Macchine di Turing e Calcolabilità

Cap. 6 dispensa

Jocelyne Elias
https://www.unibo.it/sitoweb/jocelyne.elias/
Moreno Marzolla
https://www.moreno.marzolla.name/
Dipartimento di Informatica—Scienza e Ingegneria (DISI)
Università di Bologna

Ringraziamenti

- Parte del materiale presente in queste slide è tratto dalla presentazione *Calcolo e Simboli: la scienza digitale* del prof. Simone Martini, Università di Bologna

Alan Mathison Turing (1912—1954)

- Nato il 23 giugno 1912
- Morto il 7 giugno 1954 per avvelenamento da cianuro
  - Probabilmente suicida a causa delle persecuzioni subite per la sua omosessualità
- Contributi fondamentali in
  - Criptanalisi
  - Macchine per il calcolo
  - Gioco dell'imitazione
  - …

Possono le macchine pensare?

VOL. LIX. No. 236.] [October, 1950

MIND

A QUARTERLY REVIEW

OF

PSYCHOLOGY AND PHILOSOPHY

I.—COMPUTING MACHINERY AND INTELLIGENCE

BY A. M. TURING

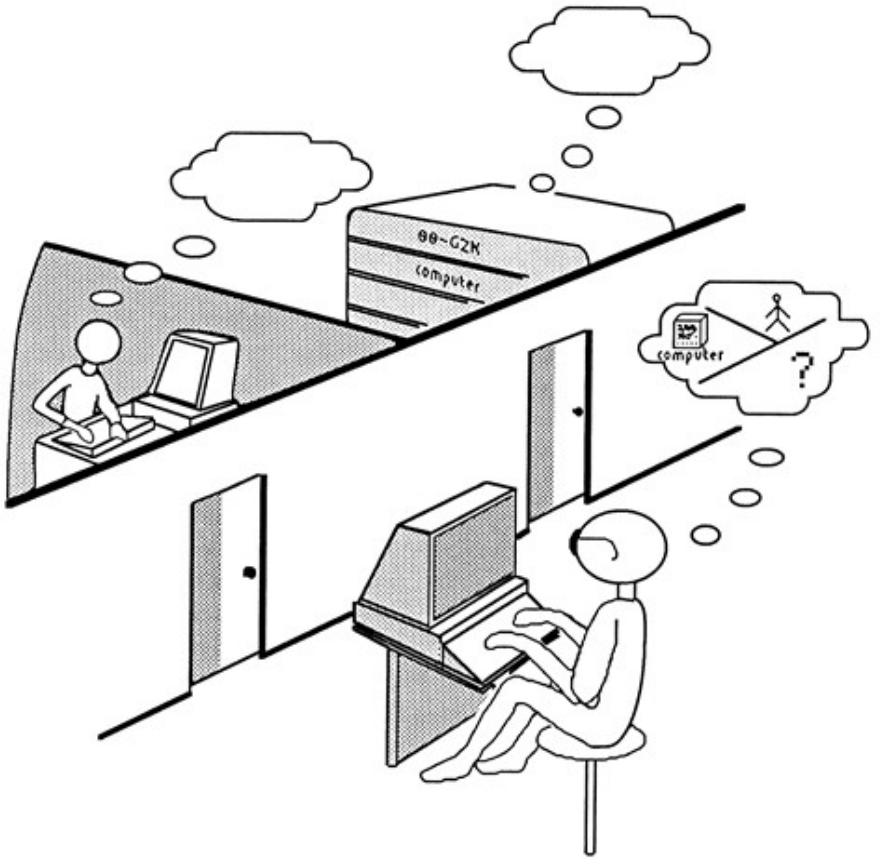
1. *The Imitation Game.*

I PROPOSE to consider the question, 'Can machines think?' This should begin with definitions of the meaning of the terms 'machine' and 'think'. The definitions might be framed so as to reflect so far as possible the normal use of the words, but this

Il gioco dell'imitazione

Cos'è l'intelligenza?

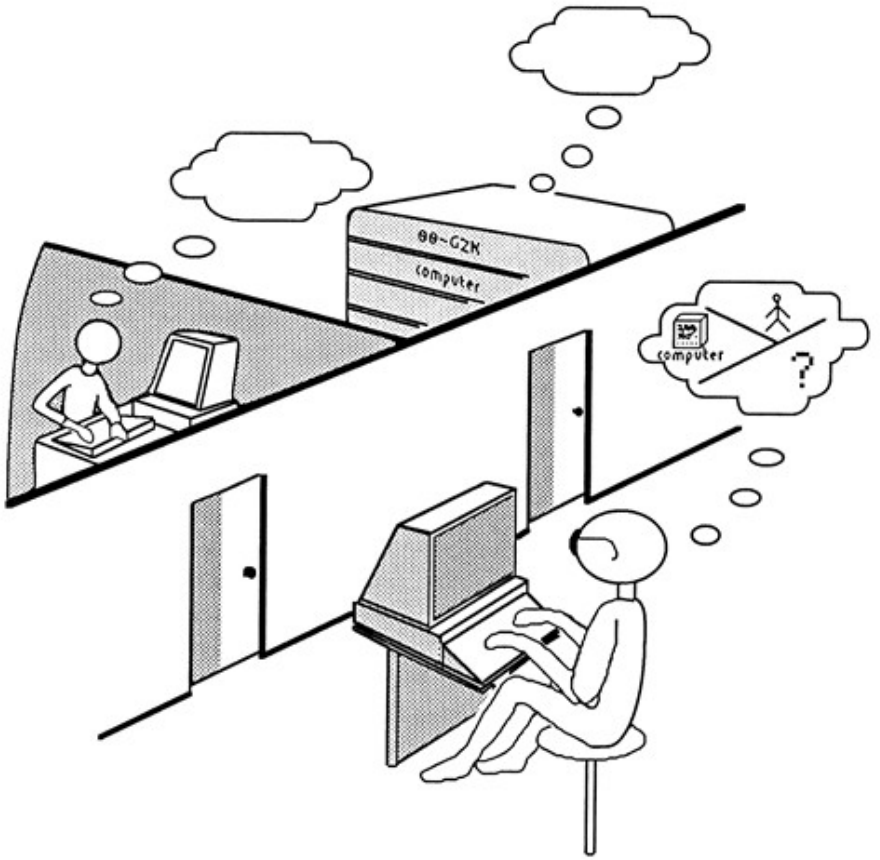
- Un interrogante è collegato mediante un terminale con due stanze
- In una di esse si trova una persona, nell'altra un computer
- Ponendo domande tramite il terminale, l'interrogante deve decidere chi è l'essere umano e chi la macchina



https://academic.oup.com/mind/article/LIX/236/433/986238

Il gioco dell'imitazione

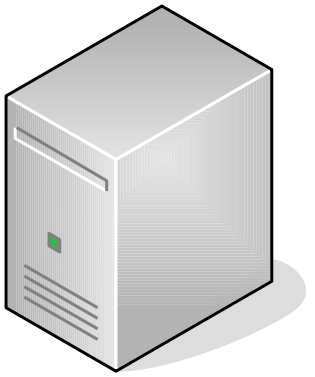
- Scopo della macchina è di farsi passare per un essere umano
- Se l'interrogante non è in grado di identificare in modo affidabile chi è l'umano e chi è il computer, la macchina ha superato il Test di Turing



Un ipotetico dialogo, descritto da Turing

Mi scriva un sonetto sul tema del ponte sul Forth

?



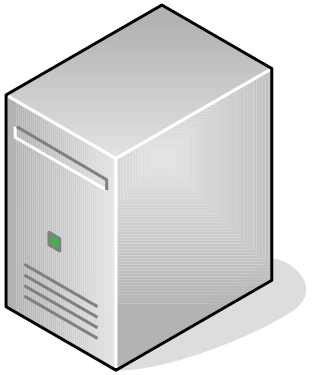
...il ponte sul Forth

Un ipotetico dialogo, descritto da Turing

Mi scriva un sonetto sul tema del ponte sul Forth

Non faccia affidamento a me su questo. Non ho mai saputo scrivere poesie

?



Sommi 34957 a 70764

Pausa di circa 30 secondi...

105621 ✗ ⇨ 105721

Gioca a scacchi?

Sì

Ho il re in e1 e nessun altro pezzo. Lei ha solo il re in c3 e una torre in h8. Tocca a lei. Che mossa fa?

Pausa di 15 secondi...

Torre in h1. Scacco matto. ✗

Macchine di Turing e calcolabilità

Premessa:

le macchine di Turing non c'entrano con il gioco dell'imitazione!!

Cosa significa calcolare?

- Alan M. Turing, *On Computable Numbers, with an Application to the Entscheidungsproblem*, Proc. Lond. Math. Soc. (2) 42 pp. 230-265 (1936)
  - *Entscheidungsproblem* (secondo Problema di Hilbert): *Esiste un metodo definito di calcolo che, data una proposizione matematica, permetta di decidere se è o non è dimostrabile*?
- Turing ha iniziato a porsi domande sulla natura e i limiti dei computer molto prima che questi esistessero
  - Primo calcolatore "universale" elettromeccanico: Z3, 1941; Konrad Zuse, Germania.
  - Primo calcolatore "universale" elettronico: ENIAC, 1946; University of Pennsylvania's Moore School of Electrical Engineering

Cosa significa calcolare?

		1	7	5	+		
			9	2	=		
		2	6	7			

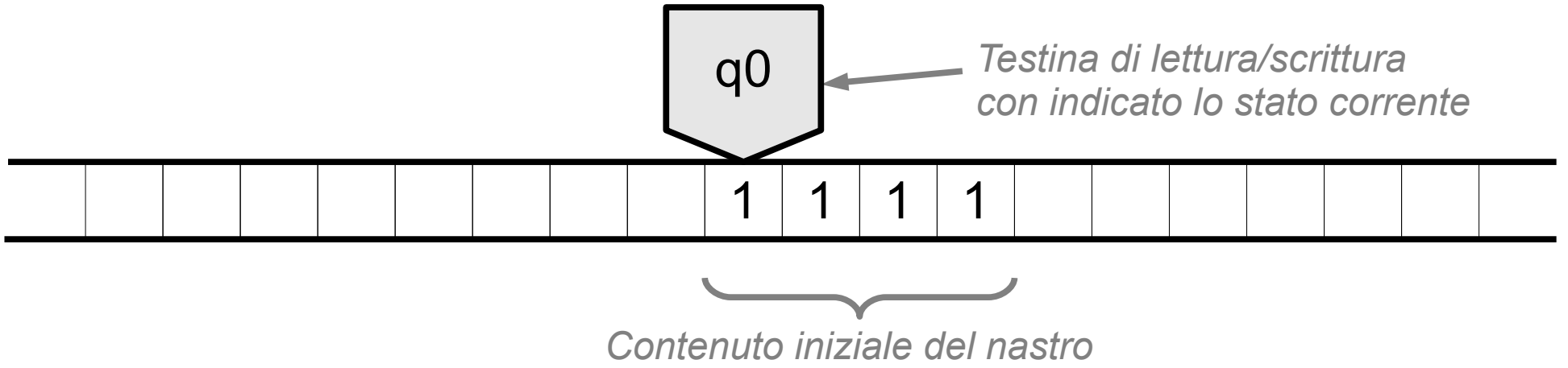
Macchina di Turing (MdT)

- Un nastro infinito in entrambe le direzioni, diviso in celle
- Un dispositivo di lettura/scrittura
  - Posizionato su una delle celle, si può spostare di una sola posizione alla volta verso destra o verso sinistra
- Un alfabeto finito
  - L'alfabeto include un simbolo “spazio” (*blank*) che compare su tutte le (infinite) celle non inizializzate del nastro.
  - Un insieme finito di celle può contenere inizialmente simboli diversi da *blank*, e rappresentano l'input della macchina di Turing
- Un insieme finito di stati in cui la macchina può trovarsi
  - q0 è lo stato iniziale, in cui la macchina si trova quando viene fatta partire
  - halt è lo stato finale: se la macchina raggiunge tale stato, si ferma.
  - Gli stati possono avere nomi arbitrari; “q0” e “halt” sono una convenzione
- Una funzione di transizione che determina il comportamento della macchina

Funzione di transizione

- Una lista di regole che indicano, per ogni possibile simbolo *X* che si trovi al momento sotto la testina e per ogni possibile stato *P* della macchina:
  - quale simbolo *Y* scrivere al posto di *X* (è possibile riscrivere nuovamente *X*);
  - qual è il nuovo stato *Q* della macchina (è possibile che lo stato rimanga sempre *P*);
  - se la testina deve essere spostata a destra oppure a sinistra di una casella;
- Inizialmente la macchina si trova nello stato iniziale q0, e la testina è posizionata su una data cella del nastro (che di solito dipende dal problema da risolvere)

Esempio 1

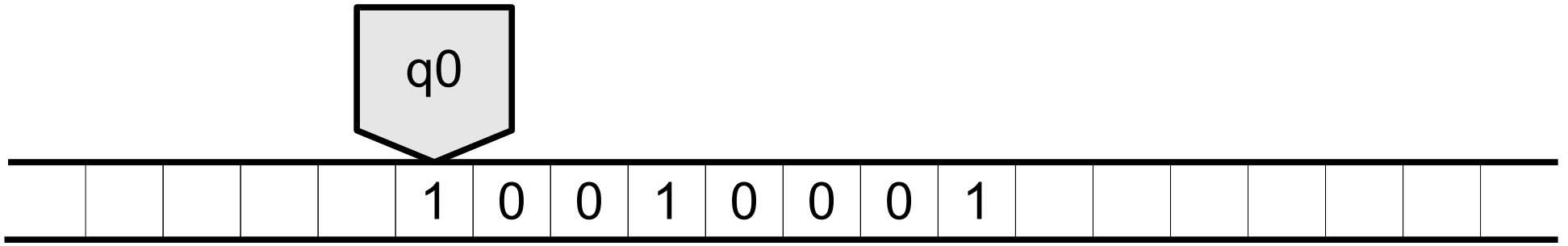


- Alfabeto: {1, *blank*}
- Stati: {q0, halt}
- Funzione di transizione (tavola di istruzioni):

Stato corrente	Simbolo corrente	Nuovo Simbolo	Nuovo Stato	Spostamento
q0	1	1	q0	right
q0	*blank*	1	halt	---

Stato Corrente	Simbolo Corrente	Nuovo Simbolo	Nuovo Stato	Spostamento
q0	1	0	q0	right
q0	0	1	q0	right
q0	blank	---	halt	----

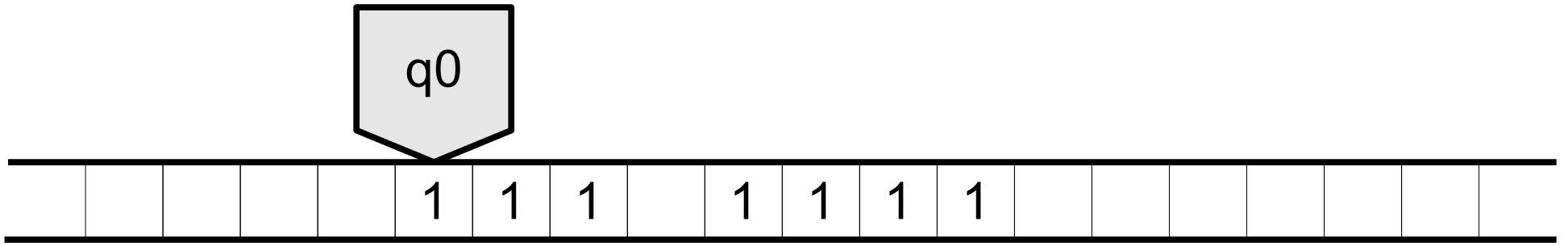
Esempio 2



- Calcolare il “complemento a uno” di un numero espresso in base 2
  - In pratica, cambiare gli '1' con '0' e viceversa
  - Es. 10010001 → 01101110
- Inizialmente il dispositivo di lettura/scrittura si trova sulla prima cifra a sinistra
- Al termine il numero originale deve essere sovrascritto con il suo complemento ad uno
- Esercizio: come definiamo la *funzione di transizione*?

Stato Corrente Simbolo Corrente Nuovo Simbolo Nuovo Stato Spostamento
q0
q0

Esempio 3



- Calcolare la somma di due numeri in base 1
  - Es. 111 + 1111 = 1111111
- Inizialmente sul nastro vengono scritti i due numeri da sommare, separati da un *blank*
- Al termine il nastro deve contenere il risultato al posto degli input
- Esercizio: come definiamo la *funzione di transizione*?

Funzioni calcolabili e Turing-Equivalenza

- Tesi di Church-Turing: le funzioni calcolabili sono tutte e sole le funzioni calcolabili da una macchina di Turing
- Qualsiasi formalismo mediante il quale sia possibile simulare una macchina di Turing si dice *Turing-Equivalente*
  - Tutti i linguaggi di programmazione esistenti sono Turing-Equivalenti
  - Dalla tesi di Church-Turing deriva che i programmi esprimibili con i linguaggi di programmazione esistenti possono calcolare tutte e sole le funzioni calcolabili da una MdT

Funzioni non calcolabili: Halting Problem

- Esistono funzioni NON calcolabili da una MdT
  - e quindi, per la tesi di Church-Turing, da nessun formalismo di calcolo
- Esempio (Halting Problem)

  "*Scrivere una funzione che, data in input la descrizione di una MdT e il contenuto iniziale del nastro, restituisce TRUE se la MdT termina (cioè se prima o poi raggiunge lo stato **halt**), FALSE altrimenti*"

Funzioni non calcolabili: Halting Problem

- Esistono funzioni NON calcolabili da una MdT
  - e quindi, per la tesi di Church-Turing, da nessun formalismo di calcolo
- Esempio (Halting Problem, formulazione equivalente)

  "*Scrivere una funzione che accetta in input (1) la descrizione di un programma **P** scritto in una opportuna notazione, e (2) l'input **I** da passare a **P**. La funzione deve restituire TRUE se e solo se il programma **P** applicato all'input **I** termina*"

Halting problem
Dimostrazione semi-formale

- Supponiamo per assurdo che sia possibile scrivere un programma che accetta in input due stringhe
  - ***P*** rappresenta il sorgente di un programma in C
  - ***I*** rappresenta l'input su cui il programma ***P*** deve operare

```
boolean Termina(String P, String I)
{
    if (P applicato a I termina) {
        return true;
    } else {
        return false;
    }
}
```

Qui dovrebbe esserci del codice “opportuno” per decidere se P(I) termina o no

Halting problem
Dimostrazione semi-formale

- Se è possibile scrivere la funzione **`Termina()`**, allora è possibile anche scrivere la funzione **`Bomba()`** seguente:
  - Se **`P(P)`** termina, **`Bomba(P)`** NON termina
  - Se **`P(P)`** non termina, **`Bomba(P)`** termina e ritorna 0

```
boolean Bomba(String P)
{
    if (Termina(P, P)) {
        while (true) ; /* ciclo infinito */
        return true;
    } else {
        return false;
    }
}
```

Halting problem
Dimostrazione semi-formale

- Che cosa possiamo dire dell'invocazione **`Bomba(`***Bomba***`)`**?
  - Cioè passiamo alla funzione **`Bomba()`** il suo stesso codice
- **`Bomba(`***Bomba***`)`** può terminare, o andare in loop
  - Se Bomba(Bomba) termina, significa che Termina(Bomba, Bomba) restituisce 0 (*falso*)
  - Cioè Bomba(Bomba) NON termina

***Assurdo*!**

```
boolean Bomba(String P)
{
    if (Termina(P, P)) {
        while (true) ;
        return true;
    } else {
        return false;
    }
}
```

Halting problem
Dimostrazione semi-formale

- Che cosa possiamo dire dell'invocazione **`Bomba(`***Bomba***`)`**?
  - Cioè passiamo alla funzione **`Bomba()`** il suo stesso codice
- **`Bomba(`***Bomba***`)`** può terminare, o andare in loop
  - Se Bomba(Bomba) NON termina, significa che Termina(Bomba, Bomba) restituisce 1 (*vero*)
  - Cioè Bomba(Bomba) termina

Assurdo!

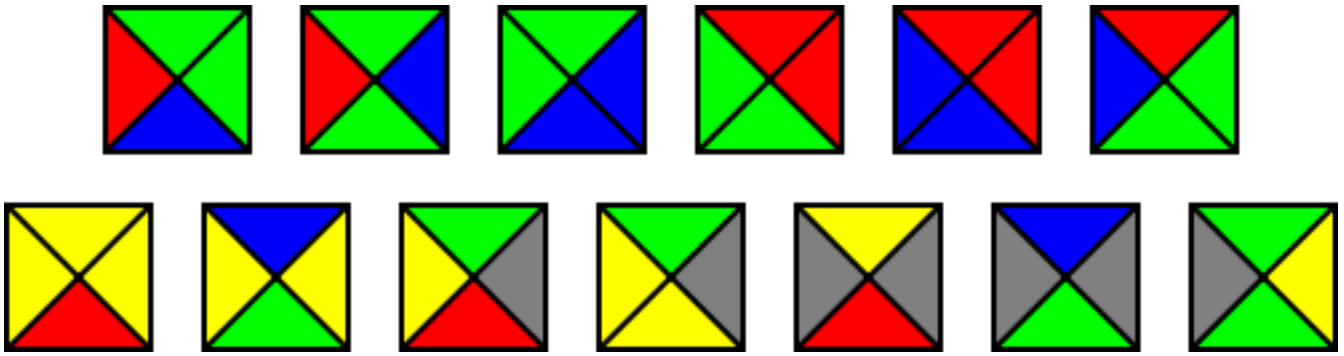
```
boolean Bomba(String P)
{
    if (Termina(P, P)) {
        while (true) ;
        return true;
    } else {
        return false;
    }
}
```

Halting problem
Dimostrazione semi-formale

- Quindi supponendo che fosse possibile scrivere la funzione **`Termina()`**, abbiamo ottenuto un assurdo
- Di conseguenza NON possiamo definire una funzione **`Termina()`** che operi come indicato
  - Nemmeno se usiamo un linguaggio di programmazione diverso: tutti i linguaggi di programmazione sono equivalenti ad una macchina di Turing, quindi possono calcolare solo le funzioni che una macchina di Turing può calcolare

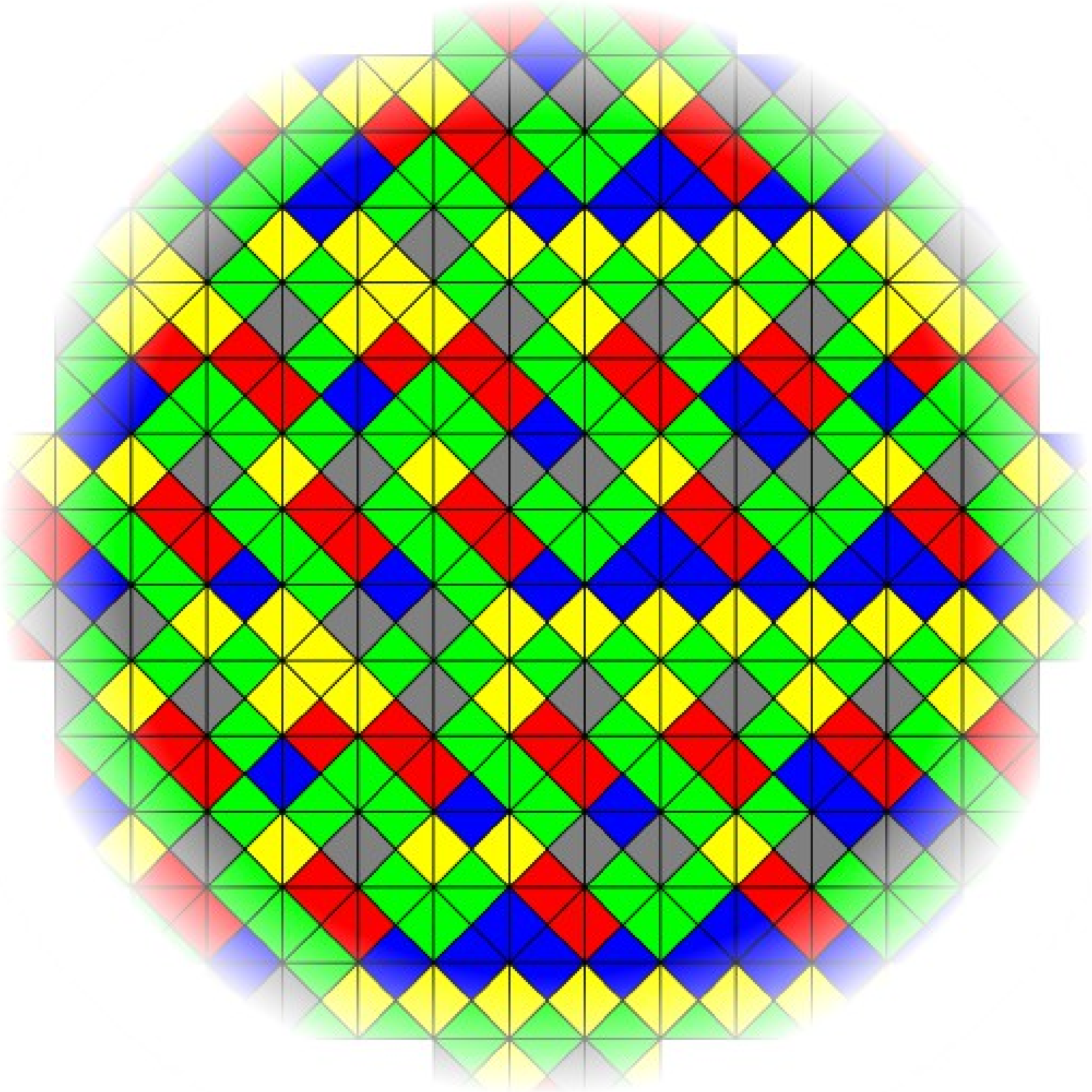
Funzioni non calcolabili: Tassellatura di Wang

- Dati ***k*** tipi di piastrelle di Wang, decidere se possono ricoprire il piano
  - È possibile usare un numero illimitato di piastrelle di ciascun tipo
  - Le piastrelle non possono essere ruotate
- Esempio: le piastrelle seguenti possono ricoprire il piano (in modo non periodico)



By Anomie - Own work, Public Domain, https://commons.wikimedia.org/w/index.php?curid=3968914

- È stato dimostrato che è impossibile definire un algoritmo per decidere se un insieme dato di tipi di piastrelle può ricoprire il piano

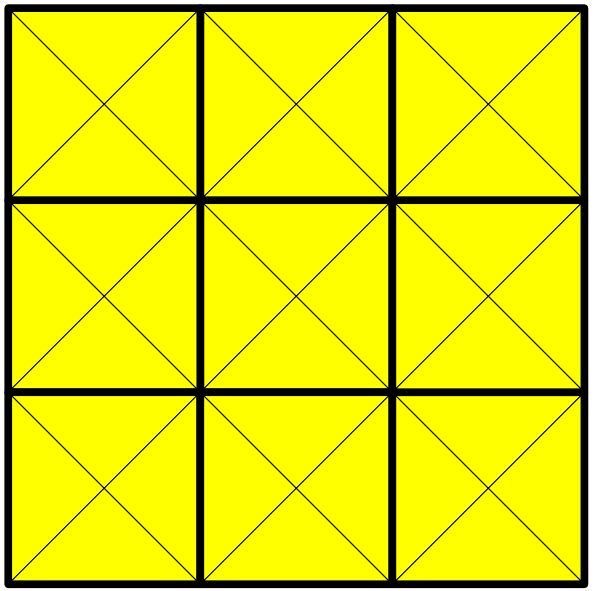


Tassellatura di Wang

- Il fatto che il problema della tassellatura di Wang non sia calcolabile non significa che non siano note soluzioni in alcuni casi particolari
- Significa però che è impossibile definire una procedura generale in grado di decidere se un dato insieme di tasselli può ricoprire il piano, qualunque sia l'insieme di tasselli

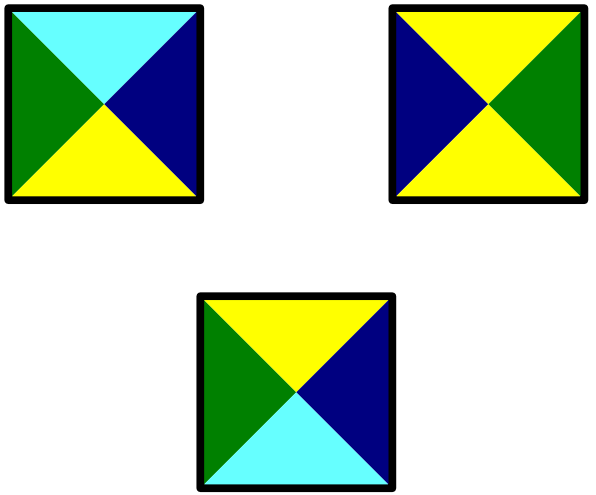
Esempio

- È facile dimostrare che con una piastrella con tutti i lati dello stesso colore si può ricoprire il piano



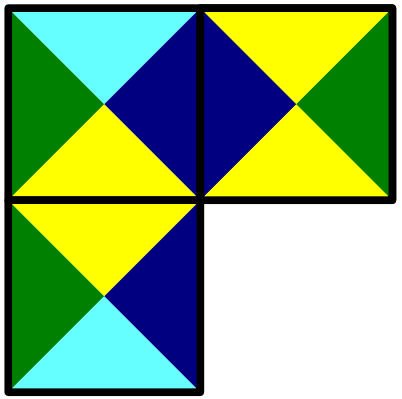
Esempio

- Altri casi semplici si possono decidere “a occhio”



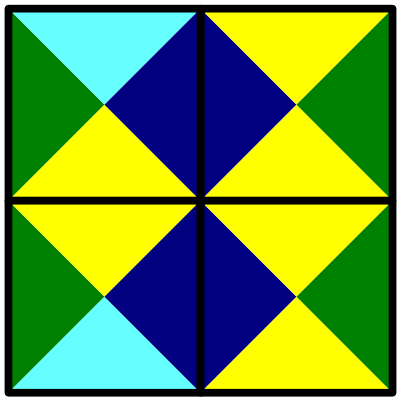
Esempio

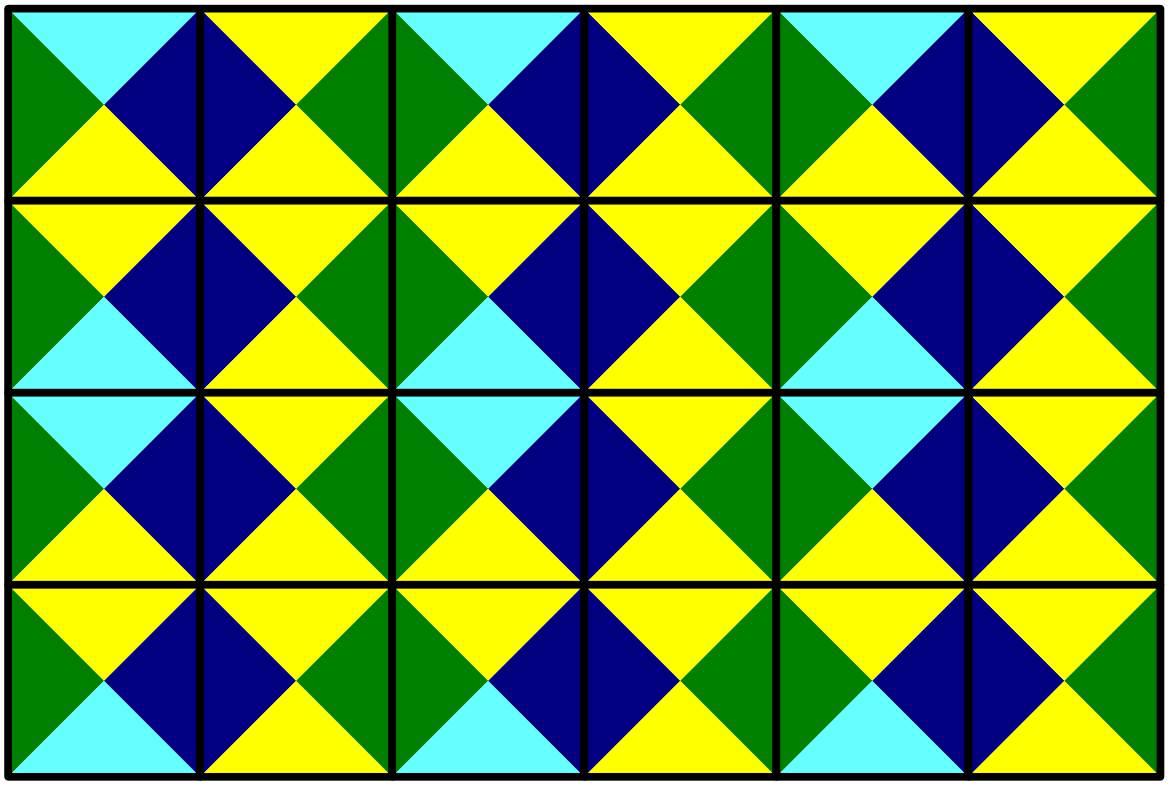
- Altri casi semplici si possono decidere “a occhio”



Esempio

- Altri casi semplici si possono decidere “a occhio”

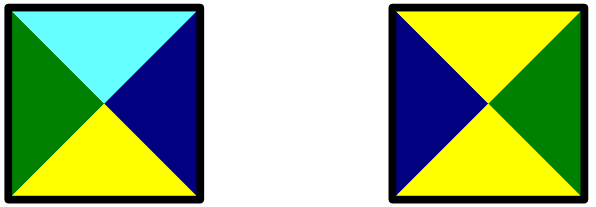




...

Esempio

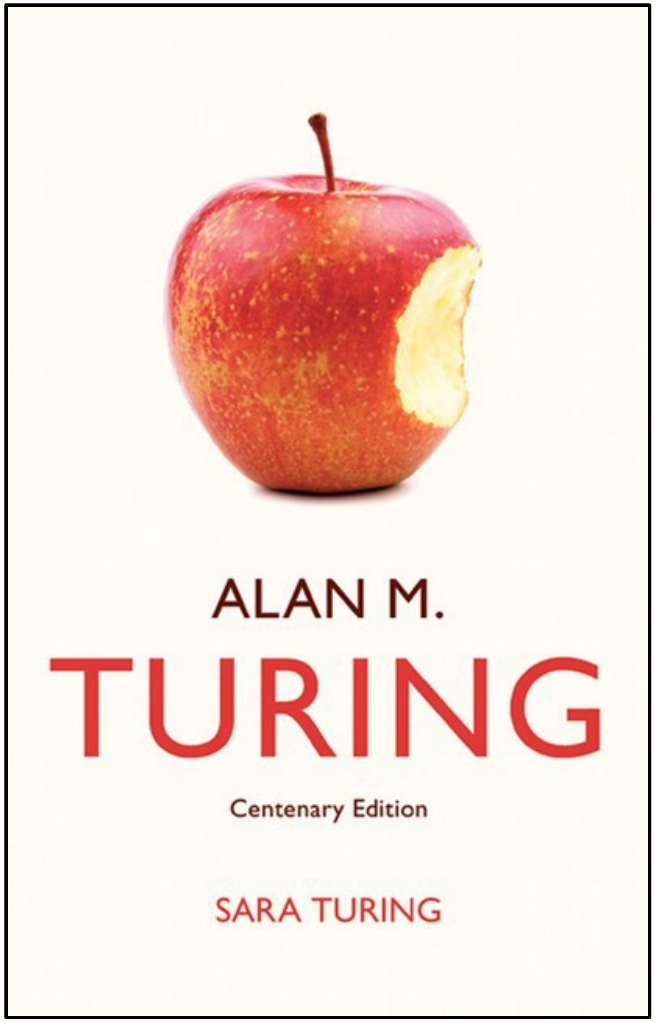
- Altri casi semplici si possono decidere “a occhio”

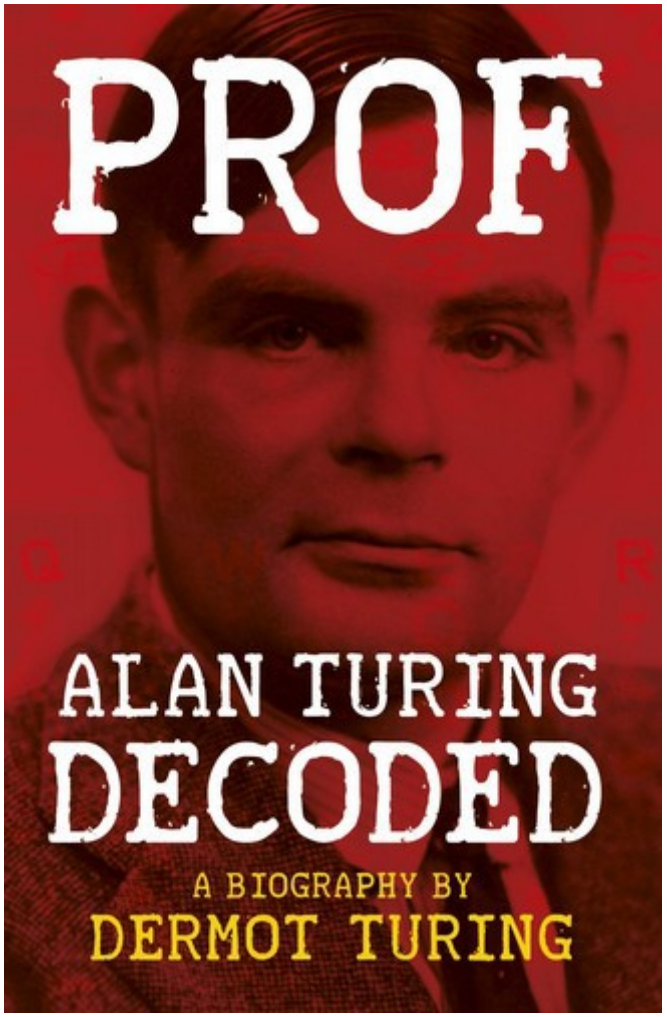


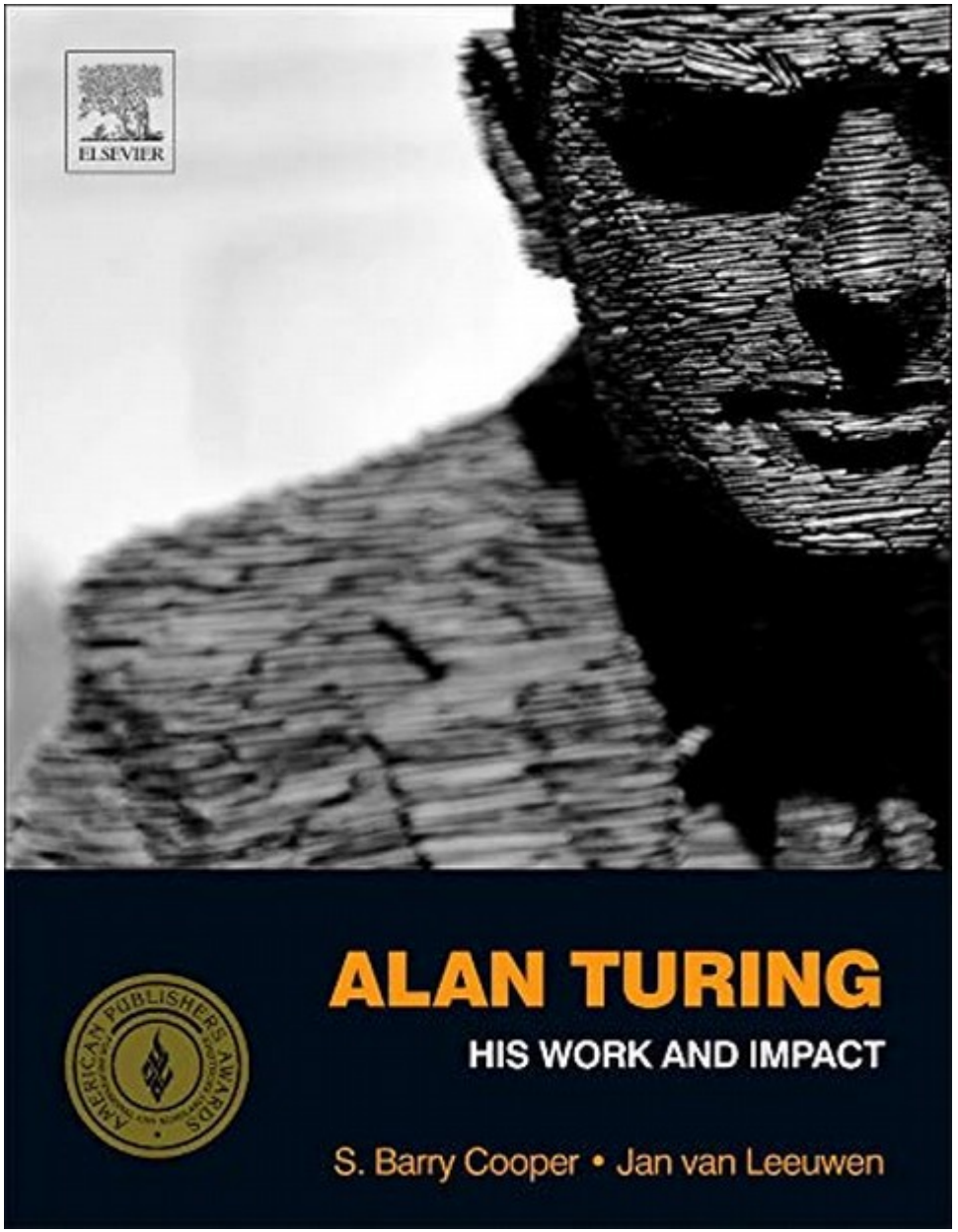
- Es: 1- il tassello di sinistra non lo posso usare
  - altrimenti non avrei nulla da mettergli sopra

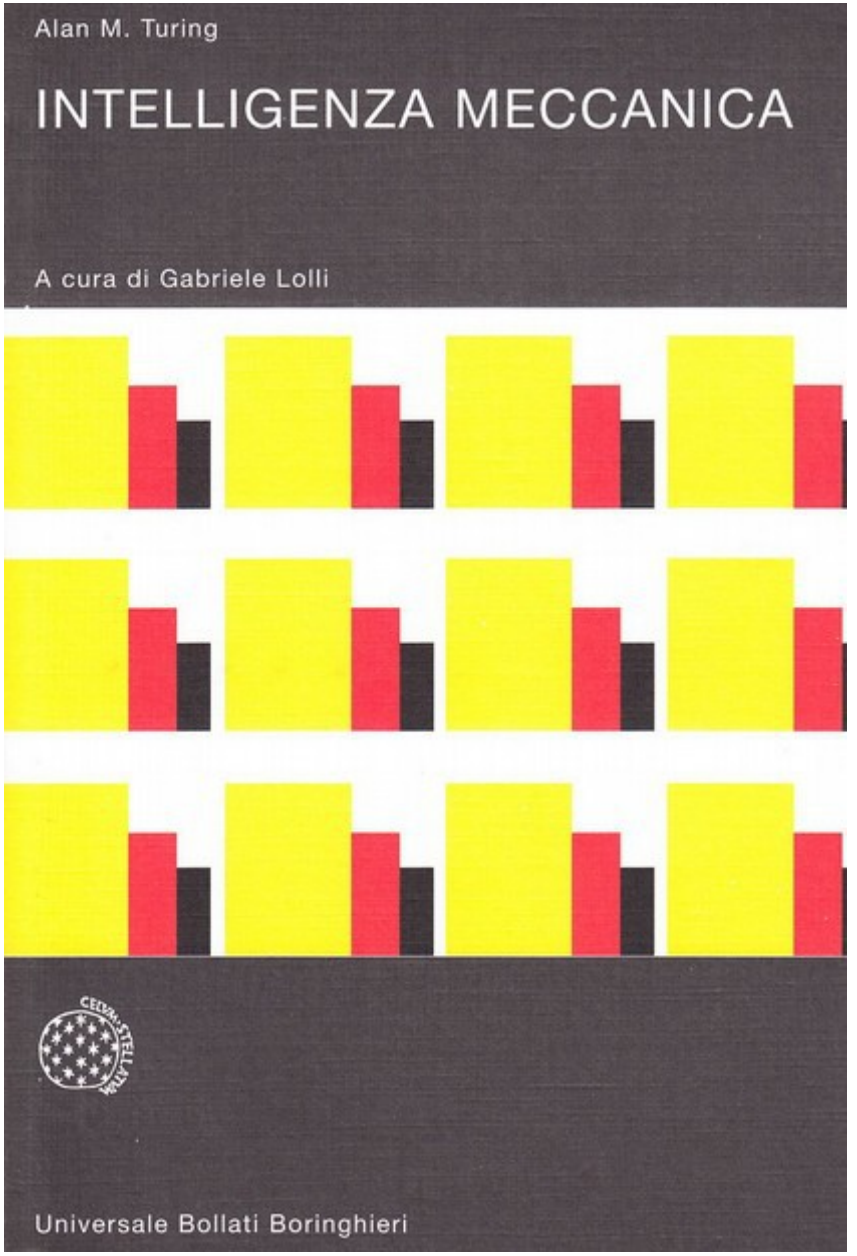
  2- Il tassello di destra non lo posso usare
  - altrimenti non avrei nulla da mettergli a fianco; di certo non quello di sinistra, per il punto precedente...


LA STORIA VERA
DI UNA MENTE STRAORDINARIA
ANDREW HODGES
ALAN TURING
STORIA
DI UN ENIGMA
IL LIBRO CHE HA ISPIRATO IL FILM
THE IMITATION GAME
Bollati Boringhieri


ALAN M.
TURING
Centenary Edition
SARA TURING


PROF
ALAN TURING
DECODED
A BIOGRAPHY BY
DERMOT TURING


ELSEVIER
ALAN TURING
HIS WORK AND IMPACT
S. Barry Cooper • Jan van Leeuwen


Alan M. Turing
INTELLIGENZA MECCANICA
A cura di Gabriele Lolli
Universale Bollati Boringhieri