Percolazione nei reticoli quadrati

Studente: Alessio Russo **Matricola:** 376856
Corso di studio: Scienze Informatiche **Esame:** Modellazione e Simulazioni Numeriche

1 Algoritmo di Hoshen-Kopelman

L'algoritmo di Hoshen-Kopelman (HK76) è una tecnica di etichettatura multipla dei cluster. Il reticolo viene visitato sito per sito per colonne, partendo dallo spigolo in alto a sinistra per arrivare a quello in basso a destra. Si prenda, ad esempio, il reticolo in Figura 1

0	1	0	0	1	1	0	0	0	1	0	1	0	1	0
0	1	1	0	0	0	1	0	0	1	1	1	1	1	1
0	1	0	0	0	1	1	1	1	0	0	1	0	0	0
1	0	1	0	1	0	1	0	0	0	1	1	0	0	0
0	1	0	1	1	0	1	0	0	0	1	0	0	1	0
0	0	1	1	1	1	1	0	1	1	1	1	1	0	1
0	0	1	1	1	0	0	0	0	1	1	1	1	0	1
0	1	0	1	1	1	1	0	1	1	1	1	0	1	0
0	1	0	0	1	1	0	0	1	1	1	1	0	0	1
1	0	0	0	0	0	0	0	1	0	0	0	0	1	0
0	0	0	0	1	0	1	0	1	1	0	1	1	1	1
1	1	0	0	0	1	0	1	0	1	1	1	1	1	1
1	1	1	0	0	1	1	0	1	0	0	1	0	0	0
0	1	0	1	1	1	1	1	0	1	1	1	0	1	1
1	1	1	1	0	0	0	0	1	1	1	1	1	1	0

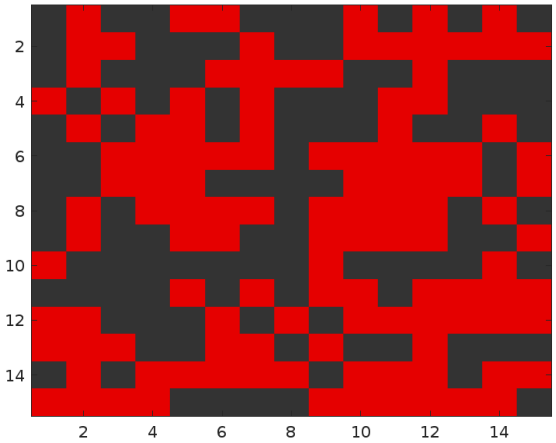

Figura 1: Esempio di reticolo quadrato 15×15 con siti colorati e non colorati

Durante la visita del reticolo, quando si incontra un sito colorato, allora: **(1)** Se il sito non è connesso ad altri siti colorato sopra o a sinistra, si inizia un nuovo cluster, a cui viene assegnata una `label` **(2)** Se c'è un primo vicino sopra o a sinistra colorato (uno solo dei due), il sito viene aggiunto al cluster del primo vicino colorato **(3)** Se i suoi primi vicini sono entrambi colorati, ma appartengono allo stesso cluster, il sito viene aggiunto al cluster dei primi vicini **(4)** Se i suoi primi vicini sono entrambi colorati, e non appartengono allo stesso cluster, il sito viene aggiunto al cluster con la `label` minore.

0	1	0	0	2	2	0	0	0	3	0	4	0	5	0
0	1	1	0	0	0	6	0	0	3	3	3	3	3	3
0	1	0	0	0	7	6	6	6	0	0	3	0	0	0
8	0	9	0	10	0	6	0	0	0	11	3	0	0	0
0	12	0	13	10	0	6	0	0	0	11	0	0	14	0
0	0	15	13	10	10	6	0	16	16	11	11	11	0	17
0	0	15	13	10	0	0	0	0	16	11	11	11	0	17
0	18	0	13	10	10	10	0	19	16	11	11	0	20	0
0	18	0	0	10	10	0	0	19	16	11	11	0	0	21
22	0	0	0	0	0	0	0	19	0	0	0	0	23	0
0	0	0	0	24	0	25	0	19	19	0	26	26	23	23
27	27	0	0	0	28	0	29	0	19	19	19	19	19	19
27	27	27	0	0	28	28	0	30	0	0	19	0	0	0
0	27	0	31	31	28	28	28	0	32	32	19	0	33	33
34	27	27	27	0	0	0	0	35	32	32	19	19	19	0

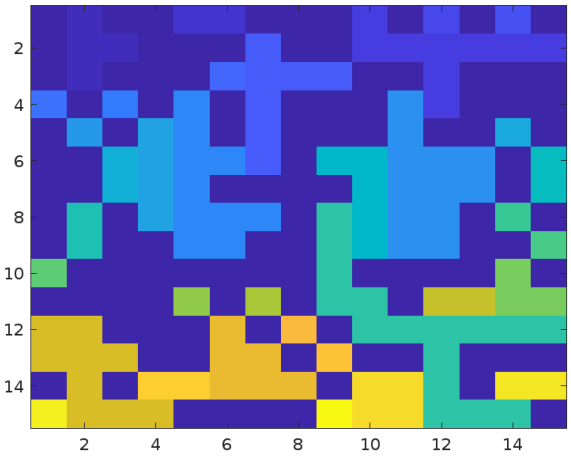

Figura 2: Esempio di etichettatura dei cluster con l'algoritmo di Hoshen-Kopelman

Ad esempio, il cluster associati al reticolo in Figura 1 sono mostrati in Figura 2.

Tuttavia, quando si incontra un caso come quello descritto nel punto **(4)**, occorre memorizzare che i due cluster sono in realtà lo stesso cluster. Questo viene fatto usando un vettore chiamato **Label of Label** (`LofL`), che contiene tutta l'informazione necessaria sui label dei cluster. In particolare, il modulo `HKclass`: per un *good label*, memorizza la taglia del cluster; per *bad label*, memorizza qual è il vero cluster label a cui questo label appartiene. Questa distinzione viene fatta attraverso i segni dei numeri interi contenuti in LofL. Di seguito è riportato il LofL corrispondete al reticolo preso in esame

ID	1	2	3	4	5	6	7	8	9	10	11	12
Val	4	2	56	-3	-3	24	-6	1	1	-6	-3	1

ID	13	14	15	16	17	18	19	20	21	22	23	24
Val	-10	1	-10	-3	2	2	-3	1	1	1	-3	1

ID	25	26	27	28	29	30	31	32	33	34	35
Val	1	-3	18	-27	1	1	-27	-3	-3	-27	-3

Tuttavia, l'algoritmo HK restituisce in modo corretto le taglie dei cluster, ma non garantisce che tutti i siti di un fissato cluster abbiano lo stesso valore. Per questo motivo, effettuiamo una rietichettatura successiva. La Figura 3 ne mostra un esempio.

0	1	0	0	2	2	0	0	0	3	0	3	0	3	0
0	1	1	0	0	0	6	0	0	3	3	3	3	3	3
0	1	0	0	0	6	6	6	6	0	0	3	0	0	0
8	0	9	0	6	0	6	0	0	0	3	3	0	0	0
0	12	0	6	6	0	6	0	0	0	3	0	0	14	0
0	0	6	6	6	6	6	0	3	3	3	3	3	0	17
0	0	6	6	6	0	0	0	0	3	3	3	3	0	17
0	18	0	6	6	6	6	0	3	3	3	3	0	20	0
0	18	0	0	6	6	0	0	3	3	3	3	0	0	21
22	0	0	0	0	0	0	0	3	0	0	0	0	3	0
0	0	0	0	24	0	25	0	3	3	0	3	3	3	3
27	27	0	0	0	27	0	29	0	3	3	3	3	3	3
27	27	27	0	0	27	27	0	30	0	0	3	0	0	0
0	27	0	27	27	27	27	27	0	3	3	3	0	3	3
27	27	27	27	0	0	0	0	3	3	3	3	3	3	0

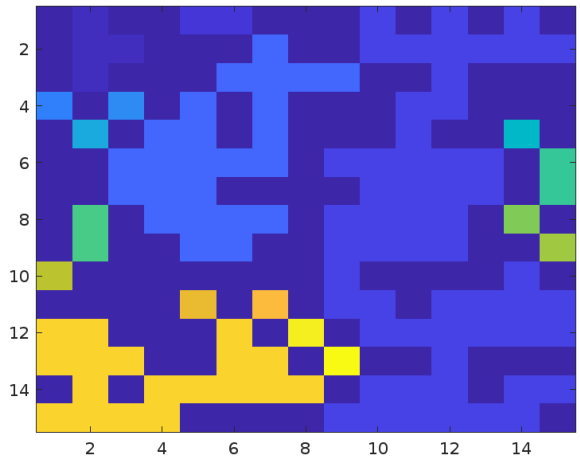

Figura 3: Esempio di rietichettatura dei cluster con etichette uniformi per ciascun gruppo

A questo punto, l'obiettivo è determinare se esistono cluster percolanti all'interno del reticolo, ossia se esiste almeno un'etichetta condivisa tra la prima e l'ultima riga (percolazione verticale) e tra la prima e l'ultima colonna (percolazione orizzontale). Per fare ciò, possiamo sviluppare un algoritmo che, basandosi sull'estrazione delle etichette **uniche** presenti lungo i bordi della matrice, e mediante l'utilizzo dell'operazione `intersect`, verifica l'esistenza di almeno una etichetta comune tra i bordi opposti. Se tale etichetta è presente, viene restituito `true` per il tipo di percolazione considerato, altrimenti `false`.

Si noti che, poiché ci interessa esclusivamente determinare il cluster di appartenenza della prima e dell'ultima riga e colonna per valutare la percolazione, è sufficiente rietichettare solo questi elementi, ignorando il centro del reticolo e risparmiando così tempo di calcolo. La Figura 4 ne mostra un esempio.

Concludiamo dicendo che l'analisi della correttezza dell'implementazione proposta è stata effettuata non soltanto tramite test individuali sul singolo algoritmo, ma anche attraverso un confronto diretto

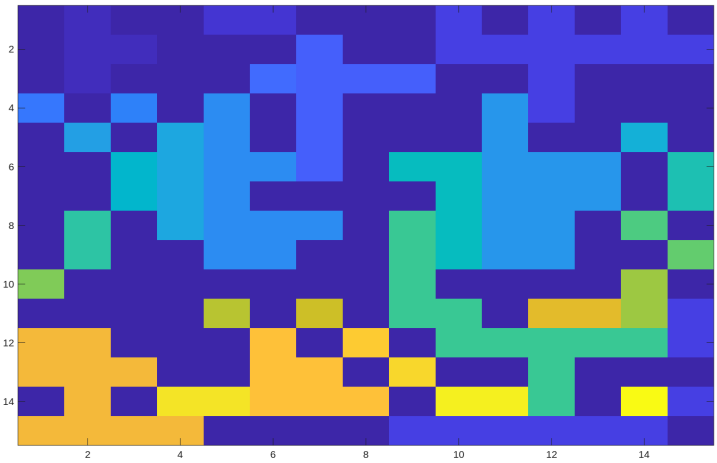

Figura 4: Esempio di rietichettatura dei soli bordi per il test di percolazione

con l'algoritmo naive presentato a lezione. Nello specifico, è stato generato un reticolo quadrato di taglia 100, con una probabilità di colorazione dei siti pari al 60%. Tale reticolo è stato analizzato prima con l'algoritmo naive e successivamente con l'algoritmo HK76, confrontando i risultati ottenuti per la percolazione verticale (top-bottom) e orizzontale (left-right). Questo confronto è stato ripetuto 10.000 volte, e in tutti i casi i risultati forniti dai due algoritmi sono stati coincidenti.

2 Confronto tra algoritmi naive e HK76

L'algoritmo di Hoshen-Kopelman (HK76, indicato in *blu*) ha mostrato una significativa efficienza computazionale superiore rispetto all'algoritmo di etichettatura naive (indicato in *arancione*), precedentemente implementato. Questa valutazione è stata condotta attraverso due diverse analisi: la prima ha considerato il tempo di calcolo mantenendo costante la probabilità di colorazione dei siti e variando la dimensione del reticolo, mentre la seconda ha studiato il comportamento opposto, ovvero il tempo di calcolo mantenendo costante la dimensione del reticolo e variando la probabilità di colorazione dei siti.

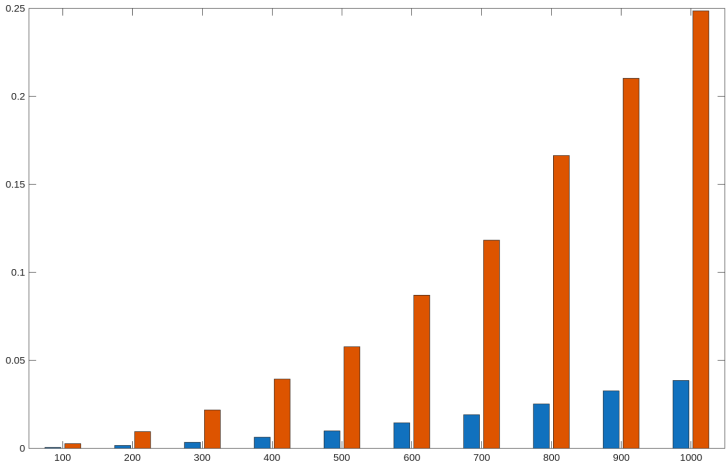

Figura 5: Tempo di esecuzione in funzione della dimensione del reticolo per gli algoritmi HK76 e naive.

2.1 Confronto con probabilità costante e taglia variabile

Nel primo caso, è stata fissata una probabilità di colorazione dei siti pari al 60%, variando la dimensione del reticolo quadrato nell'intervallo compreso tra 100 e 1000, con incrementi di 100, per un totale di 10 configurazioni. La Figura 5 mostra i tempi medi di esecuzione dell'algoritmo, calcolati eseguendo ciascuna configurazione 50 volte. Si noti come, all'aumentare della dimensione del reticolo, l'algoritmo HK76 offra prestazioni migliori rispetto all'algoritmo naive in termini di tempo di esecuzione. Di seguito sono riportati i tempi medi di esecuzione:

hk76	0.0005	0.0018	0.0037	0.0065	0.0099	0.0140	0.0192	0.0255	0.0313	0.0385
naive	0.0023	0.0090	0.0199	0.0354	0.0552	0.0806	0.1097	0.1482	0.1877	0.2383

e i relativi errori (da moltiplicare per 10^{-3}):

hk76	0.0107	0.0511	0.0261	0.0830	0.1095	0.1083	0.1271	0.1536	0.1598	0.1649
naive	0.0108	0.0224	0.0177	0.1090	0.1216	0.2112	0.2641	0.2072	0.3270	0.4488

2.2 Confronto con taglia costante e probabilità variabile

Nel secondo caso, invece, la dimensione del reticolo è stata mantenuta fissa a 500, mentre la probabilità di colorazione dei siti è stata variata da 0,1 a 1, con incrementi di 0,1, per un totale di 10 esperimenti.

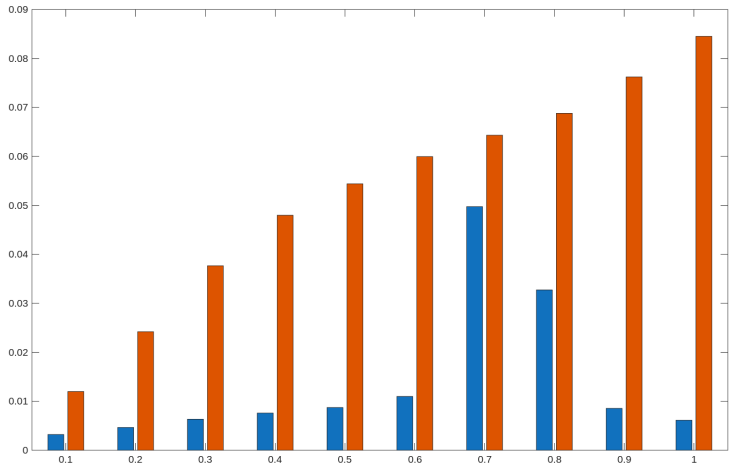

Figura 6: Tempo di esecuzione in funzione della probabilità di colorazione per gli algoritmi HK76 e naive.

La Figura 6, che mostra i tempi medi di esecuzione su 50 iterazioni, evidenzia come l'algoritmo HK76 risulti nettamente migliore rispetto alla versione naive. Di seguito sono riportati i tempi medi di esecuzione:

hk76	0.0032	0.0047	0.0063	0.0076	0.0087	0.0110	0.0497	0.0327	0.0086	0.0061
naive	0.0120	0.0242	0.0377	0.0480	0.0544	0.0600	0.0643	0.0688	0.0762	0.0845

e i relativi errori (da moltiplicare per 10^{-3}):

hk76	0.0776	0.0643	0.0935	0.0997	0.0864	0.1096	0.6884	0.4468	0.0981	0.0650
naive	0.1488	0.2100	0.4623	0.4539	0.3350	0.4271	0.2585	0.2870	0.2396	0.3537

3 Ricerca della soglia di percolazione

Per analizzare il comportamento della probabilità di percolazione P_{perc} in funzione della probabilità di occupazione dei siti p_{col}, è stata condotta una serie di simulazioni su reticoli quadrati di diverse dimensioni ($L = 100$, 300 e 1000). L'intervallo di p_{col} considerato va da 0.55 a 0.65, con incrementi di 0.01, in modo da esplorare con buona risoluzione la zona critica.

Per ogni coppia di valori (L, p_{col}), l'esperimento è stato ripetuto 50 volte, così da permettere una stima della probabilità di percolazione e del relativo errore. Ogni simulazione consiste nella generazione di un reticolo in cui ogni sito viene occupato con probabilità p_{col}. Una volta creato il reticolo, i cluster connessi di siti occupati sono stati individuati tramite l'algoritmo hk76. Successivamente, è stato verificato se esiste almeno un cluster che attraversa il reticolo da sinistra a destra (percolazione orizzontale) o dall'alto in basso (percolazione verticale).

Un confronto tra i risultati ottenuti nelle due direzioni ha permesso di confermare che, come previsto, tenendo conto dei relativi errori, la probabilità di percolazione orizzontale è la stessa di quella verticale.

Probabilità media di percolazione LR

	0.55	**0.56**	**0.57**	**0.58**	**0.59**	**0.60**	**0.61**	**0.62**	**0.63**	**0.64**	**0.65**
100	0.02	0.06	0.06	0.18	0.38	0.60	0.90	0.94	1.00	1.00	1.00
300	0.00	0.00	0.00	0.00	0.30	0.82	0.98	1.00	1.00	1.00	1.00
1000	0.00	0.00	0.00	0.00	0.16	0.98	1.00	1.00	1.00	1.00	1.00

Probabilità media di percolazione TB

	0.55	**0.56**	**0.57**	**0.58**	**0.59**	**0.60**	**0.61**	**0.62**	**0.63**	**0.64**	**0.65**
100	0.00	0.06	0.10	0.18	0.48	0.70	0.82	0.90	1.00	1.00	1.00
300	0.00	0.00	0.00	0.02	0.36	0.86	0.98	1.00	1.00	1.00	1.00
1000	0.00	0.00	0.00	0.00	0.18	1.00	1.00	1.00	1.00	1.00	1.00

Errore nel calcolo di LR

	0.55	**0.56**	**0.57**	**0.58**	**0.59**	**0.60**	**0.61**	**0.62**	**0.63**	**0.64**	**0.65**
100	0.02	0.03	0.03	0.05	0.07	0.07	0.04	0.03	0.00	0.00	0.00
300	0.00	0.00	0.00	0.00	0.07	0.05	0.02	0.00	0.00	0.00	0.00
1000	0.00	0.00	0.00	0.00	0.05	0.02	0.00	0.00	0.00	0.00	0.00

Errore nel calcolo di TB

	0.55	**0.56**	**0.57**	**0.58**	**0.59**	**0.60**	**0.61**	**0.62**	**0.63**	**0.64**	**0.65**
100	0.00	0.03	0.04	0.05	0.07	0.07	0.05	0.04	0.00	0.00	0.00
300	0.00	0.00	0.00	0.02	0.07	0.05	0.02	0.00	0.00	0.00	0.00
1000	0.00	0.00	0.00	0.00	0.05	0.00	0.00	0.00	0.00	0.00	0.00

Per ogni dimensione del reticolo L è stata stimata la soglia di percolazione p_c come il valore della probabilità di colorazione p_{col} per cui la probabilità di percolazione P_{perc} raggiunge il valore 0.5.

Poiché i dati ottenuti tramite simulazione possono contenere duplicati (ad esempio valori estremi come 0.00 e 1.00), sono stati innanzitutto rimossi i punti ripetuti nei valori di P_{perc}.

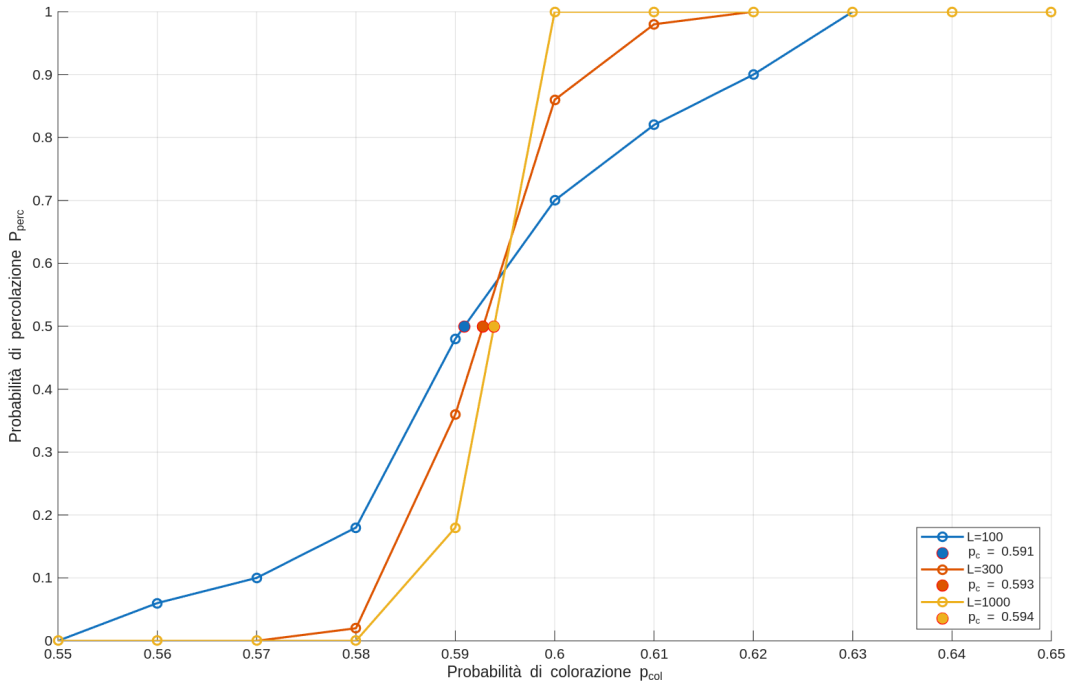

Figura 7: Andamento della probabilità di percolazione al variare di p_{col}.

Successivamente, è stata utilizzata un'interpolazione lineare tra i punti ottenuti per stimare il valore di p_{col} tale che $P_{perc} = 0.5$. Per ciascuna taglia del reticolo, il valore stimato di p_c è stato evidenziato nel grafico in Figura 7 come punto pieno sovrapposto alla curva $P_{perc}(p_{col})$. La procedura è stata ripetuta per tutte le dimensioni considerate, in modo da ottenere una stima della soglia di percolazione in funzione di L.

Questo approccio consente anche di osservare l'avvicinamento di $p_c(L)$ al valore critico nel limite termodinamico. In particolare, sulla base dei risultati ottenuti, questo valore asintotico è stimato attorno a $p_c \sim 0.59$.

3.1 Calcolo di P_1

La quantità P_1 viene calcolata come rapporto tra la taglia del cluster più grande (s_{max}) e il numero totale di siti presenti nel reticolo (L^2):

$$P_1 = \frac{s_{max}}{L^2} \tag{1}$$

Dal punto di vista probabilistico, P_1 rappresenta la frazione dell'intero reticolo occupata dal cluster dominante. Quando p_{col} è molto basso, i cluster tendono ad essere piccoli e isolati, quindi P_1 assume valori trascurabili. Ma man mano che p_{col} si avvicina alla soglia critica, P_1 cresce rapidamente: questo riflette la formazione di cluster massicci, capaci di connettere porzioni opposte del reticolo. Di seguito sono riportati i valori ottenuti in fase di simulazione

	0.55	**0.56**	**0.57**	**0.58**	**0.59**	**0.60**	**0.61**	**0.62**	**0.63**	**0.64**	**0.65**
100	0.07	0.10	0.13	0.16	0.26	0.33	0.41	0.48	0.53	0.57	0.60
300	0.02	0.03	0.05	0.08	0.19	0.37	0.46	0.54	0.57	0.59	0.61
1000	0.00	0.00	0.01	0.03	0.12	0.42	0.51	0.55	0.58	0.60	0.62

e i relativi errori

3.2 Calcolo di P_2

La seconda quantità, P_2, mantiene lo stesso numeratore (s_{max}) ma usa come denominatore il valore atteso dei siti occupati ($p_{col} \cdot L^2$):

$$P_2 = \frac{s_{max}}{p_{col} \cdot L^2} \tag{2}$$

Questa formulazione permette di interpretare P_2 come la frazione dei siti "potenzialmente occupati" che appartiene al cluster più grande. In pratica, si osserva quanto il cluster percolante è significativo rispetto al materiale disponibile. Un valore elevato di P_2 indica che il cluster dominante raccoglie una parte consistente dei siti teoricamente colorati. Vicino alla soglia, anche P_2 mostra una transizione netta, che può essere usata come indicatore della percolazione. Di seguito sono riportati i valori ottenuti in fase di simulazione

	0.55	**0.56**	**0.57**	**0.58**	**0.59**	**0.60**	**0.61**	**0.62**	**0.63**	**0.64**	**0.65**
100	0.12	0.18	0.23	0.28	0.44	0.55	0.68	0.77	0.85	0.89	0.92
300	0.03	0.05	0.08	0.14	0.33	0.61	0.75	0.86	0.90	0.93	0.94
1000	0.01	0.01	0.02	0.05	0.20	0.71	0.83	0.88	0.91	0.93	0.95

e i relativi errori

3.3 Calcolo di P_3

Nel calcolo di P_3, il numeratore è sempre s_{max}, ma il denominatore cambia nuovamente: si utilizza il numero effettivo di siti colorati nel reticolo, cioè la somma delle taglie di tutti i cluster presenti:

$$P_3 = \frac{s_{max}}{\sum_s s \cdot n_s} \tag{3}$$

Questa quantità riflette la frazione di siti occupati che appartengono al solo cluster dominante. È quindi una misura "interna" al sistema: tra tutti i siti effettivamente colorati, quanti sono coinvolti nel cluster principale? A valori bassi di p_{col}, il sistema è disperso e P_3 è piccolo. Attorno alla soglia, il valore cresce bruscamente, segnalando che il cluster dominante comincia ad assorbire gran parte dei siti attivi. Sopra la soglia, P_3 tende rapidamente a 1.

Questa crescita è indicativa del passaggio da una fase dominata da piccoli cluster a una dominata da uno solo, molto esteso, coerentemente con la definizione di percolazione su reticoli infiniti.

3.4 Calcolo di $RACS$

La misura $RACS$ (Reduced Average Cluster Size) si basa su una media pesata delle taglie dei cluster, **escludendo** esplicitamente il cluster più grande. È definita come:

$$RACS = \frac{\sum\limits_{s<s_{max}} s^2 \cdot n_s}{\sum\limits_{s<s_{max}} s \cdot n_s} \tag{4}$$

Qui si ha una vera e propria media della taglia dei cluster, con pesi proporzionali alla taglia stessa. L'esclusione del cluster più grande è fondamentale: vicino alla soglia di percolazione, questo cluster raccoglie la gran parte dei siti occupati, rendendo il sistema fortemente sbilanciato. Rimuovendolo, è possibile osservare cosa accade nella "popolazione" restante di cluster minori.

Dal punto di vista interpretativo, $RACS$ consente di evidenziare il comportamento critico: crescendo con p_{col}, $RACS$ raggiunge un massimo in prossimità della soglia, per poi diminuire quando il

sistema entra in fase percolante e il cluster dominante diventa preponderante. Questo massimo è un indicatore utile per identificare la transizione.

In un reticolo di taglia infinita, dove non si può più parlare di percolazione da un bordo all'altro, $RACS$ (insieme a P_3) offre un'alternativa concettuale per descrivere il fenomeno: la transizione non è più geometrica ma statistica, legata all'emergere di un cluster infinito che assorbe la maggior parte della massa disponibile.