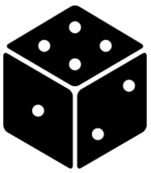
METODI PROBABILISTICI PER L'INFORMATICA

Prof. Massimiliano Goldwurm
6 CFU

Luca Cappelletti
Desilda Toska

Lecture Notes
Year 2017/2018



Magistrale Informatica
Università di Milano
Italy
21 giugno 2018

Indice

1 Probabilità

1.1 Disuguaglianza di Chernoff

$$\mathbb{P}\left(\left|X_{n,p} - np\right| \geqslant n\epsilon\right) < 2e^{-2e^2 n}$$

Figura 1.1: Disuguaglianza di Chernoff per le binomiali

La **disuguaglianza di Chernoff** utilizza quando è necessario calcolare la probabilità di una **funzione** f che:

1. Non è semplice da calcolare, oppure,
2. Non è nota.

1.1.1 Caratteristiche

Variabili di indipendenti

Essa richiede che le **variabili siano indipendenti**, *condizione che la disuguaglianza di Markov non richiede* e nel caso delle disuguaglianza di Chebyshev è necessaria solo l'indipendenza da coppie di variabili casuali.

Non è una disuguaglianza vera

La disuguaglianza di Chernoff **non è una disuguaglianza vera**, ma piuttosto va vista come una tecnica per ottenere limiti esponenziali decrescenti sulle probabilità di coda.

Il valore di n è arbitrario

La disuguaglianza di Chernoff non fornisce un preciso valore di n oltre il quale siamo sicuri che la probabilità della coda sia minore di una δ fissata.

1.1.2 Applicazioni alle binomiali

Nell'analisi degli algoritmi probabilistici occorre spesso valutare la coda di una binomiale, ovvero la probabilità che questa variabili aleatoria disti dalla media per una quantità fissata.

Partendo dalla **disuguaglianza di Chebyshev** per una variabile X con $\mathrm{Var}(X)$ finita:

$$\mathbb{P}\left(\left|X - \mathbb{E}(X)\right| \geqslant a\right) \leqslant \frac{\mathrm{Var}(X)}{a^2} \quad \forall a > 0$$

Figura 1.2: Disuguaglianza di Chebyshev

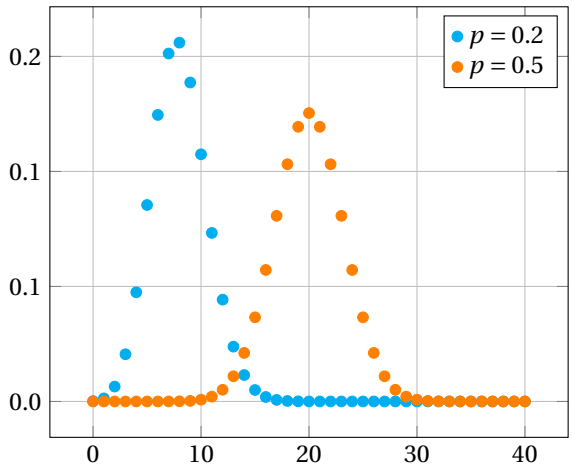


Figura 1.3: Distribuzioni binomiali al variare del parametro p

Nel caso di $X_{n,p}$ binomiale, con media $\mathbb{E}(X) = np$ e varianza $\mathrm{Var}(X) = npq$, $\forall \epsilon > 0$ vale che:

$$\mathbb{P}\left(\left|X_{n,p} - np\right| \geqslant n\epsilon\right) \leqslant \frac{pq}{\epsilon} \cdot \frac{1}{n} = \mathcal{O}\left(\frac{1}{n}\right)$$

Per il **teorema del limite centrale** vale che:

$$\frac{X_{n,p} - np}{\sqrt{npq}} \longrightarrow \mathcal{N}(0,1)$$

Applico il limite ed ottengo:

$$\lim_{n\to+\infty} \mathbb{P}\left(\frac{X_{n,p} - np}{\sqrt{npq}} \geqslant \epsilon\right) = \frac{2}{\sqrt{2\pi}} \int_{\epsilon}^{+\infty} e^{-\frac{t^2}{2}}\, dt$$

Vado ad aggiungere il termine $\sqrt{\frac{n}{pq}}$ come coefficiente a ϵ per consentire l'integrazione:

$$\lim_{n\to+\infty} \mathbb{P}\left(\frac{X_{n,p} - np}{\sqrt{npq}} \geqslant \sqrt{\frac{n}{pq}} \cdot \epsilon\right) = \frac{2}{\sqrt{2\pi}} \int_{\sqrt{\frac{n}{pq}}\cdot\epsilon}^{+\infty} e^{-\frac{t^2}{2}}\, dt$$

Maggioro l'integrale:

$$\frac{2}{\sqrt{2\pi}} \int_{\sqrt{\frac{n}{pq}}\cdot\epsilon}^{+\infty} e^{-\frac{t^2}{2}}\, dt \leqslant \frac{2}{\sqrt{2\pi}} \int_{\sqrt{\frac{n}{pq}}\cdot\epsilon}^{+\infty} \frac{t}{\sqrt{\frac{n}{pq}} \cdot \epsilon} e^{-\frac{t^2}{2}}\, dt = \mathcal{O}\left(\frac{1}{\sqrt{n}} e^{-\frac{\epsilon^2}{2pq} n}\right)$$

Il risultato ottenuto, si avvicina allo 0 molto più velocemente del valore $\mathcal{O}\left(\frac{1}{n}\right)$ ottenuto precedentemente tramite la disuguaglianza di Chebyshev.

La disuguaglianza di Chebyshev è più generale, di conseguenza è più debole. Per esempio nel caso della distribuzione gaussiana risulta molto debole, per cui se è necessario avere un'accuratezza maggiore conviene assolutamente usare la disuguaglianza di Chernoff.

1.1.3 Esempio: Riduzione della probabilità di errore di un algoritmo

Supponiamo di voler calcolare una funzione $f_I \to O$ difficile da calcolare e di disporre di un algoritmo probabilistico $\mathscr{A}$ tali che $\forall x \in I, T_{\mathscr{A}} \leqslant p(|x|)$ dove p è un polinomio di grado **piccolo** e che $\mathbb{P}\left(A(x) = f(x)\right) \geqslant \frac{3}{4}$, col valore di destra maggiore di $\frac{1}{2}$ e indipendente da x, altrimenti non potrei determinare il risultato corretto la frequenza con cui appare

Dato un intero $t > 0$, chiamiamo $\mathscr{A}_t$ l'algoritmo probabilistico che ripete $\mathscr{A}$ per un numero t di volte e ne restituisce il valore più frequente, se esso esiste.

L'algoritmo non garantisce che il risultato sia corretto, infatti anche con n iterazioni rimane possibile che l'algoritmo calcoli con frequenza maggiore la soluzione errata.

Algorithm 1: Riduzione della probabilità di errore

input : $x \in I$
output: Valore più frequente o non so

```
1  begin
2      for i ∈ [1,...,t] do
3          Esecuzioni indipendenti di 𝒜 su x;
4          A[i] = 𝒜(x)
5      end
6      if ∃z ∈ (A[1],...,A[t]) : #{j ∈ {1,...,t} : z = A[j]} > t/2 then
7          return z;
8      end
9      else
10         return non so rispondere;
11     end
12 end
```

Quante volte devo iterare l'algoritmo per ridurre la probabilità di errore ad un valore δ?

Utilizzando la disuguaglianza di Chernoff per le binomiali ottengo:

$$\mathbb{P}\left(\mathscr{A}_t \neq f(x)\right) \leqslant \mathbb{P}\left(X_{t,\frac{3}{4}} \leqslant \frac{t}{2}\right) = \mathbb{P}\left(X_{t,\frac{3}{4}} - \frac{3}{4}t \leqslant -\frac{t}{4}\right) \leqslant e^{-2\frac{1}{16}t} = e^{-\frac{1}{8}} \leqslant \delta$$

Calcolo il logaritmo naturale ed ottengo il valore di t per $\delta = 1000^{-1}$.

$$t \geqslant 8\ln\frac{1}{\delta} \Rightarrow t \geqslant 8\ln 1000 = 24\ln 10 \approx 56$$

Se avessimo invece utilizzato la **disequazione di Chebyshev** avremmo invece ottenuto un risultato molto peggiore:

$$\mathbb{P}\left(X_{t,\frac{3}{4}} - \frac{3}{4}t \leqslant -\frac{t}{4}\right) \leqslant \mathbb{P}\left(\left|X_{t,\frac{3}{4}} - \frac{3}{4}t\right| \geqslant \frac{t}{4}\right) \leqslant \frac{3}{t} \Rightarrow t \geqslant 3000$$

1.1.4 Esempio: Lancio di monete

Consideriamo il caso di una variabile aleatoria $X_{n,\frac{1}{2}}$ con media $\mathbb{E}(X) = \frac{n}{2}$:

$$\mathbb{P}\left(\left|X_{n,\frac{1}{2}} - \frac{n}{2}\right| \geqslant \sqrt{n\log n}\right) \leqslant 2e^{-2\log n} = \frac{2}{n^2}$$

Per esempio, posto $n = 100$ si ottiene:

$$\mathbb{P}\left(\left|X_{n,\frac{1}{2}} - 50\right| \geqslant 21\right) \leqslant \frac{1}{5000}$$

Rendendo la maggiorazione più aderente, dividendo il termine destro per due, si ottiene:

$$\mathbb{P}\left(\left|X_{n,\frac{1}{2}} - \frac{n}{2}\right| \geqslant \sqrt{\frac{n\log n}{4}}\right) \leqslant 2e^{\frac{-2\log n}{4}} = \frac{2}{\sqrt{n}}$$

Nuovamente risolvendo per $n = 100$ si ottiene:

$$\mathbb{P}\left(\left|X_{n,\frac{1}{2}} - 50\right| \geqslant 10.5\right) \leqslant \frac{1}{5} \rightarrow \mathbb{P}\left(39 \leqslant X_{n,\frac{1}{2}} \leqslant 61\right) \geqslant \frac{4}{5}$$

1.1.5 Esempio: Intervalli di confidenza

Si vuole valutare la probabilità p di un evento A disponendo di un test f sull'evento A che restituisce un valore:

$$f: \begin{cases} \text{SI} & \text{con probabilità } p \\ \text{NO} & \text{con probabilità } 1-p \end{cases}$$

$\frac{X_{n,p}}{n} = \overline{p}$ è una variabile aleatoria con media $\mathbb{E}\left(X_{n,p}\right) = np$.

$$\mathbb{P}\left(\left|p - \overline{p}\right| \geqslant \delta\right) = \mathbb{P}\left(\left|np - X_{n,p}\right| \geqslant n\delta\right) \leqslant 2e^{-2\delta^2 n} \leqslant t$$

$$-2\delta^2 n \leqslant \log\frac{2}{t} \Rightarrow n \leqslant \frac{1}{2\delta^2}\log\frac{2}{t}$$

Dove t rappresenta la probabilità di errore.

Ponendo $\delta = 0.1$ e $t = 100^{-1}$ si ottiene:

$$n \leqslant 50\log 200 = 50\left(\log 2 + 2\log 10\right) \approx 250 \Rightarrow n \geqslant 250$$

2

Algoritmi probabilistici

Gli algoritmi probabilistici richiedono un numero minore di calcoli degli algoritmi deterministici e quindi determinano una soluzione in un tempo minore. Esistono principalmente due famiglie di algoritmi probabilistici:

1. Per cercare una soluzione esatta.
2. Per cercare una soluzione approssimata.

Per **algoritmi Monte-Carlo** si intendono gli algoritmi della categoria **1-sided error** e **2-sided error**.

Generalmente, per diminuire la probabilità di errore viene reiterato l'algoritmo probabilistico per un numero di volte ottenuto utilizzando la **disequazione di Chernoff** sino a che l'errore è inferiore ad un δ accettabile.

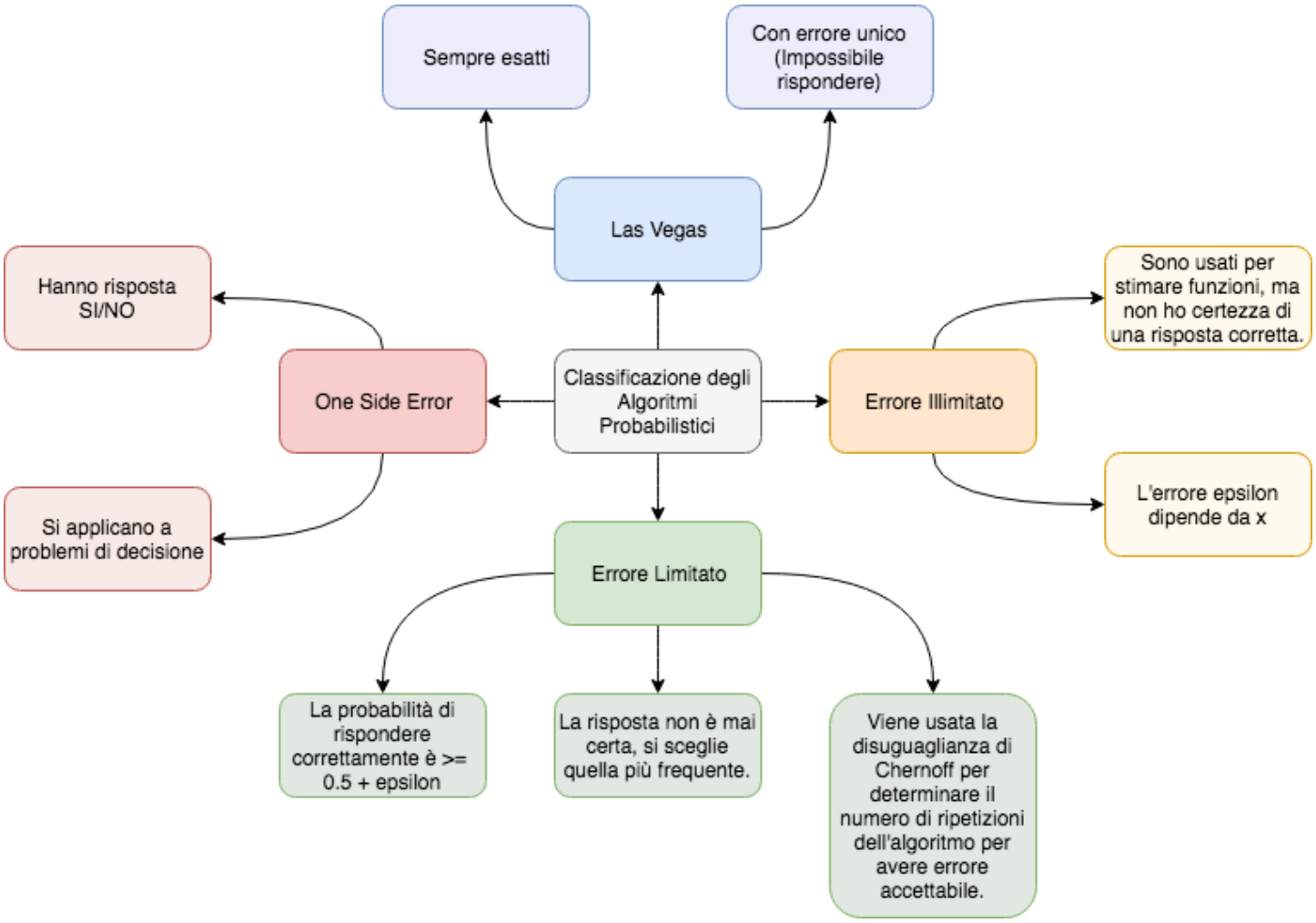


Figura 2.1: Mappa degli algoritmi probabilistici

2.1 1-Sided Error

Hanno risposta SI/NO e vengono applicati a problemi di decisione.

2.1.1 Esempio: Problema di protocollo di comunicazione

Si vuole determinare se due stringhe binarie a e b, lunghe n, salvate su due terminali separati sono uguali senza inviare una stringa completamente da un terminale all'altro.

Un approccio banale risolverebbe in n messaggi il problema.

Un approccio sofisticato utilizza i resti della divisione con numeri primi. Dato un numero primo $p \in \{2,\dots,n^2\}$ con probabilità uniforme, di dimensione $\lceil 2\log_2 n \rceil$ bit.

$$r_a = \text{resto}\left(\frac{a}{p}\right) \in \{0,1,\dots,p-1\} \qquad r_b = \text{resto}\left(\frac{b}{p}\right) \in \{0,1,\dots,p-1\}$$

$$r_a \neq r_b \Rightarrow a \neq b$$

$$r_a = r_b \Rightarrow a \text{ potrebbe essere uguale a } b$$

I valori del resto e di p sono inviati al secondo dei due nodi usando al più $2\lceil 2\log_2 n \rceil$ bit, dove viene ripetuta la divisione ed effettuato il controllo, rispondendo con un bit se diversi o zero se uguali.

Se $\mathscr{A}$ restituisce "diverso" la risposta è certamente corretta, mentre se restituisce "uguale" potrebbe essere sbagliato, con una probabilità pari a:

$$\mathbb{P}\left(\text{errore} \mid \mathscr{A} \text{ dice uguali}\right) = \mathbb{P}\left(p \text{ divide } a-b \;\middle|\; a \neq b\right)$$

Usando il **teorema dei numeri primi** che enuncia:

Teorema 2.1.1 (Teorema dei numeri primi). Se $\text{prim}(m) = \#\{i \in \{2,\dots,m\} : i \text{ è primo}\}$ allora vale che:

$$\text{prim}(m) \approx \frac{m}{\ln m} \qquad \text{ovvero} \qquad \frac{\text{prim}(m)}{m} \ln m \longrightarrow 1$$

Inoltre è noto che:

$$\forall m > 67, \text{prim}(m) > \frac{m}{\ln m}$$

Ogni numero intero di k bit possiede al più $k-1$ divisori primi.

Si determina che la probabilità di scegliere un certo p risulta essere pari a:

$$\mathbb{P}(p) = \frac{1}{\text{prim}\left(n^2\right)}, \qquad \mathbb{P}(\text{errore}) = \frac{\#\{i \in \{1,\dots,n^2\}, i \text{ primo e divisore di } a-b\}}{\text{prim}(n)^2}$$

$$\mathbb{P}(\text{errore}) \leqslant \frac{n-1}{{}^{n^2}/_{2\log_2 n}} \leqslant \frac{2\log_2 n}{n} \longrightarrow 0$$

$$\mathbb{P}(\text{errore}) \leqslant 10^{-14}$$

Anche per n piccolo, se $\frac{2\ln n}{n} < \frac{1}{2}$ ripeto l'algoritmo t volte ed ottengo una probabilità di errore pari a $\mathbb{P}(\text{errore}) < \frac{1}{2}^t$

2.1.2 Esempio: Quicksort

Dato $\mathbb{E}(n)$, numero medio di confronti richiesto da **Quicksort** su un input di n elementi determinato come:

$$\mathbb{E}(n) = (n+1) + \frac{1}{n}\sum_{k=0}^{n-1}\left[\mathbb{E}(k) + \mathbb{E}(n-1-k)\right]$$

Usando il corollario del **teorema delle probabilità totali** che ricordiamo essere:

Definizione 2.1.2 (Formula delle probabilità totali). Sia $(\Omega, \mathscr{F}, P)$ uno spazio di probabilità e $F_1, F_2, \ldots, F_n \in \mathscr{F}$ una partizione finita di ω, $\bigcup_{k=1}^{n} F_k = \Omega$ e $F_h \cap F_k = \emptyset$ se $h \neq k$, tale che $\mathbb{P}(F_k) > 0$ per $k = 1, 2, \ldots, n$. Allora ogni evento $E \in \mathscr{F}$ si ha:

$$\mathbb{P}(E) = \sum_{k=1}^{n} \mathbb{P}(E \mid F_k)\mathbb{P}(F_k)$$

Corollario 2.1.2.1 (Corollario sulla media).

$$\mathbb{E}(E) = \sum_{k=1}^{n} \mathbb{E}(E \mid F_k)\mathbb{P}(F_k)$$

Ottengo che i **tempo di calcolo che Quicksort** impiegati con un input di dimensione n sono pari a:

$$\mathbb{E}(T_n) = \sum_{i=0}^{n-1} \mathbb{E}(T_n \mid B_i)\mathbb{P}(B_i) = n - 1 + \sum_{i=0}^{n-1} \mathbb{E}(T_i) + \mathbb{E}(T_{n-1-i})$$

$$\mathbb{E}(n) = 2(n+1)H_n - 4n$$

Dove H_n è l'n-esimo numero armonico, cioè $H_n = \sum_{i=1}^{n} \frac{1}{i}$.

2.1.3 Esempio: commutatività di matrici

L'approccio deterministico usualmente termina in $\mathscr{O}(k^3)$ o $\mathscr{O}(k^\alpha)$ con $2 < \alpha < 3$.

La procedura probabilistica cerca di determinare se $\boldsymbol{AB} \neq \boldsymbol{BA}$.

Algorithm 2: Riduzione della probabilità di errore

input : $\boldsymbol{A} = (A[1], \ldots, A[n]), A \in \mathbb{U}^n$
output: Il k-esimo elemento di A

```
1  begin
2      Scegli a ∈ {−1, 1}^k a caso in modo uniforme;
3      u = (a′ · A) B v = (a′ · B) A
4      if u ≠ v then
5          return "Sono diverse";
6      end
7      else
8          return "Sono uguali";
9      end
10 end
```

$$\mathbb{P}(\text{errore}) = \mathbb{P}\left(\underline{\boldsymbol{u}} = \underline{\boldsymbol{b}}, \boldsymbol{AB} = \boldsymbol{BA}\right)$$

$$\boldsymbol{C} = \boldsymbol{AB} - \boldsymbol{BA} \neq 0 \Rightarrow \exists \text{ una colonna di } \boldsymbol{C} : \underline{\boldsymbol{c}}_i \neq \underline{\boldsymbol{0}}$$

Supponiamo che $\underline{\boldsymbol{c}}_i \neq \underline{\boldsymbol{0}}$, sapendo che $\underline{\boldsymbol{a}}\boldsymbol{AB} = \underline{\boldsymbol{a}}\boldsymbol{BA}$. Prendiamo $\underline{\boldsymbol{a}} = \begin{bmatrix} a_1 & a_2 & \ldots & a_k \end{bmatrix}$, $a_i \in \{-1, 1\}$. Il prodotto quando $\underline{\boldsymbol{u}} = \underline{\boldsymbol{v}}$ risulta essere:

$$\underline{\boldsymbol{a}}'\boldsymbol{AB} = \underline{\boldsymbol{a}}'\boldsymbol{BA} \Rightarrow \underline{\boldsymbol{a}}'\underline{\boldsymbol{c}} = 0 \Rightarrow \underline{\boldsymbol{a}}' \begin{bmatrix} c_1 \\ \vdots \\ c_k \end{bmatrix} = 0 \Rightarrow \sum_{i=1}^{k} a_i c_i = 0$$

Modifichiamo il vettore:

$$\underline{\boldsymbol{\alpha}} = \begin{bmatrix} -1 & a_2 & \ldots & a_k \end{bmatrix} \qquad \underline{\boldsymbol{\beta}} = \begin{bmatrix} 1 & a_2 & \ldots & a_k \end{bmatrix}$$

Uno di questi due vettori deve coincidere con A.

$$\underline{\boldsymbol{\alpha}}'\boldsymbol{C} \quad \vee \quad \underline{\boldsymbol{\beta}}'\boldsymbol{C} = 0$$

È importante notare che non può capire che entrambe le relazioni siano vere allo stesso tempo poiché implicherebbe, ricordando la premessa $\underline{\boldsymbol{c}}_i \neq 0$, che $\underline{\boldsymbol{c}}_i = 0$.

2.2 Las Vegas

Sono sempre esatti o ritornano 'impossibile determinare soluzione' (spesso rappresentato con un punto di domanda).

2.2.1 Esempio: RSEL

Si tratta di un algoritmo **Las Vegas** utilizzato per estrarre il k-esimo elemento da un vettore definito su dominio $\mathbb{U}$ totalmente ordinato (non esistono termini intercambiabili nell'ordine).

Algorithm 3: Riduzione della probabilità di errore

input : $A, B \in \mathbb{N}^{k \times k}$
output: Il k-esimo elemento di A

```
1  begin
2      if n=2 then
3          Risolvi il problema direttamente.
4      end
5      else
6          Scegli t ∈ {1,2,...,n} a caso in modo uniforme. Calcola A ⩽ {A[j], A[j] < A[t]}. Calcola A ⩾ {A[j], j ≠ t, A[j] ⩾ A[t]}.
7          if #{A} ⩽ k − 1 then
8              return A[t]
9          else if #{A<} ⩾ k then
10             return RSEL(A<, k)
11         else
12             return RSEL(A>, k − #{A<} − 1)
13         end
14     end
15 end
```

$$T(n,k) \leqslant T_{\text{Quicksort}}(n)$$

Caso pessimo $T(n,k) = \mathcal{O}\left(n^2\right)$

Caso medio $T(n,k) = \mathcal{O}\left(n \log n\right)$

2.2.2 Numero medio di confronti in RSEL

$$T(n,k) = n - 1 + \frac{1}{n}\sum_{l=k}^{n-1} T(l,k) + \frac{1}{n}\frac{1}{n}\sum_{l=0}^{k-2} T(n-1-l, k-l-1)$$

Con $k \leqslant n, \#\{A_<\} = l$.

2.2.3 Esempio: Protocollo 10

Si tratta di un algoritmo **Las Vegas** per la comunicazione, che vuole stabilire se esiste un termine condiviso tra le due liste: $\exists i : x_i = y_i$.

$$\begin{aligned} A &: x_i, x_2, \ldots, x_1 0, \qquad x_i \in \{0,1\}^n \qquad \forall i \in 1, \ldots, 10 \\ B &: y_i, y_2, \ldots, y_1 0, \qquad y_i \in \{0,1\}^n \qquad \forall i \in 1, \ldots, 10 \end{aligned}$$

1. A genera 10 numeri primi $p_1, p_2, \ldots, p_10$, anche uguali, poiché estratti a caso in modo uniforme in $[0, n^2]$.

   A calcola $r_i = \text{resto}\left(\frac{x_i}{p_i}\right) \quad \forall i, \ldots, 10$

   A invia le 10 coppie di resti e numeri primi.

2. B calcola $s_i = \text{resto}\left(\frac{y_i}{p_i}\right) \quad i = 1, \ldots, 10.$

   Se sono tutti diversi allora B risponde "NO".

   Altrimenti sia $j = \min\{i : r_i = s_i\}$ e B invia (y_j, J) ad A.

3. In quest'ultimo caso (2) A verifica se $x_j = y_j$ e risponde "SI", altrimenti risponde "non so".

L'algoritmo termina in $10n$ messaggi.

L'algoritmo nel caso di risposta "NO" richiede $10 \cdot n \cdot (2\log_2 n) + 1 = 40\log_2 n + 1$ e nel caso "SI" oppure "?" richiede $40\log n + 4 + n + 1$.

Le risposte "SI" e "NO" sono corrette.

2.2.4 Probabilità di errore del protocollo 10

Primo Caso $\mathbb{P}\left(? \mid \forall i\, x_i \neq y_i\right) = \mathbb{P}\left(\exists l \mid r_l = s_l : \forall i\, x_i \neq y_i\right) \leqslant \sum_{l=1}^{1} 0\mathbb{P}\left(r_l = s_l : \forall i\, x_i \neq y_i\right) = 10 \cdot \frac{2\ln n}{n} = \frac{10\ln n}{n} \Rightarrow n \geqslant 40\ln n$

Secondo Caso

$$\begin{aligned}
\mathbb{P}\left(? \mid \exists l : x_l = y_l\right) &\leqslant \mathbb{P}\left(\exists j < l : r_j = s_j \;\middle|\; l = \#\{i : x_i = y_i\}\right) \\
&\leqslant \mathbb{P}\left(\exists j < 10 : r_j = s_j \;\middle|\; 10 = \min\{i : x_i = y_i\}\right) \\
&\leqslant g \cdot \mathbb{P}\left(r_1 = s_1 \mid x_1 \neq y_1\right) \\
&\leqslant g \cdot \frac{2\ln n}{n} = 18 \cdot \frac{\ln n}{n} \leqslant \frac{1}{2} \\
&\Rightarrow n \geqslant 40\ln n
\end{aligned}$$

2.3 Errore limitato (2-Sided Error)

$$f : I \to O, \qquad \exists \epsilon > 0 \quad \mathbb{P}\left(A(x)) f(x)\right) \geqslant \frac{1}{2} + \epsilon$$

La probabilità di rispondere correttamente è $\mathbb{P}$ (corretto) $\geqslant \frac{1}{2} + \epsilon$. Non si è mai assolutamente certi della risposta ottenuta ma si sceglie quella che appare più frequentemente. Generalmente si usa la **disuguaglianza di Chernoff** per determinare il numero di ripetizioni dell'algoritmo per avere un errore accettabile.

L'errore ϵ non dipende da x.

2.4 Errore Illimitato

$$f : I \to O, \qquad \exists \epsilon > 0 \quad \mathbb{P}\left(A(x)) f(x)\right) \geqslant \frac{1}{2} + \epsilon(x)$$

Sono una tipologia di algoritmi probabilistici usati per stimare funzioni ma **non ho mai** la certezza di una risposta corretta.

L'errore ϵ dipende dalla x.

3

Catene di Markov

3.1 Matrici e grafi

3.1.1 Grafo orientato

Si tratta di una coppia $(\mathcal{V}, \mathcal{E})$, dove $\mathcal{V}$ rappresenta un insieme finito di elementi detti **nodi** (o vertici) mentre $\mathcal{E}$ rappresenta un insieme ordinato di nodi che vengono chiamati **lati** (edges).

3.1.2 Matrice associata a grafo

Ogni matrice non negativa può essere associata a un grafo orientato nel quale i lati sono pesati con valori positivi.

3.1.3 Cammino

Si tratta di una sequenza di nodi del tipo $c = v_0, v_1, \ldots, v_n$ non necessariamente distinti tali che la tupla $\left(v_{i-1}, v_{v_i}\right) \in \mathcal{E}$. Se esiste un cammino da un nodo v_0 a v_k la sua lunghezza è k.

3.1.4 Cammino semplice

Un cammino è detto semplice se tutti i suoi nodi sono distinti tranne al più il primo e l'ultimo.

3.1.5 Ciclo

Si tratta di un cammino con lunghezza non nulla nel quale primo e ultimo nodo coincidono.

3.1.6 Grafo fortemente connesso

Una grafo in cui per ogni coppia di nodi esiste una arco.

3.1.7 Componente fortemente connessa

Un sotto-grafo formato da una classe C e dai lati di C_e che connettono tali nodi.

3.1.8 Classe essenziale

Una classe C è detta essenziale se $\forall v_i \in C \quad v_i \rightarrow v_j \Rightarrow v_j \in C$.

Non è possibile uscire da una classe essenziale seguendo un cammino che porta ad uno dei suoi nodi.

3.1.9 Nodo essenziale

Ogni nodo appartenente ad una classe essenziale viene detto essenziale $\Leftrightarrow \forall v_j : v_i \rightarrow v_j \Rightarrow v_j \rightarrow v_i$.

3.1.10 Periodicità

Dato un grado $\mathcal{G}$, consideriamo v_i, un nodo per il quale passa almeno un ciclo. Il **periodo** $d(v_i)$ di v_i è il massimo comun divisore delle lunghezze dei cicli che passano per v_i.

$$d(v_i) = \text{mcd}\left\{n \in \mathbb{N} : v_i \xrightarrow{n} v_j\right\}$$

Figura 3.1: Periodo di un nodo

Ogni nodo con un **auto-anello** è detto **aperiodico**. I nodi appartenenti alla stessa classe hanno lo stesso periodo.

3.1.11 Matrice di adiacenza

Matrice che assume solo valori binari: $(0, 1)$.

3.1.12 P

Si tratta di una matrice quadrata non negativa con grafo $\mathcal{G}$ associato che $\forall n \in \mathbb{N}, \forall (i, j) : a_{ij}^{(n)} = \sum$ passi di tutti i cammini di lunghezza n da v_i

Nelle matrici di **adiacenza** $a_{ij}^{(n)}$ denota il numero di cammini da $v_i \to v_j$ con lunghezza n.

3.1.13 Matrice irriducibile

Una matrice non-negativa $\boldsymbol{A}$ si dice **irriducibile** se $\forall (i, j) \exists n \in \mathbb{N} : a_{ij}^n > 0$.

3.1.14 Proprietà di una matrice irriducibile

1. $\forall i = 1, \ldots, k \exists N(i) : \forall n \in N(i) \quad a_{ij}^{(n)} > 0$.
2. $\forall i = 1, \ldots, k \exists r \in \mathbb{N} : 0 \leqslant r \leqslant d \wedge a_{ij}^{(s)} > 0 \Rightarrow s = r \mod (d)$
3. $\forall i, j = 1, \ldots, k \exists q(i, j) \in \mathbb{N} : \forall n \geqslant q a_{ij}^{(r+nd)} > 0$

3.1.15 Matrice riducibile

Quando non è irriducibile, cioè se:

1. Se e solo se il grafo associato possiede almeno due componenti totalmente connesse, o equivalentemente,
2. Se e solo se mediante una permutazione di righe e delle colonne può essere trasformata in una matrice triangolare a blocchi.

3.1.16 Matrici primitive

Definizione 3.1.1 (Matrice primitiva). Ogni matrice $\boldsymbol{A} \in \mathbb{R}_+^{k \times k}$ è primitiva se e solo se:

1. È aperiodica.
2. È irriducibile

Matrice primitiva. Supponiamo che $\boldsymbol{A}$ sia irriducibile e abbia periodo $d = 1$. Sappiamo che $\forall i, j \in 1, \ldots, k \exists q(i, j) \in \mathbb{N} : a_{ij}^{(r+nd)} > 0 \Rightarrow a_{ij}^{(n)} > 0$. Chiamiamo $M = \max\{q(i, j) : i, j = 1, \ldots, k\} \Rightarrow \forall n > M, a_{ij}^{(n)} > 0 \forall i, j$. Questo significa che $\boldsymbol{A}^n > 0$ e quindi $\boldsymbol{A}$ è primitiva. □

3.1.17 Proprietà delle matrici irriducibili (Teorema di Perron-Frobenius)

Teorema 3.1.2 (Teorema di Perron-Frobenius). Sia $\boldsymbol{A} \geqslant 0$ una matrice primitiva. Allora esiste un autovalore λ di $\boldsymbol{A}$, detto autovalore di Perron-Frobenius, che gode delle seguenti proprietà:

1. $\lambda > 0$
2. $\forall \mu \neq \lambda \Leftrightarrow |\mu| < \lambda$
3. λ ammette autovettori destri e sinistri strettamente positivi.
4. λ è radice di $\det(\boldsymbol{I}\underline{\boldsymbol{x}} - \boldsymbol{A})$

3.1.18 Vettori stocastici

Un vettore $\underline{\boldsymbol{v}} \in \mathbb{R}_+^{k\times k}$ è stocastico se:

1. $v_i \geqslant 0 \quad \forall i = 1,\dots,k$
2. $\sum_{i=1}^{n} v_i = 1$

3.1.19 Matrice stocastica

Una matrice è stocastica se e solo se:

1. $a_{ij} \geqslant 0$
2. $\forall i, \sum_{i=1}^{n} a_{ij} = 1$

3.1.20 Proprietà delle matrici stocastiche

1. Ogni matrice stocastica possiede autovalore 1, ed è il suo autovalore di Perron-Frobenius.
2. La matrice risultante dal prodotto di due matrici stocastiche è stocastica.
3. Una matrice è **bi-stocastica** se e solo se l'autovettore sinistro di $\boldsymbol{A}$ corrisponde all'autovettore dell'autovalore1:

$$\sum_{i=1}^{n} = 1 \quad \wedge \quad \sum_{j=1}^{n} a_{ij} = 1$$

4. Se $\boldsymbol{A} > 0$ è stocastica e primitiva allora 1 è l'unico autovalore di modulo massimo per $\boldsymbol{A}$.
5. Se μ è un autovettore di una matrice stocastica, allora $|\mu| < 1$.

   Dimostrazione. $\exists \underline{\boldsymbol{v}}' = \left[v_1,\dots,v_k\right]$ autovettore sinistro di $\boldsymbol{A}$ corrispondente a $\mu, \underline{\boldsymbol{v}}'\boldsymbol{A} = \mu\underline{\boldsymbol{v}}'$:

$$\mu \cdot \sum_{i=1}^{k} v_i = \sum_{i=1}^{k} |\mu v_i| = \sum_{i=1}^{k} \left| \sum_{j=1}^{k} v_j \cdot a_{ij} \right| = \sum_{i=1}^{k} \sum j = 1^{k} |v_j \cdot a_{ij}| = \sum_{j=1}^{k} |v_j| \sum_{i=1}^{k} a_{ij} = |\mu| \cdot \sum_{i=1}^{k} |v_i| \leqslant \sum_{j=1} |v_i| \leqslant \sum_{j=1}^{k} |v_j| \Rightarrow |\mu| \leqslant 1$$

   □

3.2 Definizione di una catena di Markov

Una catena di Markov è definita da 3 quantità:

1. Un insieme finito $S = \{1,2,\dots,k\}$ di stati.
2. μ una distribuzione di probabilità di un insieme $\mu = (\mu_1,\dots,\mu_n)$.
3. $\boldsymbol{P}$, una matrice stocastica $\sum righe = 1$.

3.3 Proprietà fondamentali sulle catene

Una catena di Markov finita e omogenea definita sulla tripla $S, \mu, \boldsymbol{P}$ è una successione $\{X_u\}_{n\in N}$ di variabili aleatorie a valori in S che soddisfano le seguenti proprietà seguenti:

1. $\forall i \in S \qquad \mathbb{P}(X_0 = i) = \mu(i)$
2. **Proprietà di Markov**: la probabilità di essere al passo $n+1$ dipende solo dal passo n.

$$\forall n > 0,\ \forall i, j \in S\ \forall i_0, \ldots, i_{n-1} \in S \qquad \mathbb{P}\left(X_{n+1} = j \mid X_0 = i_0, x_1 = i_1, \ldots, X_n = i\right) = \mathbb{P}\left(X_{n+1} = j \mid X_n = i\right)$$

3. $\forall n \in \mathbb{N}, \forall i, j \in S \quad \mathbb{P}\left(X_{n+1} = j \mid X_n = i\right) = p(i, j)$

Proposizione 3.3.1. Per ogni $n, k \in \mathbb{N}$ e $\forall (i, j) \in S$, la $\mathbb{P}\left(X_{n+k} = j \mid X_k = i\right) = p^{(n)}(i, j)$.

Dimostrazione. Procediamo per induzione su n:

Caso con $n = 0$: $\mathbb{P}\left(X_{n+k} = j \mid X_k = i\right) = \mathbb{P}\left(X_k = j \mid X_k = i\right) = p^{(0)}(i, j)$

Caso con $n = 1$: $\mathbb{P}\left(X_{n+k} = j \mid X_k = i\right) = p(i, j)$ (da proprietà 3).

Dimostriamo per $n = n + 1$: Sappiamo che per ogni tripla di eventi $(\,) A, B, C)$ vale che:

$$\mathbb{P}(A \cap B \mid C) = \mathbb{P}(B \mid C)\,\mathbb{P}(A \mid B \cap C)$$

Da cui otteniamo la seguente catena di uguaglianze:

$$\begin{aligned}
\mathbb{P}_\mu\left(X_{n+k+1} = j \mid X_k = i\right) &= \mathbb{P}_\mu\left(X_{n+k+1} = j, \exists l \in S : X_{k+1=l} \mid X_k = i\right) \\
&= \sum_{l\in S} \mathbb{P}_\mu\left(X_{k+n+1} = j, X_{k+1} = l \mid X_k = i\right) \\
&= \sum_{l\in S} \mathbb{P}_\mu(X_{k+1} = l \mid X_k = i) \cdot \mathbb{P}\left(X_{k+n+1} = j \mid X_{k+1} = l, X_k = i\right)
\end{aligned}$$

Notiamo che la $\mathbb{P}_\mu(X_{k+1} = l \mid X_k = i) > 0$ implica che $\mathbb{P}\left(X_{k+1} = l, X_k = i \mid \mu\right) > 0$. Quindi, per la **proprietà di Markov** si ottiene che:

$$\begin{aligned}
\mathbb{P}_\mu\left(X_{n+k+1} = j \mid X_k = i\right) &= \sum_{l\in S} \mathbb{P}_\mu(X_{k+1} = l \mid X_k = i) \cdot \mathbb{P}_\mu\left(X_{k+n+1} = j \mid X_{k+1} = l\right) \\
&= \sum_{l\in S} p(i, l) \cdot \mathbb{P}\left(X_{n+k+1} = j \mid X_{k+1} = l\right)
\end{aligned}$$

che per ipotesi di induzione diventa:

$$\mathbb{P}\left(X_{k+n+1} = j \mid X_k = i\right) = \sum_{l\in S} p(i, l) \cdot p^{(n)}(l, j) = p^{(n+1)}(i, j)$$

□

3.4 Stati ricorrenti

La ricorrenza è una proprietà della classe. Se $i \in S$ è uno stato ricorrente e C è la sua classe, allora per ogni stato $j \in C$ è ricorrente.

Se uno stato i è ricorrente $i \to j$ allora anche j è ricorrente.

3.4.1 Prima definizione

Definizione 3.4.1 (Stato ricorrente). Uno stato $i \in S$ è ricorrente se e solo se:

$$\mathbb{P}_i(X_n = i \quad \forall n > 0) = 1$$

3.4.2 Seconda definizione

Definizione 3.4.2 (Stato ricorrente). Uno stato $i \in S$ è ricorrente se e solo se $f(i,i) = 1$:

$$\mathbb{P}_i(\tau_i = +\infty) = 0$$

Dove τ_i è il tempo di attesa necessaria per entrare in i per la prima volta.

3.4.3 Terza definizione

Definizione 3.4.3 (Stato ricorrente). Un stato $i \in S$ è ricorrente se e solo se:

$$\sum_{n \geqslant 0} p^{(n)}(i,i) = +\infty$$

3.4.4 Stati essenziali

3.4.5 Gli stati ricorrenti sono stati essenziali

Teorema 3.4.4 (Stati ricorrenti = stati essenziali). Gli stati ricorrenti sono stati essenziali.

Dimostrazione. Sia $i \in S$ essenziale e C sia la sua classe $\Rightarrow \forall n \in \mathbb{N}, \forall i \in C \quad \sum_{j \in C} p^{(n)}(i,j) = 1$. Procediamo per assurdo: i è transiente $\Rightarrow$ ogni $j \in C$ è transiente.

$$\sum_{j \in C} p^{(n)}(i,j) < +\infty, \quad p^{(p)}(i,j) \to 0 \quad \text{per } n \to +\infty$$

Allora:

$$f = \sum_{j \in C} p^{(n)}(i,j) = \lim_{n \to n} \sum_{j \in C} p^{(n)}(i,j) \quad \wedge \quad C \text{ è finito } \Rightarrow \sum_{j \in C} \lim_{n \to \infty} p^{(n)}(i,j) \to 0$$

Da cui l'assurdo. □

3.5 Tempi medi di rientro

3.5.1 Caso degli stati ricorrenti (*)

Teorema 3.5.1 (Tempi di rientro negli stati ricorrenti). $\forall i \in S$ ricorrente il valor medio di tempo di rientro è $< \infty$.

$$\mathbb{E}_i(\tau_i) < +\infty$$

Tempi di rientro negli stati ricorrenti. Sia C la classe di i tale che $\sum_{j \in C} p(i,j) = 1$.

Vogliamo provare che $\mathbb{E}_i(\tau_i) = \sum_{n \geqslant 1} n f^n(i,i)$, cioè $f^n(i,i) = \mathcal{O}(()\epsilon^n)$.

Andiamo a costruire una nuova catena su $(S, \mu, \tilde{p})$, dove $\tilde{p} \to$ è una matrice di ransizione che è uguale alla precedente ma con i assorbente, cioè:

$$\begin{cases} \tilde{p}(u,v) = p(u,v) & \text{se } u \neq i, \quad \forall v \in C \\ \tilde{p}(i,u) = 1 & v = i \\ \tilde{p}(i,u) = 0 & v \neq i \end{cases}$$

Nella nuova catena i è l'unico stato ricorrente e $\forall u \in T, \quad n \neq i$, n è transiente.

$$\forall n \geqslant 2, \quad f^{(n)}(i,i) = \sum_{u,v \in C \wedge u \neq i \neq v} p(i,n)\tilde{p}^{n-2}(u,v)p(v,i)$$

Applicando la proprietà della lemma ottengo:

$$f^n(i,i) \leqslant \mathcal{O}\left(\epsilon^n\right) \cdot \sum_{k,j \in C,\, k,j \neq i} p(i,k)p(j,i) = \mathcal{O}\left(\epsilon^n\right)$$

□

3.6 Definizione di catena ergodica

3.7 Esistenza di distribuzioni stazionarie (*)

3.8 Ergodicità delle catene con matrici di transizione primitive (*)

4

Applicazioni algoritmiche

4.1 Catene reversibili

4.2 Passeggiate a cso su grafi

4.2.1 Algoritmo probabilistico per 2-SODD

4.3 Metodo MCMC (Monte Carlo Markov Chain): rappresentazione di algoritmo e proprietà

4.3.1 Generazione di insiemi indipendenti in grafi (anche di dimensione fissata)

4.4 Algoritmo di Metropolis

4.5 Campionatore di Gibbs

5

Velocità di convergenza degli algoritmi MCMC

5.1 La problematica

5.2 Come stimare la velocità di convergenza

5.3 Stimare il numero di passi

5.4 Metodo generale basato sul coefficiente ergodico

6 Coupling

6.1 Metodo di accoppiamento (Coupling method)

6.1.1 Caso del campionatore di set indipendenti di dimensione fissa

6.1.2 Campionatore di colorazioni

7 Colorazioni di grafi

7.1 Stima del numero di colorazioni di un grafo (algoritmo ed enunciati)