

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
5 luglio 2018

Indice

1	Probabilità	3
1.1	Disuguaglianza di Chernoff	3
1.1.1	Caratteristiche	3
1.1.2	Applicazioni alle binomiali	3
1.1.3	Esempio: Riduzione della probabilità di errore di un algoritmo	4
1.1.4	Esempio: Lancio di monete	5
1.1.5	Esempio: Intervalli di confidenza	5
2	Algoritmi probabilistici	7
2.1	1-Sided Error	7
2.1.1	Esempio: Problema di protocollo di comunicazione	8
2.1.2	Esempio: commutatività di matrici	8
2.2	Las Vegas	9
2.2.1	Esempio: RSEL	9
2.2.2	Esempio: Quicksort	10
2.2.3	Numero medio di confronti in RSEL	10
2.2.4	Esempio: Protocollo 10	11
2.2.5	Probabilità di errore del protocollo 10	11
2.3	Errore limitato (2-Sided Error)	11
2.4	Errore Illimitato	11
3	Catene di Markov	12
3.1	Matrici e grafi	12
3.1.1	Grafo orientato	12
3.1.2	Matrice associata a grafo	12
3.1.3	Cammino	12
3.1.4	Cammino semplice	12
3.1.5	Ciclo	12
3.1.6	Grafo fortemente connesso	12
3.1.7	Componente fortemente connessa	12
3.1.8	Classe essenziale	12
3.1.9	Nodo essenziale	12
3.1.10	Periodicità	13
3.1.11	Teorema per la periodicità	13
3.1.12	Matrice di adiacenza	13
3.1.13	P	13
3.1.14	Matrice irriducibile	13
3.1.15	Proprietà di una matrice irriducibile	13
3.1.16	Matrice riducibile	14
3.1.17	Matrici primitive	14
3.1.18	Proprietà delle matrici irriducibili (Teorema di Perron-Frobenius)	14
3.1.19	Matrice stocastica	14
3.1.20	Proprietà delle matrici stocastiche	14
3.2	Definizione di una catena di Markov	15
3.3	Proprietà fondamentali sulle catene	15
3.4	Stati ricorrenti	16
3.4.1	Prima definizione	16
3.4.2	Seconda definizione	16

3.4.3	Terza definizione	16
3.4.4	Stati essenziali	16
3.4.5	Gli stati ricorrenti sono stati essenziali	16
3.5	Tempi medi di rientro	16
3.5.1	Caso degli stati ricorrenti (*)	16
3.6	Definizione di catena ergodica	17
3.7	Distribuzioni stazionarie	17
3.7.1	Proprietà dei coefficienti di una distribuzione stazionaria	17
3.8	Esistenza di distribuzioni stazionarie (*)	18
3.9	Catene primitive	19
3.10	Ergodicità delle catene con matrici di transizione primitive (*)	19
4	Applicazioni algoritmiche	20
4.1	Catene reversibili	20
4.2	Passeggiate a caso su grafi	20
4.2.1	Algoritmo probabilistico per 2-SODD	20
4.3	Metodo MCMC (Monte Carlo Markov Chain): rappresentazione di algoritmo e proprietà	20
4.3.1	Generazione di insiemi indipendenti in grafi (anche di dimensione fissata)	20
4.4	Algoritmo di Metropolis	20
4.5	Campionatore di Gibbs	20
5	Velocità di convergenza degli algoritmi MCMC	21
5.1	La problematica	21
5.2	Come stimare la velocità di convergenza	21
5.3	Stimare il numero di passi	21
5.4	Metodo generale basato sul coefficiente ergodico	21
6	Coupling	22
6.1	Metodo di accoppiamento (Coupling method)	22
6.1.1	Caso del campionario di set indipendenti di dimensione fissa	22
6.1.2	Campionatore di colorazioni	22
7	Colorazioni di grafi	23
7.1	Stima del numero di colorazioni di un grafo (algoritmo ed enunciati)	23

1.1 Disuguaglianza di Chernoff

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

Figura 1.1: Disuguaglianza di Chernoff per le binomiali

La **disuguaglianza di Chernoff** è utilizzata 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

Possiede 3 distinte 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.

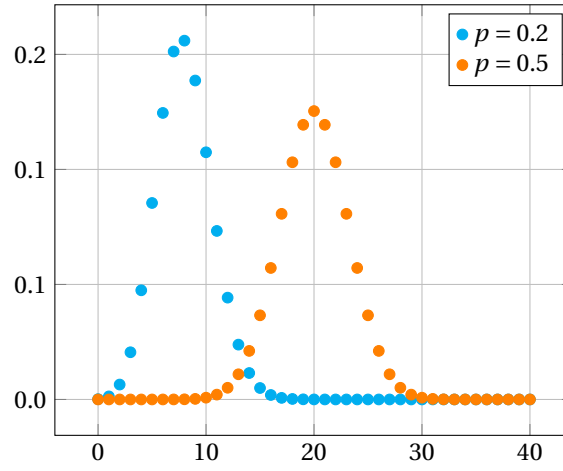
1.1.2 Applicazioni alle binomiali

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

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

$$\mathbb{P}(|X - \mathbb{E}(X)| \geq a) \leq \frac{\text{Var}(X)}{a^2} \quad \forall a > 0$$

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 $\text{Var}(X) = npq$, $\forall \epsilon > 0$ vale che:

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

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

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

Applico il limite ed ottengo:

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

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

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

Maggioro l'integrale:

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

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 $f_I \rightarrow O$ difficile da calcolare e di disporre di un algoritmo probabilistico \mathcal{A} tali che $\forall x \in I, T_{\mathcal{A}} \leq p(|x|)$ dove p è un polinomio di grado **piccolo** e che $\mathbb{P}(A(x) = f(x)) \geq \frac{3}{4}$, 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 \mathcal{A}_t l'algoritmo probabilistico che ripete \mathcal{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  $i \in [1, \dots, t]$  do
3     Esecuzioni indipendenti di  $\mathcal{A}$  su  $x$ ;
4      $A[i] = \mathcal{A}(x)$ 
5   end
6   if  $\exists z \in (A[1], \dots, A[t]) : \#\{j \in \{1, \dots, t\} : z = A[j]\} > \frac{t}{2}$  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:

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

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

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

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

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

1.1.4 Esempio: Lancio di monete

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

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

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

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

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

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

Nuovamente risolvendo per $n = 100$ si ottiene:

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

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:

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

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

$$\begin{aligned}\mathbb{P}(|p - \bar{p}| \geq \delta) &= \mathbb{P}(|np - X_{n,p}| \geq n\delta) \leq 2e^{-2\delta^2 n} \leq t \\ -2\delta^2 n &\leq \log \frac{2}{t} \Rightarrow n \leq \frac{1}{2\delta^2} \log \frac{2}{t}\end{aligned}$$

Dove t rappresenta la probabilità di errore.

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

$$n \leq 50 \log 200 = 50 (\log 2 + 2 \log 10) \approx 250 \Rightarrow n \geq 250$$

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.

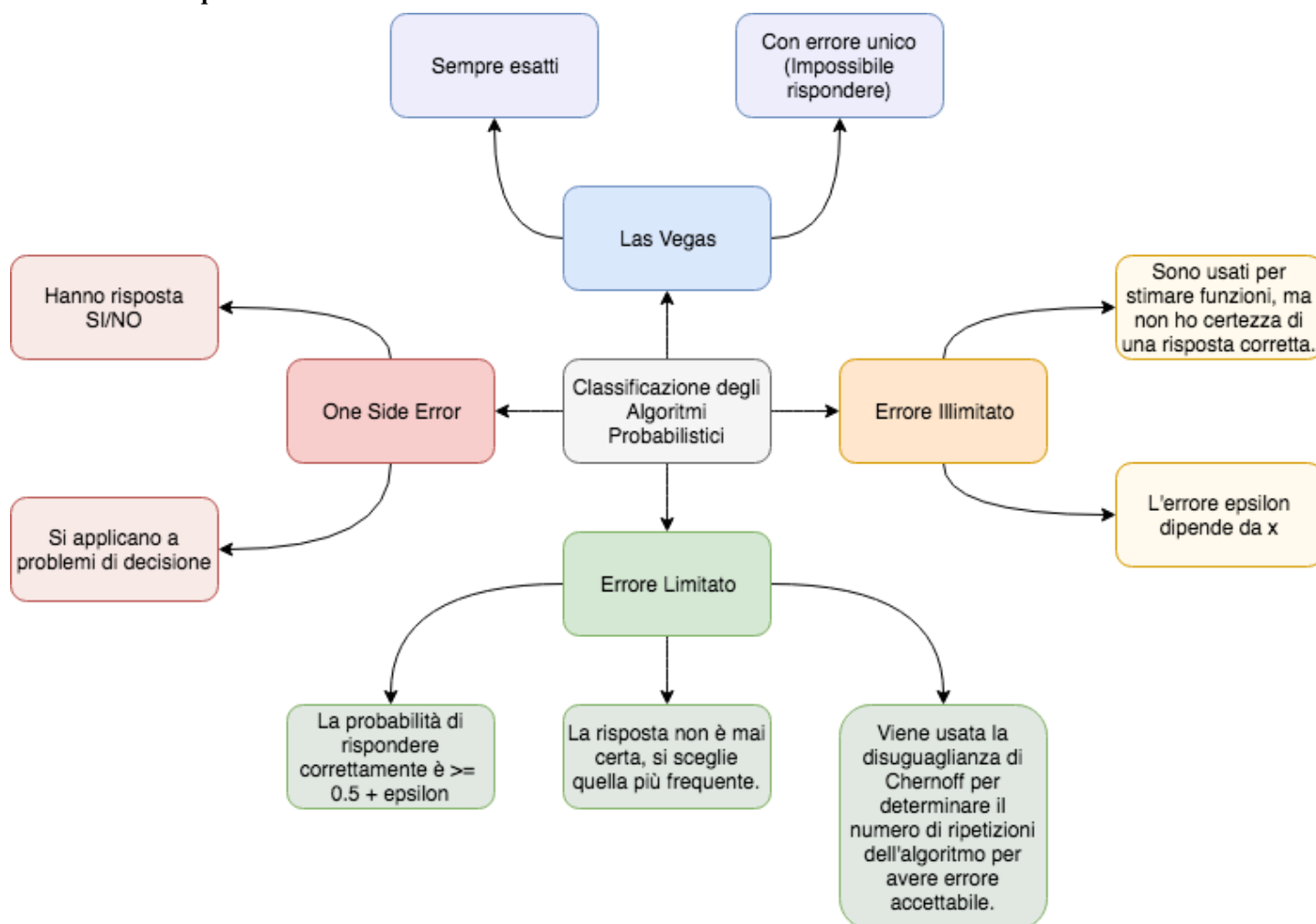


Figura 2.1: Mappa degli algoritmi probabilistici

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.

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

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

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

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 \mathcal{A} restituisce “diverso” la risposta è certamente corretta, mentre se restituisce “uguale” potrebbe essere sbagliato, con una probabilità pari a:

$$\mathbb{P}(\text{errore} \mid \mathcal{A} \text{ dice uguali}) = \mathbb{P}(p \text{ divide } a-b \mid a \neq b)$$

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:

$$\text{prim}(m) \approx \frac{m}{\ln m} \quad \text{ovvero} \quad \frac{\text{prim}(m)}{m} \ln m \rightarrow 1$$

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:

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

$$\mathbb{P}(\text{errore}) \leq \frac{n-1}{n^2 / 2 \log_2 n} \leq \frac{2 \log_2 n}{n} \rightarrow 0$$

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

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: commutatività di matrici

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

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

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

Algorithm 2: Commutatività di matrici

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

```

1 begin
2   Scegli  $\underline{a} \in \{-1, 1\}^k$  a caso in modo uniforme;
3    $\underline{u} = (\underline{a}' \cdot A) B$ ;
4    $\underline{v} = (\underline{a}' \cdot B) A$ ;
5   if  $\underline{u} \neq \underline{v}$  then
6     | return “Sono diverse”;
7   end
8   else
9     | return “Sono uguali”;
10  end
11 end

```

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

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

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

Modifichiamo il vettore:

$$\underline{\alpha} = \begin{bmatrix} -1 & a_2 & \dots & a_k \end{bmatrix} \quad \underline{\beta} = \begin{bmatrix} 1 & a_2 & \dots & a_k \end{bmatrix}$$

Uno di questi due vettori deve coincidere con A .

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

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

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).

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

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

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

Algorithm 3: RSEL

```

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  $t \in \{1, 2, \dots, n\}$  a caso in modo uniforme.;
7     Calcola  $A \leq \{A[j], A[j] < A[t]\}.$ ;
8     Calcola  $A \geq \{A[j], j \neq t, A[j] \geq A[t]\}.$ ;
9     if  $\# \{A_{<}\} \leq k-1$  then
10      return  $A[t]$ 
11     else if  $\# \{A_{<}\} \geq k$  then
12      return  $\text{RSEL}(A_{<}, k)$ 
13     else
14      return  $\text{RSEL}(A_{>}, k - \# \{A_{<}\} - 1)$ 
15     end
16   end
17 end

```

2.2.2 Esempio: Quicksort

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

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

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

Definizione 2.2.1 (Formula delle probabilità totali). Sia (Ω, \mathcal{F}, P) uno spazio di probabilità e $F_1, F_2, \dots, F_n \in \mathcal{F}$ 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 $k = 1, 2, \dots, n$. Allora ogni evento $E \in \mathcal{F}$ si ha:

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

Corollario 2.2.1.1 (Corollario sulla media).

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

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

$$\begin{aligned} \mathbb{E}(T_n) &= \sum_{i=0}^{n-1} \mathbb{E}(T_n \mid B_i) \mathbb{P}(B_i) = n-1 + \sum_{i=0}^{n-1} \mathbb{E}(T_i) + \mathbb{E}(T_{n-1-i}) \\ \mathbb{E}(n) &= 2(n+1)H_n - 4n \end{aligned}$$

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

2.2.3 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 $k \leq n, \# \{A_{<}\} = l$.

2.2.4 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: $\exists i : x_i = y_i$.

$$\begin{aligned} A : x_1, x_2, \dots, x_{10}, \quad x_i \in \{0, 1\}^n \quad \forall i \in 1, \dots, 10 \\ B : y_1, y_2, \dots, y_{10}, \quad y_i \in \{0, 1\}^n \quad \forall i \in 1, \dots, 10 \end{aligned}$$

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

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

A invia le 10 coppie di resti e numeri primi.

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

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.

2.2.5 Probabilità di errore del protocollo 10

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

Secondo Caso

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

2.3 Errore limitato (2-Sided Error)

$$f : I \rightarrow O, \quad \exists \epsilon > 0 \quad \mathbb{P} (A(x)) f(x) \geq \frac{1}{2} + \epsilon$$

La probabilità di rispondere correttamente è $\mathbb{P}(\text{corretto}) \geq \frac{1}{2} + \epsilon$. 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

$$f : I \rightarrow O, \quad \exists \epsilon > 0 \quad \mathbb{P} (A(x)) f(x) \geq \frac{1}{2} + \epsilon(x)$$

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 .

3.1 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 $c = v_0, v_1, \dots, v_n$ non necessariamente distinti tali che la tupla $(v_{i-1}, v_{v_i}) \in \mathcal{E}$. 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 un 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 .

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

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 Teorema per la periodicità

Teorema 3.1.1 (Teorema per la periodicità). Dato un grafo orientato \underline{G} consideriamo due nodi distinti i, j tali che $i \leftrightarrow j$. Allora $d(i) = d(j)$.

Teorema per la periodicità. Sia C un ciclo qualunque che passa per j e sia s la sua lunghezza.

Per ipotesi esistono cammini sia da i a j (chiamiamolo C_1) che da j a i (chiamiamolo C_2): chiamiamo le rispettive lunghezze u e v .

Proviamo innanzitutto che $d(i)$ è un divisore di s . I cammini C_1 e C_2 formano un ciclo di lunghezza $u + v$ passante per i , per cui $d(i)$ è un divisore di $u + v$. Analogamente esso deve essere un divisore anche per $u + v + s$ poichè C_1, C e C_2 formano un ciclo per i .

Ne segue che $d(i)$ divide anche la differenza: $(u + v + s) - (u + v) = s$ e quindi risulta essere divisore della lunghezza di qualsiasi ciclo passante per j .

Proviamo che $d(i) = d(j)$ Poichè $d(j)$ è il massimo di tali divisore, $d(i) \leq d(j)$. Ripetendo il ragionamento con i e j scambiati, si ottiene che:

$$d(i) \leq d(j) \quad \wedge \quad d(i) \geq d(j) \Rightarrow d(i) = d(j)$$

□

3.1.12 Matrice di adiacenza

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

3.1.13 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 a v_j .

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

3.1.14 Matrice irriducibile

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

3.1.15 Proprietà di una matrice irriducibile

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

3.1.16 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.

3.1.17 Matrici primitive

Definizione 3.1.2 (Matrice primitiva). Ogni matrice $A \in \mathbb{R}_+^{k \times k}$ è primitiva se e solo se:

1. È aperiodica.
2. È irriducibile

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

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

Teorema 3.1.3 (Teorema di Perron-Frobenius). Sia $A \geq 0$ una matrice primitiva. Allora esiste un autovalore λ di 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 $\det(Ix - A)$

3.1.19 Matrice stocastica

Una matrice è stocastica se e solo se:

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

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 A corrisponde all'autovettore dell'autovalore 1:

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

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

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

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

\square

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. P , una matrice stocastica $\sum \text{righe} = 1$.

3.3 Proprietà fondamentali sulle catene

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

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

$$\forall n > 0, \forall i, j \in S \forall i_0, \dots, i_{n-1} \in S \quad \mathbb{P}(X_{n+1} = j \mid X_0 = i_0, X_1 = i_1, \dots, X_n = i) = \mathbb{P}(X_{n+1} = j \mid X_n = i)$$

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

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

Dimostrazione. Procediamo per induzione su n :

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

Caso con $n = 1$: $\mathbb{P}(X_{n+k} = j \mid X_k = i) = p(i, j)$ (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:

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

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

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

che per ipotesi di induzione diventa:

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

□

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 $i \rightarrow j$ 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:

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

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.

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

Allora:

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

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

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

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

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

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

Applicando la proprietà della lemma ottengo:

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

□

3.6 Definizione di catena ergodica

Definizione 3.6.1 (Catena ergodica). Una catena di Markov finita $\{X_n\}$ su un insieme di stati S si dice **ergodica** se esiste un vettore stocastico $\underline{\pi}^* = (\pi_i^*)_{i \in S}$ tale che, $\forall i, j \in S$:

$$\lim_{n \rightarrow +\infty} \mathbb{P}_i(X_n = j) = \pi_j^* = \lim_{n \rightarrow +\infty} p^{(n)}(i, j)$$

Dove $P = [p(i, j)]_{i, j \in S}$ è la matrice di transizione della catena.

Cioè, al crescere di n , la distribuzione limite esiste e risulta indipendente dalla distribuzione iniziale.

3.7 Distribuzioni stazionarie

Definizione 3.7.1 (Distribuzione stazionaria). Data una catena di Markov finita $\{X_n\}$ sull'insieme degli stati $S = \{1, 2, \dots, k\}$, con matrice di transizione P , chiamiamo **distribuzione stazionaria** un vettore $\underline{\pi} \in \mathbb{R}^k$ tale che:

1. $\underline{\pi}$ è un vettore stocastico, ovvero $\pi_i \geq 0 \quad \forall i = 1, 2, \dots, k$.
2. $\sum_{i=1}^k \pi_i = 1$
3. $\underline{\pi}$ è un autovettore sinistro di P corrispondente a 1, ovvero $\underline{\pi}^T = \underline{\pi}^T P$

3.7.1 Proprietà dei coefficienti di una distribuzione stazionaria

Poichè ogni catena finita ammette sempre una classe ricorrente, senza perdita di generalità possiamo supporre che lo stato 1 sia ricorrente.

Per ogni $i \in S$, definiamo:

$$\rho_i = \sum_{n=0}^{+\infty} \mathbb{P}_1(X_n = i, \tau_1 > n)$$

Tali coefficienti godono delle seguenti 3 proprietà:

1. Ogni ρ_i è finito.
2. $\rho_1 = 1$
3. Ogni ρ_i rappresenta il numero medio di entrate nello stato i nell'intervallo di tempo $\{0, 1, \dots, \tau_1 - 1\}$ supponendo $X_0 = 1$.

Il vettore $\underline{\pi}$ viene definito come:

$$\underline{\pi} = \left(\frac{\rho_1}{\mathbb{E}_1(\tau_1)}, \frac{\rho_2}{\mathbb{E}_1(\tau_1)}, \dots, \frac{\rho_k}{\mathbb{E}_1(\tau_1)} \right)$$

Procediamo a dimostrare che il vettore così ottenuto sia **stocastico**:

Positività Ogni termine del vettore è certamente positivo.

Somma a 1 Procediamo a dimostrare che la somma dei termini nel vettore è unitaria. Tenendo a mente che il vettore ha una componente normalizzante pari a $\mathbb{E}_1(\tau_1)$ procediamo sui coefficienti:

$$\begin{aligned}\sum_{i=1}^k \rho_i &= \sum_{i=1}^k \sum_{n=0}^{+\infty} \mathbb{P}_1(X_n = i, \tau_1 > n) \\ &= \sum_{n=0}^{+\infty} \sum_{i=1}^k \mathbb{P}_1(X_n = i, \tau_1 > n) \\ &= \sum_{n=0}^{+\infty} \mathbb{P}_1(\tau_1 > n) \\ &= \mathbb{E}_1(\tau_1)\end{aligned}$$

Dividendo quindi per $\mathbb{E}_1(\tau_1)$ si ottiene 1.

3.8 Esistenza di distribuzioni stazionarie (*)

Teorema 3.8.1 (Esistenza di distribuzioni stazionarie). Sia $\{X_n\}$ una catena di Markov sugli stati $\{1, 2, \dots, k\}$ con matrice di transizione $P = [p(i, j)]$ e supponiamo che lo stato 1 sia ricorrente.

Allora il vettore $\underline{\pi}$ è una distribuzione stazionaria per $\{X_n\}$.

Dimostrazione. È sufficiente provare che il vettore $\underline{\rho} = \{\rho_1, \rho_2, \dots, \rho_k\}$ è autovettore sinistro di P corrispondente all'autovalore 1.

Consideriamo il caso $j \neq 1$ e proviamo che $\rho_j = (\underline{\rho}^T P)_j$.

$$\begin{aligned}\rho_j &= \sum_{n=1}^{+\infty} \mathbb{P}_1(X_n = j, \tau_1 > n) \\ &= \sum_{n=1}^{+\infty} \mathbb{P}_1(X_n = j, \tau_1 > n-1) \\ &= \sum_{n=1}^{+\infty} \sum_{i=1}^k \mathbb{P}_1(X_n = j, X_{n-1} = i, \tau_1 > n-1) \\ &= \sum_{n=1}^{+\infty} \sum_{i=1}^k \mathbb{P}_1(X_n = j \mid X_{n-1} = i, \tau_1 > n-1) \mathbb{P}_1(X_{n-1} = i, \tau_1 > n-1)\end{aligned}$$

Se $n > 1$ il primo termine coincide con $p(i, j)$, mentre se $n = 1$ il secondo termine risulta nullo.

$$\begin{aligned}\rho_j &= \sum_{n=1}^{+\infty} \sum_{i=1}^k p(i, j) \mathbb{P}_1(X_{n-1} = i, \tau_1 > n-1) \\ &= \sum_{i=1}^k p(i, j) \sum_{n=1}^{+\infty} \mathbb{P}_1(X_{n-1} = i, \tau_1 > n-1) \\ &= \sum_{i=1}^k p(i, j) \rho_i \\ &= (\underline{\rho}^T P)_j\end{aligned}$$

Consideriamo ora il caso $j = 1$:

$$\begin{aligned}
 \rho_1 &= \sum_{n=1}^{+\infty} \mathbb{P}_1(\tau_1 = n) \\
 &= \sum_{n=1}^{+\infty} \sum_{i=1}^k \mathbb{P}_1(X_{n-1} = i, \tau_1 = n) \\
 &= \sum_{n=1}^{+\infty} \sum_{i=1}^k \mathbb{P}_1(X_n = 1 \mid X_{n-1} = i, \tau_1 > n-1) \mathbb{P}_1(X_{n-1} = i, \tau_1 > n-1) \\
 &= \sum_{n=1}^{+\infty} \sum_{i=1}^k p(i, 1) \mathbb{P}_1(X_{n-1} = i, \tau_1 > n-1) \\
 &= \sum_{i=1}^k p(i, 1) \sum_{n=1}^{+\infty} \mathbb{P}_1(X_{n-1} = i, \tau_1 > n-1) \\
 &= \sum_{i=1}^k p(i, 1) \rho_i \\
 &= (\underline{\rho}^T \mathbf{P})_1
 \end{aligned}$$

Abbiamo quindi dimostrato che $\underline{\rho} = \underline{\pi} = (\underline{\rho}^T \mathbf{P})$

□

3.9 Catene primitive

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

4

Applicazioni algoritmiche

4.1 Catene reversibili

4.2 Passeggiate a caso 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

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 campionario di set indipendenti di dimensione fissa

6.1.2 Campionario di colorazioni

7

Colorazioni di grafi

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