

# Introduzione alla teoria della percolazione

Leonardo Ongari *Università degli studi di Parma*  
Cremona, Italia  
leonardo.ongari@studenti.unipr.it

**Sommario**—La teoria della percolazione si occupa di sviluppare modelli matematici per descrivere il fenomeno fisico della percolazione, caratterizzato dallo scorrimento di un *fluido* all'interno di un *mezzo*. In questa relazione vengono mostrate le caratteristiche principali dei modelli, le nozioni teoriche associate ed alcune analisi sugli algoritmi utilizzabili per effettuare simulazioni consistenti.

**Keywords**—Percolazione, Modelli, Simulazioni

## I. INTRODUZIONE

La teoria della percolazione nasce con l'obiettivo di ottenere un modello matematico per descrivere il fenomeno fisico della percolazione. Questo fenomeno descrive lo scorrimento di un fluido all'interno di un materiale tipicamente poroso. È importante specificare che il significato del termine “percolazione” può riferirsi a diversi contesti, a seconda del campo in cui si sta operando. Alcuni esempi sono:

- un soluto che diffonde attraverso un solvente;
- elettroni che migrano attraverso un reticolo atomico;
- molecole che penetrano un solido poroso;
- una malattia che infetta una comunità.

La formalizzazione matematica del problema ha reso possibile la creazione di un *modello*. Un punto chiave molto importante di un modello è il concetto di *astrazione*, caratteristica che permette di operare senza considerare alcune variabili dipendenti dal contesto [1].

## II. BACKGROUND

Un metodo efficace per l'astrazione del concetto consiste in una rappresentazione tramite un reticolo, in inglese “lattice”.

**Definizione 1** (Reticolo). Un reticolo  $L$  di dimensione  $n$  è una coppia  $\langle S, B \rangle$ , dove:

- $S = \{s_i : i \in \mathbb{N}, 0 \leq i < n\}$  è l'insieme dei siti;
- $B = \{(s_i, s_j) \in S^2 : s_i \text{ e } s_j \text{ siano primi vicini}\}$  è l'insieme dei legami.

In un reticolo in cui vale questa relazione è possibile descrivere due tipi di percolazione:

- percolazione di legame;
- percolazione di sito.

### Percolazione di legame

È la prima versione del modello fornita da Broadbent e Hammersley [1]. Ogni legame ha probabilità  $p$  di essere “aperto”, quindi probabilità  $1-p$  di essere “chiuso”. Se due siti formano un legame aperto, vi è una connessione diretta tra i due. Al contrario, un legame chiuso elimina la connessione. In

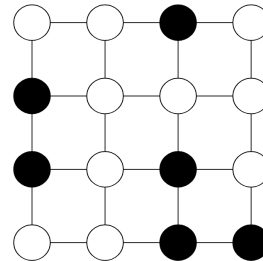


Figura 1: Esempio di reticolo quadrato bidimensionale

questa versione, “avviene percolazione” se esiste un percorso (insieme di connessioni dirette) che attraversa l'intero reticolo. La percolazione può verificarsi sia in verticale (alto-basso) sia in orizzontale (sinistra-destra).

### Percolazione di sito

In questo modello ogni sito ha una probabilità  $p$  di essere occupato, di conseguenza probabilità  $1-p$  di essere vuoto. In figura 1 viene mostrato un reticolo bidimensionale quadrato con nodi occupati (neri) e vuoti (bianchi). Questo è il modello che verrà utilizzato per lo studio dell'argomento e dei vari algoritmi e verrà approfondito in dettaglio nella sezione III.

### Soglia di percolazione

I due modelli appena introdotti rappresentano soluzioni valide per lo studio del fenomeno. Nonostante la somiglianza, vi sono differenze concettuali che si riflettono anche nel calcolo di alcuni valori caratteristici [2], [3], [4].

**Definizione 2** (Soglia di percolazione). Sia  $L$  un reticolo di dimensione infinita<sup>1</sup>. Sia  $p$  la probabilità di occupazione di un sito o di apertura di un legame, a seconda del modello scelto. Sia  $p$  uguale per ogni elemento del reticolo. La soglia di percolazione per  $L$  è definita come la probabilità  $p_c$  tale per cui:

- se  $p > p_c$ , allora vi è percolazione;
- se  $p < p_c$ , allora non vi è percolazione.

È possibile visualizzare il concetto di soglia nel grafico mostrato in figura 2, in cui l'asse delle ascisse è associato alla probabilità  $p$ , mentre l'asse delle ordinate è associato alla probabilità che avvenga percolazione  $p_{perc}$ .

<sup>1</sup>Con il termine “infinito” si fa riferimento all'estensione intuitiva delle varie proprietà della struttura, come avviene in matematica per il concetto di *limite all'infinito*.

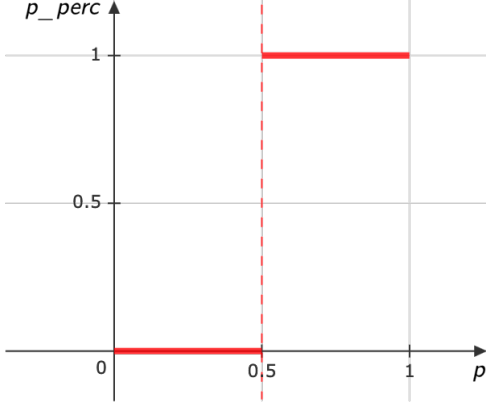


Figura 2: Grafico relativo alla soglia di percolazione ( $p_c = 0.5$ )

Nei casi reali, ovvero per reticoli di taglia finita, non è possibile stabilire un'affermazione così forte. Ciò che è possibile preservare dalla definizione è che esiste un **punto critico**  $p_c$  relativo alla probabilità  $p$ , oltre al quale è più probabile che avvenga percolazione e, al contrario, al di sotto del quale è meno probabile che questo si verifichi.

Lattice	$p_c$ (Site)	$p_c$ (Bond)
Cubic (body-centered)	0.246	0.1803
Cubic (face-centered)	0.198	0.119
Cubic (simple)	0.3116	0.2488
Diamond	0.43	0.388
Honeycomb	0.6962	0.65271*
4-Hypercubic	0.197	0.1601
5-Hypercubic	0.141	0.1182
6-Hypercubic	0.107	0.0942
7-Hypercubic	0.089	0.0787
Square	0.592746	0.50000*
Triangular	0.50000*	0.34729*

Tabella I: Punti critici ( $p_c$ ) per reticoli regolari.

In letteratura sono presenti diversi studi sulle caratteristiche di vari reticoli e i rispettivi valori di soglia. La tabella I mostra i punti critici per diversi reticoli regolari, ovvero composti da elementi ripetuti della stessa forma. La colonna **Lattice** indica la forma del reticolo, mentre le colonne **Site** e **Bond** distinguono i valori in percolazione di sito e di legame, rispettivamente [5]. Vi è una lieve, ma evidente, discrepanza tra i valori nelle due colonne. In generale, la rappresentazione tramite occupazione dei siti è considerata più generica rispetto alla sua controparte, questo perché la percolazione di legame può essere riformulata in termini di percolazione di sito, ma non si può affermare il contrario. I valori affiancati da un asterisco possono essere trovati tramite calcoli analitici, grazie ad alcune caratteristiche della forma del reticolo, sono quindi considerati *conosciuti*. È interessante notare che la tabella è composta per lo più da valori non conosciuti, cioè valori ottenuti da simulazioni al computer.

#### Cluster-finding

Nel modello che opera sui siti, la percolazione viene rilevata in seguito a un processo di *cluster-finding*, che consiste nel

partizionare il reticolo in diverse classi, tramite una relazione di equivalenza.

**Definizione 3.** Una relazione di equivalenza  $R$  su un insieme  $A$  è una relazione binaria che gode delle seguenti proprietà:

- $\forall x \in A : xRx$  (riflessività);
- $\forall x, y \in A : xRy \rightarrow yRx$  (simmetria);
- $\forall x, y, z \in A : xRy \wedge yRz \rightarrow xRz$  (transitività).

La relazione utilizzata è strettamente collegata al concetto di primi vicini. La definizione di primi vicini per un sito può variare a seconda della tipologia (forma e dimensioni) del reticolo utilizzato.

#### Distribuzione binomiale

Per formalizzare in modo completo il modello, è necessario introdurre la nozione di *variabile aleatoria*. Non verrà trattato l'argomento nel dettaglio, per lasciare più spazio alle implementazioni. In questo contesto, si introduce il concetto di variabile aleatoria come una variabile il cui valore dipende da un evento non deterministico. Questo evento è descritto attraverso funzioni di densità specifiche, che seguono leggi di *distribuzione* della probabilità [6]. Esistono diverse tipologie di distribuzioni, tra cui quella binomiale che, riassumendo, descrive un esperimento di prove ripetute.

Questo tipo di distribuzione è caratterizzato da una funzione di densità a 2 parametri

$$\mathcal{B}_{n,p}(k) = \binom{n}{k} p^k (1-p)^{n-k} \quad (1)$$

dove  $n$  rappresenta il numero di prove e  $p$  la probabilità di successo di ciascuna. La variabile  $k$  indica il risultato di cui vogliamo verificare la probabilità. L'equazione II rappresenta “la probabilità che si ottengano  $k$  successi ripetendo  $n$  volte una prova con probabilità di successo  $p$ ”.

### III. IMPLEMENTAZIONE

#### A. Prima soluzione

In questa relazione verranno discusse proprietà e risultati dell'algoritmo Hoshen-Kopelman [7]. Tuttavia, per dimostrarne la correttezza e provarne l'efficienza, verranno effettuati dei confronti con un altro algoritmo personalizzato, la cui correttezza è ben nota data la sua semplicità. L'algoritmo 1 mostra una prima soluzione al problema del partizionamento. È composto da 2 iterazioni principali: una “esterna” per tutti i siti colorati, e una “interna” per i siti adiacenti (primi vicini). Quest'ultima si appoggia ad un vettore che funge da *pila*, le cui dimensioni variano in maniera dinamica. Lo scopo della pila è quello di aggiungere, ad ogni iterazione interna, i siti occupati adiacenti al nodo corrente, per essere poi processati singolarmente. Ragionando sul funzionamento del codice, ci si convince piuttosto facilmente della ridondanza causata dalla pila. In altri termini, un sito occupato può essere processato più volte:

- una e una sola volta nell'iterazione esterna;
- una volta per ogni vicino occupato;
- altre volte per eventuali “catene di vicinanze”.

Questo dettaglio fa sì che la complessità asintotica temporale [8] dell'algoritmo appartenga alla classe  $O(n^4)$  nel caso di un reticolo bidimensionale  $n \times n$ .

## RIFERIMENTI BIBLIOGRAFICI

- [1] S. R. Broadbent and J. M. Hammersley, "Percolation processes: I. crystals and mazes," *Mathematical Proceedings of the Cambridge Philosophical Society*, vol. 53, no. 3, pp. 629-641, 1957.
- [2] C. Stover and E. W. Weisstein, "Bond percolation." [Online]. Available: <https://mathworld.wolfram.com/BondPercolation.html>
- [3] —, "Site percolation." [Online]. Available: <https://mathworld.wolfram.com/SitePercolation.html>
- [4] —, "Percolation threshold," from *Wolfram MathWorld*, 2002. [Online]. Available: <https://mathworld.wolfram.com/>
- [5] D. Stauffer and A. Aharony, *Introduction to percolation theory*. Taylor & Francis, 2018.
- [6] F. Edition, A. Papoulis, and S. U. Pillai, *Probability, random variables, and stochastic processes*. McGraw-Hill Europe: New York, NY, USA, 2002.
- [7] J. Hoshen and R. Kopelman, "Percolation and cluster distribution. i. cluster multiple labeling technique and critical concentration algorithm," *Phys. Rev. B*, vol. 14, pp. 3438-3445, Oct 1976. [Online]. Available: <https://link.aps.org/doi/10.1103/PhysRevB.14.3438>
- [8] M. Sipser, "Introduction to the theory of computation," *ACM Sigact News*, vol. 27, no. 1, pp. 27-29, 1996.

```

1 function res = CercaCluster3(L, p)
2 res.matrice=zeros(L+2);
3 aux=rand(L)<p;
4 res.matrice(2:end-1,2:end-1)=aux;
5 res.percolazioneTB=0;
6 res.percolazioneLR=0;
7 res.cluSz=[];
8 res.label=zeros(L+2);
9
10 labelC=1;
11 valid=find(res.matrice); % indici occupati
12
13 for iter=1:length(valid)
14 ii=valid(iter);
15 if(res.label(ii)==0)
16     % nuovo cluster, esploro i vicini
17     pila=ii;
18     res.label(ii)=labelC;
19     j=1;
20     while(j<=length(pila))
21         elemento=pila(j);
22         % aggiungo i vicini alla pila
23         if(res.matrice(elemento-1) &&
24             res.label(elemento-1)==0)
25             pila(end+1)=elemento-1;
26             res.label(elemento-1)=labelC;
27         end
28         if(res.matrice(elemento+1) &&
29             res.label(elemento+1)==0)
30             pila(end+1)=elemento+1;
31             res.label(elemento+1)=labelC;
32         end
33         if (res.matrice(elemento-L-2) &&
34             res.label(elemento-L-2)==0)
35             pila(end+1)=elemento-L-2;
36             res.label(elemento-L-2)=labelC;
37         end
38         if (res.matrice(elemento+L+2) &&
39             res.label(elemento+L+2)==0)
40             pila(end+1)=elemento+L+2;
41             res.label(elemento+L+2)=labelC;
42         end
43         j=j+1;
44     end
45     res.cluSz(end+1)=length(pila);
46     labelC=labelC+1;
47 end
48
49 res.label=res.label(2:end-1,2:end-1);
50 res.matrice=res.matrice(2:end-1,2:end-1);
51 left = clusters on the left edge
52 right = clusters on the right edge
53 if (~isempty(intersect(left,right)))
54     res.percolazioneLR=1;
55 end
56
57 auxT = unique(res.label(1:L:L*(L-1)+1));
58 top = auxT(auxT>0);
59 auxB = unique(res.label(L:L:L*L));
60 bottom = auxB(auxB>0);
61 if (~isempty(intersect(top, bottom)))
62     res.percolazioneTB = 1;
63 end
64 end

```

Codice 1: Algoritmo standard di cluster-finding