura, l'integrazione di misure ERP e fMRI a livello della regione occipito-temporale ventrale (vOT) sinistra ha mostrato che la modulazione del segnale BOLD era più marcata per le parole rispetto a stringhe di falsi caratteri nei bambini con sviluppo tipico della lettura, ma non in quelli con difficoltà di lettura, a indicare una sintonizzazione ortografica più avanzata nel primo gruppo; lo stesso lavoro ha inoltre evidenziato che le aree della vOT con attivazione preferenziale per specifiche categorie di materiale scritto potevano essere individuate solo tenendo conto delle differenze individuali nella loro localizzazione, a conferma dell'importanza di considerare tali differenze nello studio dello sviluppo cerebrale.

Un altro studio preso in considerazione ha condotto su un'ampia coorte longitudinale di bambini e adolescenti di età compresa tra i 4 e i 25 anni ha esaminato lo sviluppo della memoria verbale e visuo-spaziale. Per la memoria verbale, la ritenzione a breve e a lungo termine ha seguito un andamento simile fino a circa 10 anni di età; tra i 10 e i 15 anni, tuttavia, i miglioramenti nella ritenzione a lungo termine hanno superato gli incrementi registrati negli intervalli più brevi. Per la memoria visuo-spaziale si sono invece osservati aumenti lineari delle prestazioni di ritenzione a lungo termine lungo l'intero arco di età considerato, accompagnati da miglioramenti anche a breve termine. Prestazioni di rievocazione superiori, indipendentemente da intervalli temporali e compiti, sono risultate associate alla microstruttura dell'ippocampo anteriore e a un minore spessore della corteccia prefrontale laterale.

Le abilità cognitive dei bambini con abilità nella norma e di quelli con abilità inferiori alla media sono state descritte efficacemente attraverso due costrutti distinti, l'intelligenza cristallizzata e quella fluida, che tendono a differenziarsi maggiormente con l'avanzare dell'età. La microstruttura della sostanza bianca, in particolare quella del fascicolo longitudinale superiore, è risultata fortemente associata a entrambe le abilità; la relazione tra queste ultime e i rispettivi substrati di sostanza bianca varia, peraltro, in modo dinamico con l'età, con un calo tra i 7 e gli 8 anni seguito da un incremento intorno ai 10 anni.

Un ulteriore filone di ricerca, condotto su dataset di scoperta e di replica relativi a bambini di 9-10 anni, ha esaminato la connettività funzionale a riposo e le sue relazioni con l'abilità cognitiva generale, riscontrando un'elevata riproducibilità della connettività e dell'architettura dei network tra i bambini; una rete di connessioni ampiamente distribuita, comprendente sia la connettività interna ai singoli network funzionali sia quella tra network diversi, è risultata associata a livelli più elevati di abilità cognitiva.

Si è notato in uno studio che bambini, adolescenti (8-17 anni) e adulti hanno appreso associazioni stimolo-azione a partire dal feedback, con un carico di apprendimento manipolato in modo da rientrare entro la capacità della memoria di lavoro o da superarla. I partecipanti di 8-12 anni hanno appreso più lentamente rispetto a quelli di 13-17 anni e sono risultati più sensibili al carico della memoria di lavoro. Attraverso la modellazione computazionale dell'uso della memoria di lavoro e dei processi di apprendimento per rinforzo, gli autori hanno osservato differenze legate all'età più marcate nell'apprendimento per rinforzo che nella memoria di lavoro: il tasso di apprendimento per rinforzo è aumentato significativamente con l'età nel corso dell'adolescenza, mentre i parametri della memoria di lavoro hanno mostrato cambiamenti più sottili, concentrati soprattutto nella prima adolescenza, a sottolineare l'importanza dei cambiamenti nei processi di apprendimento per rinforzo, più che nella memoria di lavoro, per l'apprendimento nel corso dell'infanzia e dell'adolescenza.

Il più ampio tra gli studi sperimentali qui considerati ha coinvolto 122 bambini di quarta e quinta elementare, provenienti da quattro classi di scuole pubbliche di due città colombiane; 111 di essi (età compresa tra 9 anni e 11 anni e 11 mesi, età media 10,75 anni; 38 femmine e 73 maschi) soddisfacevano i criteri di inclusione, che escludevano condizioni mediche limitanti, diagnosi di disturbi del neurosviluppo o deficit sensoriali non corretti. Lo studio, disegnato come trial controllato randomizzato (RCT) secondo le linee guida CONSORT e le indicazioni Cochrane per il controllo del rischio di bias, ha previsto l'assegnazione ai gruppi mediante randomizzazione con busta chiusa gestita da un ricercatore esterno, l'occultamento dell'allocazione ai valutatori dei test pre- e post-intervento e l'imputazione dei dati mancanti per i partecipanti che non hanno completato lo studio, confrontata con un'analisi per-protocol. Il gruppo sperimentale ha seguito il programma COGNI-MACHINE, articolato in 20 moduli erogati nell'arco di 12 settimane attraverso incontri settimanali di due ore consecutive (26 ore complessive), pienamente integrati nel curricolo ufficiale di Tecnologia e Informatica, di cui hanno sostituito le

consuete lezioni. Ogni sessione era strutturata come un laboratorio pratico fondato sull'apprendimento basato sui problemi (PBL), con attività "unplugged" e "plugged", queste ultime imperniate sulla programmazione a blocchi, con cui gli studenti creavano giochi e animazioni, svolte in coppia o in piccoli gruppi secondo il principio "low floor, high ceiling", che garantiva un accesso inclusivo a partecipanti con diversi livelli di esperienza tecnologica pregressa, consentendo al contempo agli studenti più avanzati di affrontare compiti via via più complessi.



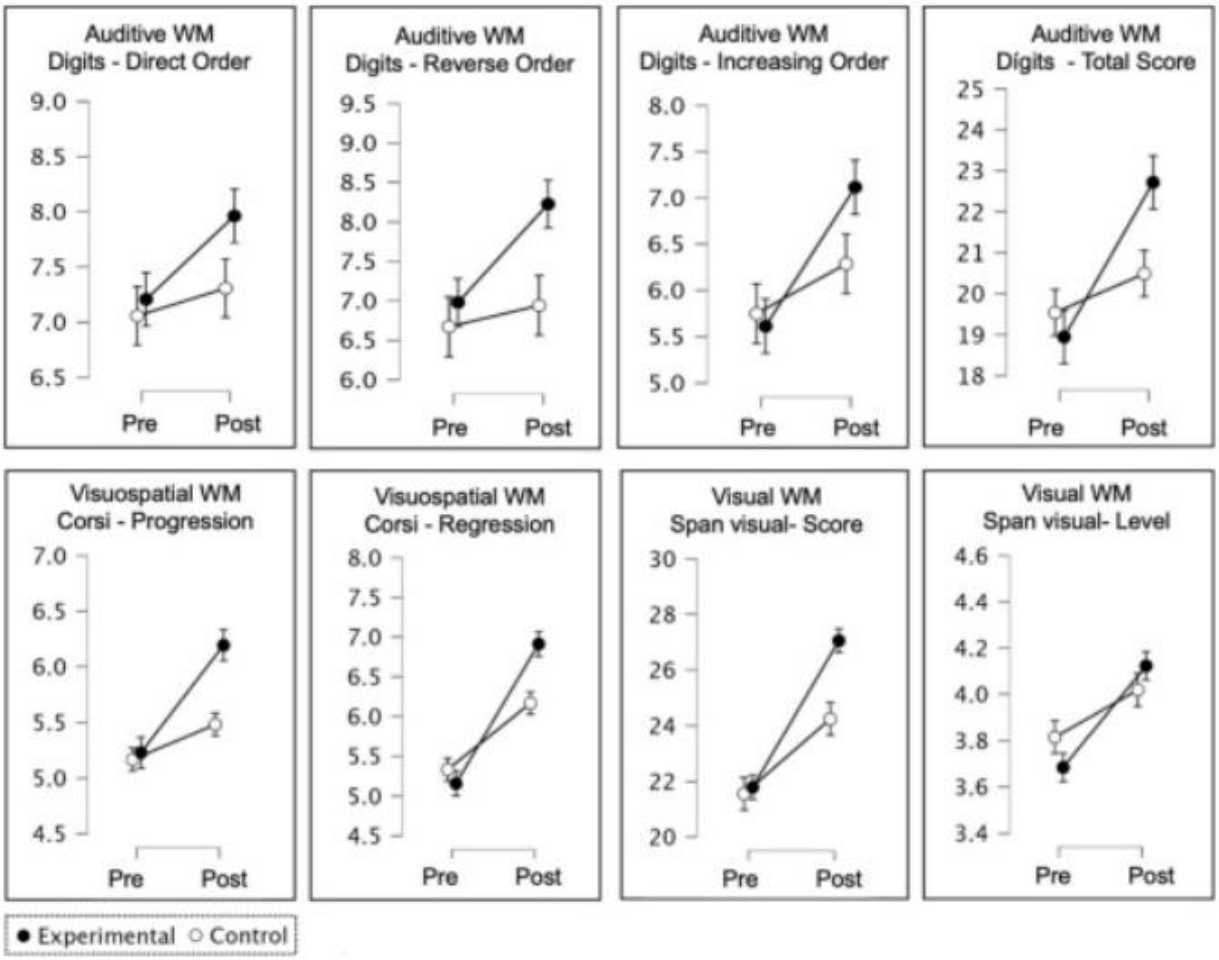
Esempi di attività plugged che utilizzano micro:bit e kit di robotica. Robledo-Castro C., Castillo-Ossa LF, Hederich-Martínez C. (2025) Computational thinking training and its effects on working memory, flexibility, and inhibition: Randomized controlled trial in fifth-grade children. ELSEVIER 100, 101850 https://doi.org/10.1016/j.appdev.2025.101850

La validità di contenuto dell'intervento è stata verificata da quattro esperti di didattica del pensiero computazionale. Il gruppo di controllo, sottoposto al trattamento abituale (TAU), ha seguito le ordinarie lezioni di Tecnologia e Informatica secondo il programma ministeriale, con la possibilità di partecipare all'intervento al termine della raccolta dati. La valutazione degli esiti, condotta da due valutatori in cieco rispetto al gruppo di appartenenza dei partecipanti, ha impiegato una batteria di test standardizzati mirati a componenti distinte delle funzioni esecutive: il subtest Digit Span della WISC-V per la memoria di lavoro uditivo-verbale; il test Visual Span della WISC-V per la memoria di

lavoro visiva; il test dei cubi di Corsi, in ordine diretto e inverso, per la memoria di lavoro visuo-spaziale; la versione a 48 carte del Wisconsin Card Sorting Test (WCST) per la flessibilità cognitiva; e il test di Stroop, preceduto da due tavole di controllo per la lettura di parole e la denominazione di colori, per il controllo inibitorio.

I risultati hanno mostrato un miglioramento significativo, con ampiezza dell'effetto elevata, della memoria di lavoro uditivo-verbale nel gruppo sperimentale rispetto al gruppo di controllo, l'effetto di maggiore entità associato all'intervento, insieme a un miglioramento di entità moderata della memoria di lavoro visuo-spaziale, confermato dai confronti post-hoc pur in presenza di un lieve effetto di pratica anche nel gruppo di controllo. La memoria di lavoro visiva non ha invece mostrato differenze statisticamente significative, verosimilmente per la minore sensibilità di questa misura rispetto al punteggio totale del compito. Sul fronte del controllo inibitorio, l'analisi non ha rilevato effetti significativi nel compito di lettura delle parole, mentre è emerso un effetto di lieve entità nel compito di denominazione dei colori; la flessibilità cognitiva, valutata tramite il WCST, non ha mostrato differenze significative riconducibili alla partecipazione all'intervento, confermandosi come la funzione esecutiva meno sensibile a questo tipo di training.

Gli autori collegano il marcato effetto sulla memoria di lavoro uditivo-verbale alla componente collaborativa del programma, che richiedeva agli studenti di discutere e negoziare verbalmente con il proprio compagno le istruzioni e le soluzioni proposte in ciascun modulo, in misura maggiore rispetto a interventi centrati esclusivamente su compiti visivi individuali. Il confronto con uno studio correlazionale precedente, che aveva rilevato un'associazione debole ma significativa tra pensiero computazionale e memoria di lavoro uditiva in bambini di 9-10 anni impegnati per sei mesi in attività di coding individuale con Scratch, rafforza questa interpretazione: l'ampiezza dell'effetto nettamente maggiore riscontrata in COGNI-MACHINE viene attribuita al lavoro collaborativo in coppia, che avrebbe favorito una maggiore interazione e comunicazione tra pari rispetto alla programmazione individuale.



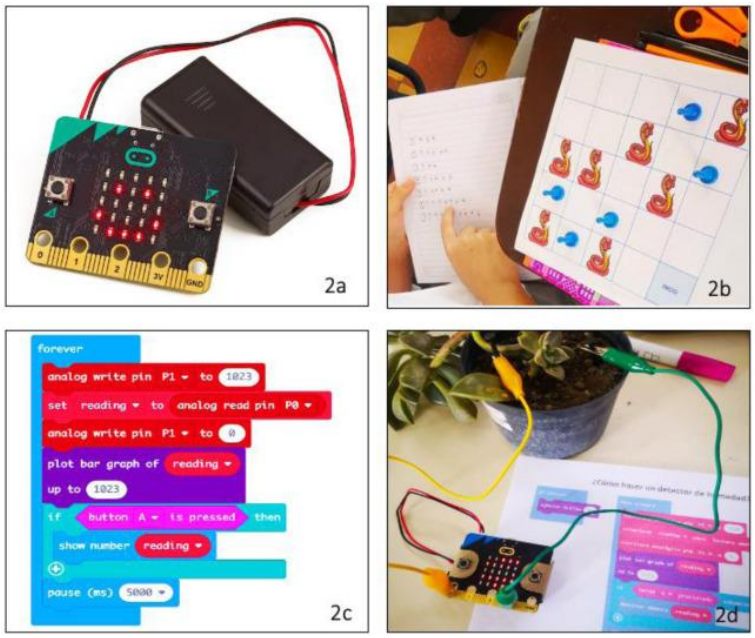
Grafici dei test di memoria di lavoro in entrambi i gruppi. Robledo-Castro C., Castillo-Ossa LF, Hederich-Martínez C. (2025) Computational thinking training and its effects on working memory, flexibility, and inhibition: Randomized controlled trial in fifth-grade children. ELSEVIER 100, 101850 https://doi.org/10.1016/j.appdev.2025.101850

Lo studio segnala infine, tra i limiti e le direzioni per la ricerca futura, l'impatto contenuto dell'intervento sulla flessibilità cognitiva, coerente con diversi studi precedenti che non hanno rilevato cambiamenti significativi in questo ambito a seguito di programmi analoghi: un dato che suggerisce come questa specifica funzione esecutiva possa richiedere strategie di allenamento più mirate rispetto a un training generico di pensiero computazionale.

Un precedente studio pilota, condotto nella stessa area geografica colombiana, aveva coinvolto bambini di quinta elementare di 10 e 11 anni con sviluppo tipico, reclutati nel periodo di ripresa della didattica in presenza dopo diciannove mesi di didattica a distanza dovuta alla pandemia di COVID-19. Su 36 bambini invitati a partecipare, 31 (e le rispettive famiglie) hanno accettato e sono stati assegnati casualmente a un gruppo sperimentale (N=17; 5 femmine, 12 maschi) e a un gruppo di controllo TAU (N=14; 3 femmine, 11

maschi), tutti provenienti da contesti socio-economici medio-bassi; un bambino del gruppo di controllo si è successivamente ritirato per un cambio di residenza, portando il campione finale a 30 partecipanti.

Il gruppo sperimentale ha seguito un intervento di otto settimane, con due incontri settimanali di 90-120 minuti svolti nei laboratori informatici della scuola durante l'orario dedicato a informatica e tecnologia, comprensivo di attività da svolgere anche a casa. Il programma, articolato in sette moduli per un totale di 25 ore, riprendeva nei primi cinque moduli la struttura del progetto "Programmazione per bambini" e prevedeva, in ciascuna sessione, un'alternanza tra attività "unplugged", volte a introdurre gradualmente i principi del pensiero computazionale attraverso materiali tangibili prima di integrare un agente computazionale, e attività "plugged", basate sui computer del laboratorio e sui microcontrollori Micro:bit, dotati di accelerometro e sensori di umidità, luce e suono, programmati in linguaggio a blocchi tramite l'ambiente MakeCode.



Esempi di attività plugged e unplugged. Robledo-Castro C., Castillo-Ossa LF, Hederich-Martínez C. (2023) Effects of a computational thinking intervention program on executive functions in children aged 10 to 11. ELSEVIER 35,100563 https://doi.org/10.1016/j.ijcci.2022.100563

Gli esiti sono stati misurati attraverso la Batteria Neuropsicologica delle Funzioni Esecutive e dei Lobi Frontali (BANFE-2), costituita da 15 test raggruppati in tre scale — orbitofrontale, prefrontale anteriore e prefrontale dorsolaterale, quest'ultima ulteriormente suddivisa in una sottoscala di memoria di lavoro (comprendente, tra gli altri, il Test di Segnalazione Autodiretta per la componente visuospaziale e il Test di Ordinamento Alfabetico per la manipolazione verbale) e una di funzione esecutiva (comprendente, tra

gli altri, la Torre di Hanoi per la pianificazione sequenziale, il Verb Generation Test per la fluenza verbale e il Test del Labirinto per la pianificazione visuospaziale) somministrata in cieco rispetto all'assegnazione ai gruppi.

I risultati non hanno evidenziato un effetto significativo del trattamento sulla scala orbitofrontale, la cui differenza complessiva tra i gruppi è apparsa riconducibile al naturale trascorrere del tempo piuttosto che all'intervento. È emerso invece un effetto significativo, con ampiezza elevata, sulla scala prefrontale anteriore, e un effetto di entità compresa tra piccola e media sulla scala dorsolaterale-funzione esecutiva, in entrambi i casi confermato dai confronti post-hoc con correzione di Bonferroni, che non hanno rilevato differenze significative né al pre-test tra i gruppi né tra pre- e post-test nel gruppo di controllo. Gli autori collegano questi esiti alla natura stessa della risoluzione computazionale di problemi, che richiede di pianificare una sequenza di azioni orientata a un obiettivo finale e di elaborare rappresentazioni flessibili del problema per valutare alternative e correggere errori, processi che, integrando attività plugged e unplugged, l'intervento avrebbe direttamente sollecitato. Nel complesso, lo studio pilota ha suggerito un impatto positivo del training su processi metacognitivi, funzioni esecutive e memoria di lavoro, ma non sui processi di inibizione e di gestione del rapporto rischio-beneficio tipicamente associati all'area orbitofrontale, delineando così la portata e i limiti di questo tipo di programmi nella fascia d'età considerata.

Poiché gran parte della formazione bioinformatica avviene tradizionalmente a livello post-universitario, molti studenti di scienze della vita giungono alla laurea con una preparazione bioinformatica scarsa o nulla, un problema aggravato, nei tentativi di integrazione a livello universitario, dall'insufficiente preparazione pregressa degli studenti. In Scozia, a differenza del resto del Regno Unito, la bioinformatica è inclusa nei programmi di Biologia e Biologia Umana di livello avanzato, sebbene sia trattata in modo sommario sia nei programmi sia nei libri di testo consigliati, senza alcun requisito formale relativo all'esperienza pratica.

Per colmare questo divario, un gruppo di ricerca ha ideato due laboratori didattici integrati nei programmi scolastici scozzesi, volti a introdurre gli studenti, e i rispettivi insegnanti, alla bioinformatica, rafforzando al contempo altri contenuti curricolari quali la mutazione e la nutrizione, con un duplice obiettivo: da un lato, far comprendere ai giovani il ruolo della bioinformatica e i suoi legami interdisciplinari con informatica, matematica e biologia; dall'altro, favorire il coinvolgimento del pubblico utilizzando la bioinformatica come

introduzione alle tecniche delle scienze computazionali. Il laboratorio meglio documentato tra i due, destinato agli alunni della scuola secondaria di primo grado e del primo biennio della secondaria di secondo grado (13-16 anni) e allineato ai programmi SQA di livello National 4 e National 5, si basa sull'analisi di sequenze di DNA reali estratte da una salsiccia di maiale artigianale ("Food Detective"): utilizzando strumenti e dati accessibili pubblicamente, gli studenti scoprono che il prodotto contiene, oltre al DNA suino, anche tracce di DNA di pecora, pollo, bovino e umano, un risultato che consente di affrontare temi quali l'affidabilità dei dati, la contaminazione e le frodi alimentari. In seguito al riscontro di insegnanti e alunni, che avevano completato l'attività più rapidamente del previsto, è stata aggiunta una seconda attività incentrata sui valori "e" (e-values) e sull'affidabilità dei risultati bioinformatici; nei laboratori successivi gli alunni si sono confrontati con sfide maggiori, pur restando coinvolti, un risultato attribuito all'approccio di ricerca-azione con cui il laboratorio è stato progressivamente affinato in base ai riscontri raccolti. Durante il laboratorio, i computer impiegati non necessitano di accesso a Internet o alla rete scolastica, il che permette di concentrarsi sui vantaggi dell'informatica a riga di comando evitando potenziali problemi tecnici, offrendo al contempo agli studenti un'esperienza autentica di ambiente di ricerca computazionale in aula. In modo coerente con le evidenze sull'efficacia dell'apprendimento attivo nelle discipline STEM, l'attività richiede agli studenti di lavorare a coppie o in piccoli gruppi compilando una scheda di lavoro, alternando i momenti pratici a discussioni collettive dei risultati.

In risposta alla pandemia di COVID-19, il gruppo di ricerca ha inoltre sviluppato risorse online aggiuntive per continuare a sostenere insegnanti e studenti, tra cui una serie di brevi video (5-10 minuti) su aspetti chiave dei workshop e sull'uso del database NCBI in altre aree del programma di Biologia, come l'impiego del visualizzatore di proteine 3D per esaminare la struttura degli enzimi, nonché un kit di risorse (un Raspberry Pi con immagine SD personalizzata per la bioinformatica, completo di custodia, alimentatore e cavi; un manuale introduttivo; il materiale didattico anche in formato elettronico) inviato agli insegnanti prima di ogni evento formativo. I riscontri raccolti presso insegnanti e studenti indicano che l'iniziativa è stata proficua, a conferma dell'ipotesi secondo cui anche gli studenti della scuola secondaria possono svolgere con successo attività pratiche di bioinformatica; il gruppo di ricerca segnala tuttavia come le principali sfide per l'integrazione della bioinformatica nel curricolo secondario riguardino le infrastrutture, la logistica e la preparazione degli insegnanti. Merita infine attenzione, in questo contesto, la

persistente disparità di genere nell'istruzione informatica: nel Regno Unito le studentesse rappresentano solo il 18% degli iscritti ai corsi universitari di informatica, un dato che però risale a scelte politiche e industriali degli anni '60-'80 volte a elevare la disciplina a un livello manageriale, dopo che i primi programmatori erano stati in prevalenza donne, a fronte del 45% in India; in Malesia la distribuzione di genere non è semplicemente equilibrata, ma vede talvolta una prevalenza femminile. Ciò contrasta con la biologia, disciplina a prevalenza femminile, e sottolinea come la disparità non sia un dato universale bensì culturalmente e storicamente determinato.

In un campione di adolescenti di età compresa tra 13 e 20 anni è stata tracciata la traiettoria evolutiva di un effetto già noto negli adulti, per cui indizi precedentemente associati a esiti di alto valore migliorano il successivo comportamento orientato all'obiettivo. L'effetto benefico delle associazioni di valore apprese sulle prestazioni successive di controllo cognitivo è emerso in modo crescente con l'età, in associazione a differenze nel reclutamento dei circuiti corticostriatali, a evidenziare l'importanza di chiarire come le interazioni mutevoli tra i processi cognitivi modifichino l'apprendimento nel corso dello sviluppo.

Se l'esistenza di periodi sensibili nella prima infanzia è stata ampiamente dimostrata per diverse funzioni, dai sistemi sensoriali al linguaggio, resta dibattuto in letteratura se, e in quale misura, anche l'adolescenza costituisca un periodo sensibile, e per quali ambiti specifici. Un'ipotesi accreditata la individua come periodo sensibile per lo sviluppo sociale: mentre adolescenti più giovani, adolescenti più grandi e adulti hanno mostrato tutti un peggioramento dell'umore in seguito a una breve esposizione all'esclusione sociale, solo gli adolescenti più giovani hanno fatto registrare anche una riduzione delle prestazioni nella memoria di lavoro verbale, a suggerire che l'ambiente sociale rivesta un ruolo particolarmente rilevante nella prima adolescenza. Sul piano delle funzioni cognitive di ordine superiore, come memoria e funzioni esecutive, le evidenze provenienti da studi di neuroimaging su animali ed esseri umani indicano che gli ormoni puberali svolgono un ruolo cruciale nella regolazione dei meccanismi di plasticità dipendente dall'esperienza durante l'adolescenza, sebbene la direzione di tale effetto, come incremento o riduzione della plasticità, possa dipendere dal sesso, dal dominio cognitivo considerato e dai network cerebrali coinvolti.

A questi fattori endogeni si affiancano influenze ambientali che agiscono con particolare intensità proprio nei periodi sensibili dello sviluppo. Lo status socio-economico (SES), pur

riflettendo una molteplicità di condizioni ambientali che possono incidere sullo sviluppo in modi differenti, è stato associato a prestazioni inferiori in diversi domini cognitivi, tra cui memoria e funzioni esecutive, e a differenze nello sviluppo cerebrale: l'esposizione alla violenza, ad esempio, è risultata associata a prestazioni di memoria inferiori, mentre una migliore qualità dell'ambiente fisico è stata collegata a una maggiore attenzione guidata dalla memoria, come confermato anche da un ulteriore studio longitudinale condotto su bambini di 6-7 anni.

Il più maturo, per età del campione, tra gli studi qui considerati ha valutato l'impatto educativo, su studenti delle scuole secondarie di secondo grado, di una serie di attività di bioinformatica di base finalizzate a scomporre una sequenza genomica batterica nei suoi geni codificanti. Il campione comprendeva 387 studenti (167 maschi e 220 femmine; età media 16,34 ± 0,67 anni) e 11 insegnanti di cinque scuole pubbliche e private di Porto e Lisbona, distribuiti in quattordici classi dell'undicesimo anno di Biologia e Geologia (16-17 anni) e cinque classi del dodicesimo anno di Biologia (17-18 anni). I partecipanti sono stati suddivisi in un gruppo sperimentale (n=292; 123 maschi, 169 femmine; età media 16,27 ± 0,68) e in un gruppo di controllo (n=95; 44 maschi, 51 femmine; età media 16,54 ± 0,62), quest'ultimo reclutato dalle medesime scuole, dai medesimi livelli scolastici e dagli stessi insegnanti del gruppo sperimentale, al fine di limitare potenziali effetti di distorsione. Come strumento didattico è stato impiegato un insieme di attività volte a identificare geni a partire da una sequenza genomica batterica, a rivelarne il contesto genomico in specie diverse e a formulare ipotesi sulla loro evoluzione, un compito reso particolarmente idoneo dalle ridotte dimensioni e dalla struttura aploide a replicone singolo tipiche dei genomi batterici, che favoriscono la genomica comparativa e permettono di approfondire argomenti di microbiologia solitamente poco trattati nella scuola secondaria superiore. Un video tutorial ha guidato insegnanti e studenti nell'analisi di una regione genomica di 2 kb di Escherichia coli, identica per tutti i partecipanti al fine di uniformare la supervisione didattica e facilitare le analisi successive. Prima delle attività di laboratorio, gli insegnanti hanno ripassato con la classe concetti già noti (genoma, geni, codoni di inizio e di stop, operoni) e introdotto nozioni nuove (Open Reading Frames, sintenia, genomica comparativa), supportati da una pagina web dedicata pensata per alleggerire il loro carico di lavoro. Il percorso didattico si è articolato in quattro fasi: lezioni teoriche su regolazione genica, genomica ed evoluzione; presentazione dei database e degli strumenti bioinformatici da utilizzare come il database NCBI, NCBI ORFfinder, NCBI Blast e

MicroScope (MaGe); svolgimento degli esercizi, accompagnato da linee guida pensate per evitare che gli studenti si sentissero sopraffatti dalla varietà di comandi e link disponibili sulle piattaforme; discussione collettiva dei risultati e delle conclusioni, con un membro del gruppo di ricerca presente per rilevare idee errate, difficoltà di ragionamento, livello di coinvolgimento e osservazioni qualitative. Il gruppo di controllo è stato esposto unicamente alle prime due fasi (lezioni introduttive e presentazione delle risorse), mentre il gruppo sperimentale ha seguito l'intero ciclo, comprensivo dei laboratori pratici e della discussione dei risultati; le classi di controllo provenivano dalle stesse scuole, dagli stessi livelli scolastici e dagli stessi insegnanti delle classi sperimentali.

I dati, raccolti tramite domande a risposta aperta e variabili dicotomiche, sono stati analizzati rispettivamente mediante test del chi-quadrato e test di McNemar, con il coefficiente phi come misura della dimensione dell'effetto, integrando un'analisi tematica qualitativa delle risposte aperte sulla base di un quadro di codifica predefinito.

Nella fase di pre-test, solo il 40,1% degli studenti del gruppo sperimentale e il 24,2% del gruppo di controllo dichiarava di aver sentito parlare di bioinformatica, generalmente senza saperla definire al di là del significato etimologico del termine; dopo una lezione introduttiva, la percentuale è salita al 99,0% in entrambi i gruppi. Nel post-test, sebbene la maggioranza degli studenti continuasse ad associare la bioinformatica alla sua etimologia (60,9% nel gruppo sperimentale, 73,6% nel gruppo di controllo), una quota maggiore di studenti del gruppo sperimentale ha citato aspetti specifici quali l'analisi dei dati, l'archiviazione delle informazioni e la genomica comparativa (31,0% contro il 22,0% del gruppo di controllo); un divario particolarmente marcato per la genomica comparativa, concetto del tutto nuovo per la maggior parte degli studenti, citata dal 6,6% del gruppo sperimentale contro l'1,1% del gruppo di controllo. Entrambi i gruppi hanno inoltre mostrato una maggiore consapevolezza dell'accessibilità delle banche dei geni come risorse ad accesso libero (81,2% contro 59,0%) e generalmente fruibili da tutti i cittadini (78,8% contro 62,1%), con percentuali più elevate nel gruppo sperimentale, un dato che, in linea con altri studi, indica un fattore di stimolo per la cultura scientifica e l'educazione STEM. Se inizialmente gli studenti associavano l'analisi bioinformatica a complessi codici informatici, dopo le attività pratiche hanno riconosciuto l'ampia gamma di applicazioni con interfacce intuitive per browser web, che non richiedono competenze di programmazione: coerentemente, dopo l'intervento solo l'1,7% degli studenti del gruppo sperimentale non ha saputo citare alcuna delle piattaforme utilizzate, contro il 16,7% del gruppo di controllo.

Interrogati sulle procedure per identificare geni putativi in una sequenza di DNA genomico, gli studenti che dichiaravano di conoscerle sono passati dal 24,9% (sperimentale) e 17,9% (controllo) nel pre-test al 74,6% e 74,7% rispettivamente nel post-test, con un incremento statisticamente significativo in entrambi i gruppi. Un'analisi del contenuto delle risposte, condotta sulla base di tre concetti chiave attesi: ottenere la sequenza di DNA bersaglio da un database, ricercare le Open Reading Frame, determinare quali ORF siano probabilmente geni tramite analisi BLAST. Questo ha mostrato che nel pre-test nessuno dei due gruppi menzionava questi concetti, facendo invece riferimento a tecniche di laboratorio "wet lab" già note, come l'elettroforesi e gli enzimi di restrizione (più frequenti nel gruppo sperimentale) o l'uso del codice genetico per identificare i codoni (più frequente nel gruppo di controllo). Nel post-test, il gruppo sperimentale ha mostrato percentuali più elevate di risposte contenenti almeno un concetto atteso (14,3% con un concetto, 11,1% con tutti e tre, contro il 9,0% e l'1,5% del gruppo di controllo) e una percentuale inferiore di risposte prive di qualunque concetto atteso (38,6% contro 47,7%); in entrambi i gruppi, inoltre, gli studenti hanno abbandonato il riferimento a tecniche di wet lab in favore di approcci bioinformatici nella progettazione dei propri disegni sperimentali, a testimonianza di come il laboratorio di bioinformatica abbia ridotto il divario tra la realtà scolastica degli studenti e le pratiche di ricerca attualmente diffuse.

Non tutte le misconcezioni indagate sono state tuttavia superate. Coerentemente con la definizione di "codogene" presente nel programma portoghese di undicesimo anno, un elemento che genera confusione nella definizione stessa di gene, gli studenti di entrambi i gruppi hanno continuato, sia nel pre- sia nel post-test, a non riconoscere l'esistenza di codoni di inizio alternativi e a ignorare l'esistenza di un codice genetico specifico per i batteri, nonostante il gruppo sperimentale avesse esplorato direttamente entrambi questi aspetti durante le attività pratiche: un risultato che gli autori collegano alla resistenza delle misconcezioni radicate, spesso originate da termini fuorvianti, spiegazioni semplificate ed esempi quotidiani usati per spiegare fenomeni biologici, anche a fronte di un insegnamento accurato e aggiornato. Per quanto riguarda genomica e genomica comparativa, nel post-test il 54,7% degli studenti del gruppo sperimentale (contro il 29,1% del gruppo di controllo) ha fornito una definizione corretta di genomica come campo di studio dei genomi; l'uso improprio dei termini "gene" e "genomica" come sinonimi è diminuito nel gruppo sperimentale

(dall'8,2% pre-test allo 0,5% post-test) mentre è lievemente aumentato nel gruppo di controllo (dall'1,5% al 5,1%), suggerendo che il solo insegnamento espositivo non sia stato sufficiente a chiarire la distinzione tra i due concetti. La genomica comparativa, essendo un concetto del tutto nuovo e quindi non condizionato da misconcezioni pregresse, è stata invece definita correttamente dalla maggioranza degli studenti di entrambi i gruppi nel post-test (79,5% sperimentale, 75,3% controllo), a riprova dell'efficacia, in questo caso, del solo insegnamento espositivo, con una nuova concezione, il confronto tra i genomi di due o più specie per indagare le relazioni filogenetiche, emersa nel 6,7% del gruppo sperimentale e nel 5,2% del gruppo di controllo.

Nel complesso, lo studio documenta un miglioramento delle conoscenze relative a gene, sintesi proteica, acidi nucleici, codoni di inizio e di stop, genoma, relazioni evolutive e genomica, oltre a una maggiore consapevolezza delle applicazioni e delle potenzialità della bioinformatica, confermando il valore educativo dell'integrazione di attività pratiche nei programmi di scienze e aprendo interrogativi, per la ricerca futura, sulle misconcezioni non risolte e sulla possibile estensione dell'approccio ad altri livelli scolastici e ad altri contenuti curricolari.

In conclusione, prendendo in considerazione questi studi di cui teniamo conto, si osserva come il pensiero computazionale abbia un impatto positivo se insegnato già da prima di un percorso universitario. Nei bambini si osserva come il pensiero computazionale non solo aiuti per i concetti di logica, ma anche in contesti sociali; inoltre anche l’adolescenza può essere considerata una buona fascia d’età per impararlo, anche insieme al concetto di bioinformatica. In questi studi è importante considerare come determinati insegnamenti siano stati utili, partendo da tecniche “unplugged” e “plugged”, da metodi di scaffolding per un maggior supporto, video tutorial e strumenti adatti come scratch.

Conclusioni

Riassumendo l'obiettivo principale è stato capire come il pensiero computazionale e le nozioni di logica influenzassero sia nello sviluppo, sia il mondo delle materie STEM e delle biotecnologie.

Diversi studi sono stati condotti sulla plasticità neuronale e su come viene condizionata con lo studio del pensiero computazionale. In particolare, in tutti gli studi presi in evidenza non si hanno mai dei riscontri negativi, ma sempre positivi o poco influenti; si è, inoltre, riscontrato che si evidenzia un maggiore miglioramento in giovane età. Nelle fasce d'età più giovani si nota che non si ha un miglioramento non solo dal punto di vista logico, ma anche doti sociali e comunicative. Si deve prendere in considerazione che la definizione di pensiero computazionale non si intende solo a livello logico, bensì si collega a una definizione più ampia, che può essere usata in tutti i contesti, compresa la creatività.

L'insegnamento è per l'appunto rilevante, soprattutto prendendo in considerazione quanto sia collegato con le materie STEM e con le biotecnologie. Nelle materie STEM è fondamentale avere delle conoscenze logiche e matematiche strettamente collegate con esso e con l'informatica. Nelle biotecnologie per alcuni nuovi metodi di sviluppo è necessaria qualche conoscenza, come nel bioprocessing in cui è vantaggioso che l'utente possa controllare l'espressione genica in modo analogico, ed è attualmente possibile implementare circuiti analogici che possono rilevare determinati cambiamenti nelle cellule. Considerando la complessità delle reti metaboliche, i circuiti genetici a feedback rappresentano strumenti promettenti per contenere l'eterogeneità metabolica, sostenere la cooperazione tra microrganismi e migliorare la produttività cellulare.

Per quanto riguarda le scuole in Italia si sta tenendo in conto di integrare maggiormente l'informatica già a partire dalle elementari e dalle medie, sarà presente una parte di educazione civica dedicata all'informatica, ma attualmente non è necessariamente presente una materia dedicata nell'orario scolastico.

Per le scuole superiori la questione varia molto in base al tipo di istituto che si frequenta, per esempio in un istituto tecnico dedicato all'informatica ci saranno diverse materie dedicate che copriranno buona parte dell'orario. Sono inoltre presenti degli istituti tecnici di biotecnologie, dove generalmente l'informatica è presente solo il primo anno, sia con attività di laboratorio, sia lezioni teoriche; nonostante sia presente solo il primo anno è importante ricordare che verrà integrato in altre materie negli anni successivi. Per quanto riguarda i licei non tutti hanno un corso dedicato, ma tre questi si può citare il liceo scientifico.

A proposito del suo impatto, come precedentemente accennato, si nota una maggiore importanza del pensiero computazionale principalmente nei bambini e negli adolescenti, mentre in età adulta la sua influenza sembra minima. Oltre l'età infantile, infatti, sembra che l'adolescenza risulti un periodo rilevante per l'apprendimento di nuove informazioni di logica e di informatica.

Per trarre le dovute conclusioni si sono analizzate diverse ricerche nella quale si insegnava nelle scuole diversi concetti, tra cui:

- il pensiero computazionale;
- la bioinformatica;
- l'uso di programmi come Scratch e MakeCode.

Sono stati considerati anche diversi metodi di apprendimento (tenendo solitamente conto di un gruppo di controllo), come metodi plugged e unplugged, metodi di scaffolding, lavori collaborativi di coppia, video tutorial, dei pre-test e post-test.

Citazioni

[1] Hongo T., Yakou T., Yoshinaga K., Kano T., Miyazaki M., Hanakawa T. (2022). Structural neuroplasticity in computer programming beginners. Oxford Academic 33, 5375–5381 https://doi.org/10.1093/cercor/bhac425

[2] Margetis K, Vadakekut ES. Neuroplasticity. (2026). Neuroplasticity. StatPearls. https://www.ncbi.nlm.nih.gov/books/NBK557811/

[3] Tymofiyeva O., Gaschler R. (2021). Training-Induced Neural Plasticity in Youth: A Systematic Review of Structural and Functional MRI Studies. Fronties in Human Neuroscience 14, 497245. https://doi.org/10.3389/fnhum.2020.497245

[4] Li, Z., Oon, P.T. (2024). The transfer effect of computational thinking (CT)-STEM: a systematic literature review and meta-analysis. International Journal of STEM Education 11, 44. https://doi.org/10.1186/s40594-024-00498-z

[5] Balovsyak S., Derevyanchuk O., Kovalchuk V., Kravchenko H., Ushenko Y., Hu Z. (2024). STEM Project for Vehicle Image Segmentation Using Fuzzy Logic. MECS Press 16, 2, 45-57. https://doi.org/10.5815/ijmecs.2024.02.04

[6] Lv Y., Qian S., Du G., Chen J., Zhou J., Xu P. (2019). Coupling feedback genetic circuits with growth phenotype for dynamic population control and intelligent bioproduction. ELSEVIER 54, 109, 116 https://doi.org/10.1016/j.ymben.2019.03.009

[7] Xiang, Y., Dalchau, N. & Wang, B. (2018). Scaling up genetic circuit design for cellular computing: advances and prospects. Natural Computing 17, 833–853 https://doi.org/10.1007/s11047-018-9715-9

[8] Govender, I. (2022). Research Status in Computational Thinking in STEM Education. In Advances in Research in STEM Education. IntechOpen. https://doi.org/10.5772/intechopen.104472

[9] Marangoni R., Bevilacqua V., Cannataro M., Mele BH, Mauri G., Marabotti A. (2023). An overview of bioinformatics courses delivered at the academic level in Italy: Reflections and recommendations from BITS. PLoS computational biology, 19(2), e1010846. https://doi.org/10.1371/journal.pcbi.1010846

[10] Consorzio Interuniversitario Nazionale per l'informatica [CINI] (2017). Proposta di indicazioni nazionali per l'insegnamento dell'informatica nella scuola. https://www.consorzio-cini.it/index.php/it/component/attachments/download/745

[11] Ministero dell'istruzione e del merito (2022). Piano Scuola 4.0. https://pnrr.istruzione.it/wp-content/uploads/2022/07/PIANO_SCUOLA_4.0_VERSIONE_GRAFICA.pdf

[12] Suherman, S., Vidákovich, T., Mujib, M., Hidayatulloh, H., Andari, T., & Susanti, V. D. (2025). The Role of STEM Teaching in Education: An Empirical Study to Enhance

Creativity and Computational Thinking. Journal of Intelligence, 13(7), 88. https://doi.org/10.3390/jintelligence13070088

[13] Sánchez-Pérez, N., Inuggi, A., Castillo, A., Campoy, G., García-Santos, J. M., González-Salinas, C., & Fuentes, L. J. (2019). Computer-Based Cognitive Training Improves Brain Functional Connectivity in the Attentional Networks: A Study With Primary School-Aged Children. Frontiers in behavioral neuroscience, 13, 247. https://doi.org/10.3389/fnbeh.2019.00247

[14] Tavares F. (2020). Bioinformatics-Based Activities in High School: Fostering Students' Literacy, Interest, and Attitudes on Gene Regulation, Genomics, and Evolution. Frontiers in Microbiology 11:578099. https://doi.org/10.3389/fmicb.2020.578099

[15] Perez Valdes GP, Boude Figueredo O and Vargas Sanchez AD (2025). Integrating computational thinking in children aged 3 to 6: challenges and opportunities in early childhood education. Front. Educ. 10:1535135. https://doi.org/10.3389/feduc.2025.1535135

[16] Chen, L., Chang, H., Rudoler, J. et al. (2022). Cognitive training enhances growth mindset in children through plasticity of cortico-striatal circuits. npj Sci. Learn. 7, 30. https://doi.org/10.1038/s41539-022-00146-7

[17] Robledo-Castro C., Castillo-Ossa LF, Hederich-Martínez C. (2025) Computational thinking training and its effects on working memory, flexibility, and inhibition: Randomized controlled trial in fifth-grade children. ELSEVIER 100, 101850 https://doi.org/10.1016/j.appdev.2025.101850

[18] Robledo-Castro C., Castillo-Ossa LF, Hederich-Martínez C. (2023) Effects of a computational thinking intervention program on executive functions in children aged 10 to 11. ELSEVIER 35,100563 https://doi.org/10.1016/j.ijcci.2022.100563

[19] Fandakova Y., Hartley CA. (2020) Mechanisms of learning and plasticity in childhood and adolescence. ELSEVIER 42,100764 https://doi.org/10.1016/j.dcn.2020.100764

[20] Scionti N, Cavallero M, Zogmaister C and Marzocchi GM (2020) Is Cognitive Training Effective for Improving Executive Functions in Preschoolers? A Systematic Review and Meta-Analysis. Frontiers in Psychology 10:2812. https://doi.org/10.3389/fpsyg.2019.02812

[21] Donenfeld J., Mudundi M., Blaser E., Kaldy Z. (2026) School changes minds: A meta-analysis shows that schooling modestly improves children's executive functions ELSEVIER 262, 106371 https://doi.org/10.1016/j.jecp.2025.106371

[22] Bain, S. A., Plaisier, H., Anderson, F., Cook, N., Crouch, K., Meagher, T. R., Ritchie, M. G., Wallace, E. W. J., & Barker, D. (2022). Bringing bioinformatics to schools with the 4273pi project. PLoS computational biology, 18(1), e1009705. https://doi.org/10.1371/journal.pcbi.1009705

[23] Dietrichson, J., Thomsen, M. K., Seerup, J. K., Strandby, M. W., Viinholt, B. C. A., & Bengtsen, E. (2022). PROTOCOL: School-based language, math, and reading interventions for executive functions in children and adolescents: A systematic review. Campbell systematic reviews, 18(3), e1262. https://doi.org/10.1002/cl2.1262