Problemi di Occupancy e Balanced Allocation

Matteo Boscariol
Università degli Studi di Padova

Relatore: Prof. Andrea Pietracaprina

Anno accademico 2010/2011
30 settembre 2011

Indice

Sommario

In questa tesina si descrive e si analizza il processo *Balls and Bins* per ottenere delle approssimazioni per il carico massimo e il numero di scatole vuote, anche nei casi in cui il numero di palline non coincide con il numero di scatole. Si cita inoltre una variante che permette di ottenere una distribuzione del carico più bilanciata. Infine si discutono alcune applicazioni del processo in algoritmi e strutture dati: bucket sort, hash table e Bloom filters.

Capitolo 1

Introduzione

Questo elaborato ha lo scopo di illustrare il processo aleatorio noto come *Balls and Bins*, in particolare alcune sue caratteristiche e applicazioni in ambito informatico.

In alcune applicazioni si è interessati a conoscere il modo in cui un insieme di elementi è allocato in un insieme di risorse, ed eventualmente a trovare nuovi metodi di allocazione per bilanciare il carico delle risorse disponibili. Questi problemi sono di particolare interesse per algoritmi e strutture dati con delle componenti aleatorie (ad esempio, bucket sort e hash table).

Il processo *Balls and Bins* è composto da un numero m di palline (*Balls*) lanciate in modo casuale in n scatole (*Bins*). Ci sono molte varianti del problema, e in [2] è presente un'esaustiva discussione su questo e altri processi collegati. Tuttavia, buona parte dei risultati in [2] sono espressi in forma esatta, rendendo difficile ricavare intervalli di confidenza, limiti inferiori e superiori.

In [1] è presente invece una analisi che fornisce dei risultati approssimati, ma che si verificano con alta probabilità. Questi limiti superiori e inferiori forniscono un valido strumento nell'analisi degli algoritmi randomizzati.

In questo elaborato si estendono i risultati ottenuti in [1] ai casi $m \neq n$. La prima parte introduce il processo tramite il paradosso del compleanno, un problema didattico che mostra come le singole scatole possono avere occupazioni ben diverse dalla media; inoltre introduce un primo esempio di applicazione del processo sulla base delle considerazioni appena fatte. Nella seconda parte si elencano nel dettaglio le varie approssimazioni che permettono di caratterizzare il processo in modo sintetico: infatti, se alcune caratteristiche del processo sono facili da ottenere con precisione (ad esempio

la media, la varianza e la distribuzione di probabilità del carico della singola scatola), altre non sono trattabili con esattezza a causa delle dipendenze tra le scatole (come la distribuzione di probabilità del carico massimo). Nella terza parte si cita una variante che con una semplice modifica al processo permette di normalizzare sensibilmente il carico delle scatole. Nell'ultima parte si mostrano alcune applicazioni del processo in algoritmi randomizzati.

1.1 Paradosso del compleanno

Un esempio introduttivo al processo *Balls and Bins* consiste nell'analizzare il seguente problema. Si supponga che in una stanza siano presenti m persone, e che ciascuna persona compia gli anni in uno dei $n = 365$ giorni dell'anno (ignorando gli anni bisestili), con distribuzione casuale uniforme. Siamo interessati a conoscere la probabilità che tutte le persone compiano gli anni in giorni distinti. Questa probabilità si può ottenere sfruttando la probabilità condizionata: sia X_i una variabile aleatoria rappresentante il giorno di nascita dell'i-esima persona, e sia ξ_i l'evento in cui le prime i persone compiono gli anni in giorni distinti. Allora

$$P[\xi_m] = P[X_m \neq X_k, 1 \leq k \leq m-1 | \xi_{m-1}] \, P[\xi_{m-1}]$$

ovvero, la probabilità che vi siano m compleanni distinti è uguale alla probabilità che i primi $m-1$ compleanni siano distinti, moltiplicata per la probabilità che l'm-esimo compleanno sia diverso dai primi $m-1$ sapendo che i primi $m-1$ sono distinti.

La probabilità che l'm-esimo compleanno sia diverso dai primi $m-1$ compleanni distinti è $\left(1 - \frac{m-1}{n}\right)$. Per induzione si può ottenere:

$$P[\xi_m] = \prod_{i=0}^{m-1} \left(1 - \frac{i}{n}\right)$$

Per $m = 23$ persone, questa probabilità è circa $0.493 < \frac{1}{2}$, ovvero è più probabile che almeno due persone compiano gli anni nello stesso giorno.

Questa espressione, pur essendo precisa, non è sempre facile da utilizzare, in particolare per valori di m elevati. Si può ottenere una approssimazione sfruttando la serie di Taylor di e^x:

$$e^x = \sum_{k=0}^{\infty} \frac{x^k}{k!} \implies e^x \geq 1 + x \quad \forall x \in \mathbb{R}$$

Per k/n positivo $e^{-k/n} \geq 1 - \frac{k}{n}$, quindi:

$$\begin{aligned}\prod_{i=0}^{m-1}\left(1-\frac{i}{n}\right) &\leq \prod_{i=0}^{m-1} e^{-\frac{i}{n}} \\ &= \exp\left(-\sum_{i=0}^{m-1}\frac{i}{n}\right) \\ &= e^{-\frac{(m-1)m}{2n}}\end{aligned}$$

Per $m = 23$ l'approssimazione fornisce una probabilità di circa 0.5, arrotondato per eccesso. Tramite questa approssimazione, eventualmente con l'ulteriore semplificazione:

$$e^{-\frac{(m-1)m}{2n}} \simeq e^{-\frac{m^2}{2n}}$$

è possibile determinare facilmente i valori di m per cui $P[\xi_m] \leq p$, poiché $e^{-\frac{(m-1)m}{2n}}$ è un upper bound per $P[\xi_m]$. Ad esempio, perché la probabilità di avere compleanni distinti sia minore di p, utilizzando anche l'ultima semplificazione si ha

$$e^{-\frac{m^2}{2n}} \leq p \implies m \geq \sqrt{-2n\ln p}$$

Nel nostro caso, con $p = 0.5$ si ottiene $m \geq 22.49$. Quindi, già con 23 persone è probabile che due di queste condividano lo stesso compleanno. Analogamente, per far sì che la probabilità che tutti i compleanni siano distinti sia minore di $\frac{1}{100}$ (ovvero al 99% due o più persone condividano lo stesso compleanno), occorrono "solo" $m \geq \sqrt{2n\ln 100} \simeq 58$ persone.

Questo risultato mostra come sia sufficiente un numero sensibilmente più piccolo di n persone per ottenere più persone con lo stesso compleanno, mentre ci si aspetterebbe in prima valutazione che avere una sovrapposizione richieda $m \simeq n$. Tuttavia è importante notare che già con $m = n + 1$ per il *pigeonhole principle* deve esistere almeno una sovrapposizione.

I valori di m per i quali la probabilità di non avere collisioni sia limitata da una costante possono essere trovati in modo rigoroso. Per trovare il limite inferiore per p, si utilizzano i seguenti lemmi.

Lemma 1. *Per* $0 \leq x \leq \sqrt{2} - 1 \sim 0.41$ *si ha*

$$e^{-x-x^2} \leq 1 - x$$

Dimostrazione. Applicando lo sviluppo in serie di Taylor per $x \geq 0$

$$e^{-x-x^2} = 1 - x - x^2 + \frac{(x+x^2)^2}{2} - \cdots \leq 1 - x - x^2 + \frac{(x+x^2)^2}{2}$$

Affinché l'ultimo termine sia minore o uguale a $1 - x$ deve essere

$$-x^2 + \frac{(x+x^2)^2}{2} \leq 0$$

$$-2x^2 + x^4 + 2x^3 + x^2 \leq 0$$

$$x^2 + 2x - 1 \leq 0$$

La disuguaglianza è verificata per $0 \leq x \leq \sqrt{2} - 1$. □

Il lemma seguente si può dimostrare per induzione:

Lemma 2.

$$\sum_{k=1}^{n} k^2 = \frac{n(n+1)(2n+1)}{6}$$

Teorema 3. *Se* $m \leq (\sqrt{2} - 1)n$

$$e^{-\frac{5m^2-6m+1}{6n}} \leq p \leq e^{-\frac{m^2-m}{2n}}$$

Dimostrazione. Il limite superiore è già stato dimostrato in precedenza. Dimostriamo quindi il limite inferiore.

$$\begin{aligned}
p = \prod_{i=0}^{m-1} \left(1 - \frac{i}{n}\right) &\geq \prod_{i=0}^{m-1} \exp\left\{-\frac{i}{n} - \frac{i^2}{n^2}\right\} \\
&= \exp\left\{-\sum_{i=1}^{m-1} \frac{i}{n} - \sum_{i=1}^{m-1} \frac{i^2}{n^2}\right\} \\
&= \exp\left\{-\frac{m(m-1)}{2n} - \frac{(m-1)m(2m-1)}{6n^2}\right\} \\
&\geq \exp\left\{-\frac{m(m-1)}{2n} - \frac{(m-1)(2m-1)}{6n}\right\} \\
&= \exp\left\{-\frac{5m^2-6m+1}{6n}\right\}
\end{aligned}$$

□

Da questi risultati è possibile ottenere i valori di m per cui p è limitata da una costante.

Corollario 4. *Se* $m \leq (\sqrt{2}-1)n$ *e* $0 < c \leq 1$

$$m \leq \frac{3}{5} + \sqrt{\frac{4}{25} - \frac{6}{5} n \ln c} \implies p \geq c$$

Per n sufficientemente grande:

$$m \leq \sqrt{-n \ln c} \implies p \geq c$$

Corollario 5.

$$m \geq \frac{1}{2} + \sqrt{\frac{1}{4} - 2n \ln c} \implies p \leq c$$

1.2 Compleanni e Hashing

Si può trovare una applicazione concreta di questo problema nell'hashing. Si dà un insieme di m elementi e una funzione h che mappa ogni elemento x dell'insieme in una stringa binaria $h(x)$ di k bit. Inoltre, si suppone che la funzione h sia tale che ogni possibile valore della funzione possa essere generato con la stessa probabilità. Un evento indesiderato è la "collisione" di due elementi, ovvero due elementi x_1, x_2 tali che $h(x_1) = h(x_2)$.

In queste condizioni, la probabilità di una o più collisioni coincide con la probabilità che due o più tra m persone condividano il compleanno tra n possibili date.

Esempio Utilizzando la semplificazione $p \approx e^{-m^2/(2n)}$, una funzione di hash uniforme con output a $k = 128$ bit avrebbe il 50% di probabilità di collisione con $m = \sqrt{2 * 2^{128} * \ln 2} \simeq 2.172 \times 10^{19}$ elementi. Se invece $k = 64$, allora $m = 5.06 \times 10^9$.

Nonostante m sia sensibilmente inferiore rispetto ad n, il numero di possibili stringhe binarie di 128 bit, è comunque molto elevato e permette di utilizzare l'hashing in molte applicazioni, ad esempio per confrontare rapidamente se un file è già presente in un database. In presenza di una collisione, si può verificare se i file hanno lo stesso hash oppure assumere che il file sia già presente con alta probabilità.

La sezione 3.2 affronterà in modo più approfondito l'uso dell'hashing.

Capitolo 2

Balls and Bins

2.1 Descrizione del processo

Il processo *Balls and Bins* è una semplice generalizzazione dell'esempio precedente. Si lanciano m palline in modo casuale all'interno di n scatole. Nelle nostre analisi assumiamo che ciascuna pallina finisca all'interno di una scatola scelta in modo casuale, indipendente dagli altri lanci e in modo uniforme (ogni scatola ha la stessa probabilità di ricevere una determinata pallina). Alla fine del processo, siamo interessati a conoscere come sono distribuite le palline nelle scatole, ad esempio determinando la probabilità che le palline si distribuiscano secondo certi criteri.

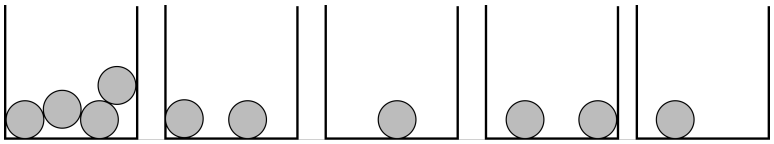

Figura 2.1: Esempio di distribuzione con $m = 10$, $n = 5$

Il processo *Balls and Bins*, insieme alle sue varianti (non necessariamente con distribuzione uniforme delle singole palline nelle scatole), trova impiego in varie situazioni reali, dalla teoria dei giochi alla meccanica statistica. In [3] è presente una lista di situazioni riconducibili al processo *Balls and Bins*.

2.2 Analisi del processo

In questa sezione sono esposti i risultati di una analisi teorica del processo. A tale scopo saranno introdotti degli elementi di teoria che serviranno per

ottenere ulteriori risultati.

2.2.1 Distribuzione di probabilità della singola scatola

La distribuzione di probabilità del numero di palline in una determinata scatola è semplice da ottenere. Si farà spesso riferimento alla prima scatola, dato che per simmetria la distribuzione di probabilità è la stessa per ogni scatola (pur non essendo indipendenti tra loro).

Lemma 6. *Sia X il numero di palline nella prima scatola. X è una variabile aleatoria binomiale di parametri $(m, 1/n)$. La probabilità che la prima scatola abbia k palline è*

$$p_X(k) = P[X = k] = \binom{m}{k} p^k (1-p)^{m-k}, \quad 0 \leq k \leq m$$

Dimostrazione. Definite m variabili aleatorie indicatrici X_i, $i = 1, 2, \ldots, m$ tali che:

$$X_i = \begin{cases} 1 & \text{se la } i\text{-esima pallina è finita nella prima scatola} \\ 0 & \text{altrimenti} \end{cases}$$

Per definizione, $X = \sum_{i=1}^{m} X_i$. Le singole variabili indicatrici X_i sono variabili Bernoulliane indipendenti tra di loro (il risultato del lancio di una pallina non influisce sul risultato delle altre per ipotesi). La probabilità che la variabile assuma il valore 1 è uguale alla probabilità che l'i-esima pallina finisca nella prima tra le n scatole:

$$p = P(X_i = 1) = \frac{1}{n}$$

È noto che la somma di m variabili Bernoulliane indipendenti di parametro p corrisponde ad una variabile aleatoria binomiale di parametri (m, p). □

Si può raggiungere questo risultato anche senza l'utilizzo delle variabili indicatrici: perché vi siano esattamente k palline nella prima scatola, le restanti $m - k$ dovranno finire nelle altre scatole. Fissato l'insieme dei k lanci che finiscono nella prima scatola, i casi favorevoli sono $(n-1)^{m-k}$, tutti equiprobabili. Le possibili scelte di questi insiemi sono $\binom{m}{k}$. In rapporto con il numero totale di casi, si ottiene:

$$p_X(k) = \binom{m}{k} \frac{(n-1)^{m-k}}{n^m} = \binom{m}{k} \frac{1}{n^k} \left(\frac{n-1}{n}\right)^{m-k}$$

e con $p = \frac{1}{n}$ si ottiene lo stesso risultato.

Media Sfruttando la linearità dell'aspettazione, la media di una variabile aleatoria binomiale è:

$$E[X] = \sum_{i=1}^{m} E[X_i] = mp = \frac{m}{n}$$

Questo risultato è appoggiato dal fatto che in ogni istante la media fra le n scatole del numero di palline in una scatola è proprio $\frac{m}{n}$.

Varianza La varianza di una variabile aleatoria binomiale è $np(1-p)$. Il metodo più semplice per ottenere la varianza è di sfruttare le proprietà dell'aspettazione e le variabili indicatrici. Innanzitutto:

$$E\left[(X-\mu)^2\right] = E\left[\left(\sum_{i=1}^{m}(X_i - \mu_i)\right)^2\right]$$

Dove $\mu = E[X]$ e $\mu_i = E[X_i]$. Sviluppando il quadrato nei singoli termini della sommatoria si ha

$$E\left[(X-\mu)^2\right] = E\left[\sum_{i=1}^{m}(X_i-\mu_i)^2\right] + E\left[\sum_{\substack{i=1,\dots,m \\ j=1,\dots,m \\ i\neq j}} (X_i-\mu_i)(X_j-\mu_j)\right]$$

Il primo membro della somma è la somma delle varianze delle singole variabili indicatrici, che per ciascuna variabile è $p(1-p)$. Il secondo termine corrisponde alla covarianza tra X_i e X_j con $i \neq j$. Poiché ciascuna coppia è indipendente, la loro covarianza è nulla. Quindi:

$$E[(X-\mu)^2] = mp(1-p)$$

2.2.2 Approssimazioni di Poisson

Un ostacolo rilevante per l'analisi del processo è la dipendenza fra i carichi delle scatole. Ad esempio, il carico medio della seconda scatola sapendo che la prima è vuota è $m/(n-1)$ anziché m/n. Questo rende più difficile trovare una espressione semplice della distribuzione di probabilità di certi parametri, ad esempio il carico massimo.

È possibile tuttavia ottenere delle approssimazioni, così come è stato fatto per calcolare la probabilità di non avere collisioni nella sezione 1.1. Queste

approssimazioni permettono di considerare ogni scatola come indipendente dalle altre e con una espressione ridotta per la distribuzione di probabilità.

Riprendendo il Lemma 6, la probabilità che una determinata scatola abbia k palline è

$$\begin{aligned} p_X(k) &= \binom{m}{k} p^k (1-p)^{m-k} = \frac{m!}{k!(m-k)!} \frac{1}{n^k} \left(1 - \frac{1}{n}\right)^{m-k} \\ &= \frac{1}{k!} \frac{m(m-1)\cdots(m-k+1)}{n^k} \left(1 - \frac{1}{n}\right)^{m-k} \end{aligned}$$

Se m ed n sono molto maggiori di k, è possibile approssimare il secondo termine come $(m/n)^k$ e il terzo come $e^{-m/n}$, ottenendo

$$p_X(k) \approx \frac{(m/n)^k e^{-m/n}}{k!}$$

Questa approssimazione coincide con la distribuzione di probabilità di una variabile aleatoria di Poisson di parametro $\lambda = m/n$:

$$p_X(k) = \frac{e^{-\lambda} \lambda^k}{k!}$$

Le variabili aleatorie di Poisson godono di importanti proprietà, in particolare faremo uso della proprietà della somma:

Lemma 7 (Somma di variabili aleatorie di Poisson)**.** *La somma di un numero finito di variabili aleatorie di Poisson indipendenti con parametri λ_i è una variabile aleatoria di Poisson con parametro $\lambda = \sum_i \lambda_i$*

Inoltre, faremo uso del seguente upper bound per il fattoriale.

Lemma 8.

$$m! \leq e\sqrt{m} \left(\frac{m}{e}\right)^m$$

Dimostrazione. Applicando il logaritmo al fattoriale:

$$\ln(m!) = \sum_{i=1}^{m} \ln i$$

Innanzitutto, poiché $\ln(x)$ è una funzione concava, per $i \geq 2$

$$\int_{i-1}^{i} \ln x \, dx \geq \frac{\ln(i-1) + \ln i}{2}$$

Quindi

$$\int_1^n \ln x\, dx \geq \sum_{i=2}^{m} \frac{\ln(i-1) + \ln i}{2} = \sum_{i=1}^{m} \ln i - \frac{\ln n}{2}$$

In quanto $\ln 1 = 0$. La primitiva di $\ln x$ è $x \ln x - x + c$, quindi

$$m \ln m - m + 1 \geq \ln(m!) - \frac{\ln m}{2}$$

Applicando l'esponenziale

$$m! \leq \frac{m^m}{e^m} e\sqrt{m}$$

□

Approssimare la distribuzione delle palline nelle scatole tramite variabili aleatorie di Poisson indipendenti permetterebbe di semplificare ulteriormente l'analisi, perdendo però il vincolo di avere esattamente m palline nelle n scatole. Nonostante ciò è possibile mostrare che la differenza tra i due processi è limitata.

Nell'ipotesi in cui k palline siano lanciate in n scatole, si definisce $X_i^{(k)}$ come il numero di palline nella i-esima scatola e $Y_i^{(m)}$ una possibile approssimazione, ovvero una variabile di Poisson con media $\mu = m/n$, con m positivo ma non necessariamente uguale a k. Il prossimo teorema afferma che le $Y_i^{(m)}$ nella condizione in cui la loro somma sia k coincidono in distribuzione con il processo Balls and Bins esatto, anche con $m \neq k$

Teorema 9. *La distribuzione del vettore aleatorio* $\left(Y_1^{(m)}, \ldots, Y_n^{(m)}\right)$ *condizionata da* $\sum_i Y_i^{(m)} = k$ *coincide con la distribuzione di* $\left(X_1^{(k)}, \ldots, X_n^{(k)}\right)$ *indipendentemente dal valore di* m.

Dimostrazione. Calcoliamo la probabilità che $\left(X_1^{(k)}, \ldots, X_n^{(k)}\right) = (k_1, \ldots, k_n)$ con $\sum_i k_i = k$. Le possibili partizioni distinte delle k palline nelle n scatole di destinazione che soddisfano la condizione sono

$$\binom{k}{k_1}\binom{k-k_1}{k_2}\cdots\binom{k-k_1-\cdots-k_{n-2}}{k_{n-1}} = \frac{k!}{(k_1!)\cdots(k_n!)}$$

Quindi, la probabilità è

$$p_X(k_1, \ldots, k_n) = \frac{1}{n^k} \frac{k!}{(k_1!)\cdots(k_n!)}$$

Calcoliamo infine la probabilità che $(Y_1^{(m)}, \ldots, Y_n^{(m)}) = (k_1, \ldots, k_n)$ con la condizione $\sum_i Y_i^{(m)} = k$ (implicitamente soddisfatta per qualunque vettore $(k_1, \ldots, k_n)$ tale che $\sum_i k_i = k$):

$$\begin{aligned} p_{Y|\sum_i Y_i^{(m)}=k}(k_1, \ldots, k_n) &= \frac{P\left[Y = (k_1, \ldots, k_n) \wedge \sum_i Y_i^{(m)} = k\right]}{P\left[\sum_i Y_i^{(m)} = k\right]} \\ &= \frac{P\left[Y = (k_1, \ldots, k_n)\right]}{P\left[\sum_i Y_i^{(m)} = k\right]} \\ &= \frac{P\left[Y_1^{(m)} = k_1\right] \cdots P\left[Y_n^{(m)} = k_m\right]}{P\left[\sum_i Y_i^{(m)} = k\right]} \end{aligned}$$

Uilizzando il Lemma 7, $P\left[\sum_i Y_i^{(m)} = k\right] = \frac{e^{-m}m^k}{k!}$ e

$$\begin{aligned} p_{Y|\sum_i Y_i^{(m)}=k}(k_1, \ldots, k_n) &= \left[\frac{e^{-m/n}(m/n)^{k_1}}{k_1!} \cdots \frac{e^{-m/n}(m/n)^{k_n}}{k_n!}\right] \frac{k!}{e^{-m}m^k} \\ &= \frac{1}{n^k} \frac{k!}{(k_1!) \cdots (k_n!)} \end{aligned}$$

□

Questa relazione tra le distribuzioni permette di affermare che le variabili $\left(X_1^{(k)}, \ldots, X_n^{(k)}\right)$ sono un "caso particolare" delle variabili di Poisson indipendenti. Questa considerazione porta al seguente teorema:

Teorema 10. *Sia $f(x_1, x_2, \ldots, x_n)$ una funzione non negativa. Allora*

$$E\left[f\left(X_1^{(m)}, \ldots, X_n^{(m)}\right)\right] \leq e\sqrt{m}\, E\left[f\left(Y_1^{(m)}, \ldots, Y_n^{(m)}\right)\right]$$

Dimostrazione. Sfruttiamo il teorema dell'aspettazione totale, e limitiamo

inferiormente il risultato rispetto al sottocaso $\sum_i Y_i^{(m)} = m$:

$$\begin{aligned}
E\left[f\left(Y_1^{(m)},\ldots,Y_n^{(m)}\right)\right] &= E\left[E\left[f\left(Y_1^{(m)},\ldots,Y_n^{(m)}\right)\,\middle|\,\sum_i Y_i^{(m)}\right]\right] \\
&= \sum_{k=0}^{\infty} E\left[f\left(Y_1^{(m)},\ldots,Y_n^{(m)}\right)\,\middle|\,\sum_i Y_i^{(m)} = k\right] P\left[\sum_i Y_i^{(m)} = k\right] \\
&\geq E\left[f\left(Y_1^{(m)},\ldots,Y_n^{(m)}\right)\,\middle|\,\sum_i Y_i^{(m)} = m\right] P\left[\sum_i Y_i^{(m)} = m\right] \\
&= E\left[f\left(X_1^{(m)},\ldots,X_n^{(m)}\right)\right] P\left[\sum_i Y_i^{(m)} = m\right] \\
&= E\left[f\left(X_1^{(m)},\ldots,X_n^{(m)}\right)\right] \frac{e^{-m}m^m}{m!}
\end{aligned}$$

Usiamo ora il Lemma 8:

$$m! \leq e\sqrt{m}\left(\frac{m}{e}\right)^m$$

Quindi

$$\begin{aligned}
E\left[f\left(Y_1^{(m)},\ldots,Y_n^{(m)}\right)\right] &\geq E\left[f\left(X_1^{(m)},\ldots,X_n^{(m)}\right)\right] \frac{e^{-m}m^m}{m!} \\
&\geq E\left[f\left(X_1^{(m)},\ldots,X_n^{(m)}\right)\right] \frac{1}{e\sqrt{m}}
\end{aligned}$$

□

Questo teorema è valido per qualunque f non negativa, e quindi anche per una funzione che vale 1 al verificarsi di un evento in funzione del vettore di occupazione e 0 altrimenti. L'aspettazione di questa funzione non è altro che la probabilità dell'evento stesso, dunque:

Corollario 11. *Un evento che si verifica con probabilità p nel caso di variabili aleatorie di Poisson si verifica con probabilità al più $p\,e\sqrt{m}$ nel processo Balls and Bins esatto.*

Dimostrazione. Sia $\xi = \xi(x_1, x_2, \ldots, x_n)$ la funzione indicatrice dell'evento, ovvero

$$\xi(x_1,\ldots,x_n) = \begin{cases} 1 & \text{se l'evento si verifica} \\ 0 & \text{altrimenti} \end{cases}$$

Allora la probabilità che l'evento si verifichi nel processo esatto è

$$E\left[\xi\left(X_1^{(m)},\ldots,X_n^{(m)}\right)\right] = P\left[\xi\left(X_1^{(m)},\ldots,X_n^{(m)}\right)=1\right] = p'$$

mentre la probabilità che l'evento si verifichi nell'approssimazione di Poisson è

$$E\left[\xi\left(Y_1^{(m)},\ldots,Y_n^{(m)}\right)\right] = P\left[\xi\left(Y_1^{(m)},\ldots,Y_n^{(m)}\right)=1\right] = p$$

Applicando infine il Teorema 10

$$p' \leq e\sqrt{m}\,p$$

□

In alcuni casi, si può raggiungere un limite ancora più preciso:

Teorema 12. *Sia $f(x_1,\ldots,x_n)$ una funzione non negativa tale che la sua aspettazione $E[f(X_1^{(m)},\ldots,X_n^{(m)})]$ è monotona crescente oppure decrescente in m. Allora*

$$E\left[f\left(X_1^{(m)},\ldots,X_n^{(m)}\right)\right] \leq 2\,E\left[f\left(Y_1^{(m)},\ldots,Y_n^{(m)}\right)\right] \tag{2.1}$$

Anche in questo caso si può utilizzare il risultato per la probabilità di un evento:

Corollario 13. *Sia ξ un evento la cui probabilità nel caso esatto è monotona crescente o monotona decrescente nel numero di palline. Se ξ si verifica con probabilità p nel caso di variabili aleatorie di Poisson si verifica con probabilità al più $2p$ nel processo Balls and Bins esatto.*

La dimostrazione di questo teorema (qui dimostrata per funzioni monotone crescenti) richiede un passaggio intermedio:

Lemma 14. *Sia Z una v.a. di Poisson di parametro $\mu \geq 1$ intero. Allora*

a) Per $0 \leq h \leq \mu - 1$ intero,

$$P[Z = \mu + h] \geq P[Z = \mu - h - 1]$$

b)

$$P[Z \geq \mu] \geq \frac{1}{2}$$

Dimostrazione. a) Applichiamo la distribuzione di Poisson sulla disuguaglianza

$$\frac{e^{-\mu}\mu^{\mu+h}}{(\mu+h)!} \geq \frac{e^{-\mu}\mu^{\mu-h-1}}{(\mu-h-1)!}$$

$$\mu^{2h+1} \geq \frac{(\mu+h)!}{(\mu-h-1)!} = (\mu+h)\cdots(\mu-h)$$

Applicando il logaritmo:

$$(2h+1)\ln\mu \geq \sum_{i=\mu-h}^{\mu+h} \ln i$$

Il logaritmo è una funzione concava, infatti la sua derivata seconda è $-1/x^2$, che è negativa per $x > 0$. Per le funzioni concave vale il seguente lemma:

Lemma 15. *Sia $f(x)$ una funzione concava, continua e derivabile in un intervallo reale I. Allora $\forall x_1, x_2 \in I$*

$$f(x_1) + (x_2 - x1)f'(x_1) \geq f(x_2)$$

Quindi, per $\mu > 0, k > 0$

$$\ln(\mu+k) \leq \ln\mu + (\mu+k-\mu)\frac{1}{\mu} = \ln\mu + k/\mu$$

Si ottiene infine la tesi:

$$\sum_{i=\mu-h}^{\mu+h} \ln i \leq (2h+1)\ln\mu + \sum_{k=-h}^{h} \frac{k}{\mu} = (2h+1)\ln\mu$$

b) Poiché $P[Z \geq \mu]$ include i casi precedenti

$$P[Z \geq \mu] = \sum_{h=0}^{\infty} P[Z = \mu+h] \geq \sum_{h=0}^{\mu-1} P[Z = \mu+h]$$
$$\geq \sum_{h=0}^{\mu-1} P[Z = \mu-h-1] = 1 - P[Z \geq \mu]$$

$$P[Z \geq \mu] \geq 1 - P[Z \geq \mu] \implies P[Z \geq \mu] \geq \frac{1}{2}$$

□

Dimostrazione del Teorema 12 per funzioni monotone crescenti.

$$\begin{aligned}
E\left[f\left(Y_1^{(m)},\ldots,Y_n^{(m)}\right)\right] &= \sum_{k=0}^{\infty}\left\{E\left[f\left(Y_1^{(m)},\ldots,Y_n^{(m)}\right) \mid \sum_i Y_i^{(m)} = k\right]\right. \\
&\qquad \left.\times P\left[\sum_i Y_i^{(m)} = k\right]\right\} \\
&\geq \sum_{k=m}^{\infty}\left\{E\left[f\left(Y_1^{(m)},\ldots,Y_n^{(m)}\right) \mid \sum_i Y_i^{(m)} = k\right]\right. \\
&\qquad \left.\times P\left[\sum_i Y_i^{(m)} = k\right]\right\} \\
&= \sum_{k=m}^{\infty} E\left[f\left(X_1^{(k)},\ldots,X_n^{(k)}\right)\right] P\left[\sum_i Y_i^{(m)} = k\right]
\end{aligned}$$

Poiché $E\left[f\left(X_1^{(k)},\ldots,X_n^{(k)}\right)\right]$ è monotona crescente in k, si ottiene

$$\begin{aligned}
E\left[f\left(Y_1^{(m)},\ldots,Y_n^{(m)}\right)\right] &\geq E\left[f\left(X_1^{(m)},\ldots,X_n^{(m)}\right)\right] \sum_{k=m}^{\infty} P\left[\sum_i Y_i^{(m)} = k\right] \\
&= E\left[f\left(X_1^{(m)},\ldots,X_n^{(m)}\right)\right] P\left[\sum_i Y_i^{(m)} \geq m\right] \\
&\geq \frac{1}{2} E\left[f\left(X_1^{(m)},\ldots,X_n^{(m)}\right)\right]
\end{aligned}$$

□

2.2.3 Disuguaglianze di Chernoff

Un altro strumento importante per l'analisi del processo sono le disuguaglianze di Chernoff. Le disuguaglianze di Chernoff sono ottenute dalla disuguaglianza di Markov applicata alla funzione generatrice dei momenti:

Definizione 1. *Sia X una variabile aleatoria. La sua funzione generatrice dei momenti è definita come*

$$M_X(t) = E[e^{tX}]$$

Poiché la funzione generatrice dei momenti è una funzione non negativa, l'applicazione della disuguaglianza di Markov è immediata.

Teorema 16 (Disuguaglianze di Chernoff)**.** *Sia X una variabile aleatoria; per $t > 0$:*

$$P(X \geq a) = P\left(e^{tX} \geq e^{ta}\right) \leq \frac{E\left[e^{tX}\right]}{e^{ta}}$$

per $t < 0$:

$$P(X \leq a) = P\left(e^{tX} \geq e^{ta}\right) \leq \frac{E\left[e^{tX}\right]}{e^{ta}}$$

Il valore di t può essere scelto in modo da minimizzare $E[e^{tX}]/e^{ta}$, oppure in modo da ottenere un'espressione conveniente da utilizzare.

È possibile sfruttare questo risultato sia nelle distribuzioni binomiali che in quelle di Poisson. Per il calcolo della funzione generatrice dei momenti si può utilizzare la seguente proprietà:

Teorema 17. *Se X ed Y sono variabili aleatorie indipendenti, allora*

$$M_{X+Y}(t) = E\left[e^{tX+tY}\right] = E\left[e^{tX}e^{tY}\right] = E\left[e^{tX}\right]E\left[e^{tY}\right] = M_X(t)M_Y(t)$$

Di conseguenza, ricordando che una variabile aleatoria binomiale è la somma di n variabili aleatorie Bernoulliane indipendenti di parametro p, calcolando la funzione generatrice dei momenti delle singole v.a. Bernoulliane è possibile ricavare la funzione generatrice dei momenti per una variabile aleatoria binomiale.

Teorema 18. *Sia X_1 una v.a. Bernoulliana di parametro p. Allora*

$$E\left[e^{tX_1}\right] \leq e^{p(e^t-1)}$$

Dimostrazione.

$$E\left[e^{tX_1}\right] = \sum_{i=0,1} e^{ti}P(X_1 = i) = pe^t + (1-p) = 1 + p(e^t - 1) \leq e^{p(e^t-1)}$$

Dove nell'ultima disuguaglianza è stata utilizzata la serie di Taylor di e^x. □

Teorema 19. *Sia X una v.a. Binomiale di parametri (n, p). Allora*

$$M_X(t) \leq e^{np(e^t-1)}$$

Dimostrazione. Essendo $X = \sum_{i=1}^{n} X_i$, con X_i v.a. Bernoulliane di parametro p, allora

$$M_X(t) = \prod_{i=1}^{n} M_{X_i}(t) \leq \prod_{i=1}^{n} e^{p(e^t-1)} = e^{np(e^t-1)}$$

□

Il risultato è inoltre generalizzabile per somme di v.a. Bernoulliane indipendenti con probabilità p_i distinte (dette *prove di Poisson*), con $\mu = \sum_{i=1}^{n} p_i$ al posto di np.

Per le distribuzioni di Poisson, la funzione generatrice dei momenti coincide con il limite superiore per le v.a. binomiali:

Teorema 20. *La funzione generatrice dei momenti di una v.a. di Poisson Y di parametro μ è*

$$M_Y(t) = e^{\mu(e^t-1)}$$

Dimostrazione.

$$M_Y(t) = E\left[e^{tY}\right] = \sum_{k=0}^{\infty} \frac{e^{-\mu}(\mu e^t)^k}{k!} = e^{\mu(e^t-1)} \sum_{k=0}^{\infty} \frac{e^{-\mu e^t}(\mu e^t)^k}{k!} = e^{\mu(e^t-1)}$$

□

L'applicazione delle disuguaglianze di Chernoff su questi risultati porta al seguente teorema:

Teorema 21. *Sia Y una v.a. Binomiale, e $\mu = np$. Allora*

1. *Per $\delta > 0$*
$$P(X \geq (1+\delta)\mu) \leq \left(\frac{e^{\delta}}{(1+\delta)^{(1+\delta)}}\right)^{\mu}$$

2. *Per $R \geq 6\mu$*
$$P(X \geq R) \leq 2^{-R}$$

Dimostrazione.

$$P(X \geq (1+\delta)\mu) \leq \frac{M_X(t)}{e^{t(1+\delta)\mu}} \leq \frac{e^{\mu(e^t-1)}}{e^{t(1+\delta)\mu}}$$

Per $\delta > 0$ possiamo porre $t = \ln(1+\delta) > 0$:

$$P(X \geq (1+\delta)\mu) \leq \frac{e^{\mu\delta}}{(1+\delta)^{(1+\delta)\mu}} = \left(\frac{e^{\delta}}{(1+\delta)^{(1+\delta)}}\right)^{\mu}$$

Per $R \geq 6\mu$ si ha $\delta \geq 5$ e

$$\begin{aligned} P(X \geq (1+\delta)\mu) &\leq \left(\frac{e^{\delta}}{(1+\delta)^{(1+\delta)}}\right)^{\mu} \\ &\leq \left(\frac{e}{1+\delta}\right)^{(1+\delta)\mu} \\ &\leq \left(\frac{e}{6}\right)^{R} \\ &\leq \frac{1}{2^R} \end{aligned}$$

□

2.2.4 Carico massimo

I risultati ottenuti fino a qui ci permettono di formulare una stima per il carico massimo fra tutte le scatole. Ricavare la probabilità che il carico massimo sia in un certo intervallo risulta problematico nel caso esatto a causa delle dipendenze tra i carichi delle scatole.

Per la singola scatola, la probabilità che essa riceva almeno k palline è

$$P(X_1 \geq k) = \sum_{i=k}^{m} \binom{m}{i} p^i (1-p)^{m-i}$$

È possibile utilizzare una approssimazione per eccesso per questa probabilità sfruttando il limite per l'unione:

Lemma 22 (Disuguaglianza di Boole o limite per l'unione)**.** *Dato un insieme finito di eventi* $A_1, \ldots, A_n$

$$P\left(\bigcup_{i=1}^{n} A_i\right) \leq \sum_{i=1}^{n} P\left(A_i\right)$$

Sapendo che almeno k palline sono finite nella prima scatola, allora esiste almeno un sottoinsieme di k palline finite nella prima scatola. Il numero di possibili sottoinsiemi è $\binom{m}{k}$, mentre la probabilità che un determinato insieme finisca nella prima scatola è $(1/n)^k$. Quindi, usando il limite per l'unione:

$$P(X_1 \geq k) \leq \binom{m}{k} \left(\frac{1}{n}\right)^k$$

Utilizzando un'altra volta il limite dell'unione si ottiene un limite superiore alla probabilità di avere un carico massimo di almeno k:

$$P(\max_i X_i \geq k) \leq n\binom{m}{k}\left(\frac{1}{n}\right)^k$$

Da questa approssimazione è possibile trovare un limite superiore al carico massimo con alta probabilità nel caso in cui $m = n$:

Teorema 23. *Quando n palline sono lanciate indipendentemente fra loro e in modo casuale uniforme in n scatole, la probabilità che il carico massimo sia maggiore di* $b \ln n / \ln \ln n$ *per* $b \geq e$ *è al più* $1/n^{b(1-1/e)-1}$

Dimostrazione. Nel caso in cui $m = n$, si ha:

$$P(X_1 \geq k) \leq \binom{n}{k}\left(\frac{1}{n}\right)^k = \frac{n!}{k!(n-k)!n^k} = \frac{n(n-1)\cdots(n-k+1)}{k!n^k} \leq \frac{1}{k!}$$

Dalla serie di Taylor dell'esponenziale otteniamo un ulteriore limite inferiore per il fattoriale:

$$e^k = \sum_{i=0}^{\infty} \frac{k^i}{i!} \geq \frac{k^k}{k!} \implies k! \geq \left(\frac{k}{e}\right)^k \implies \frac{1}{k!} \leq \left(\frac{e}{k}\right)^k$$

Di conseguenza

$$P\left(\max_i X_i \geq k\right) \leq n\left(\frac{e}{k}\right)^k$$

Con $k = b \ln n / \ln \ln n$, $b \geq e$, si ottiene:

$$\begin{aligned}
P\left(\max_i X_i \geq k\right) &\leq n\left(\frac{e \ln \ln n}{b \ln n}\right)^{b \ln n / \ln \ln n} \\
&\leq n\left(\frac{\ln \ln n}{\ln n}\right)^{b \ln n / \ln \ln n} \\
&= \exp\left\{\ln n + \frac{b \ln n}{\ln \ln n} \ln\left(\frac{\ln \ln n}{\ln n}\right)\right\} \\
&= \exp\left\{\ln n + \frac{b \ln n}{\ln \ln n} (\ln \ln \ln n - \ln \ln n)\right\} \\
&= \exp\left\{(1-b)\ln n + b(\ln n)\frac{\ln \ln \ln n}{\ln \ln n}\right\}
\end{aligned}$$

Analizzando $\ln\ln\ln n/\ln\ln n$, si può osservare che ha massimo in e^{e^e}

$$\frac{d}{dx}\frac{\ln\ln\ln n}{\ln\ln n} = \frac{\frac{\ln\ln n}{n(\ln n)(\ln\ln n)} - \frac{\ln\ln\ln n}{n\ln n}}{[\ln\ln n]^2} = \frac{1-\ln\ln\ln n}{[\ln\ln n]^2 n\ln n}$$

$$\frac{d}{dx}\frac{\ln\ln\ln n}{\ln\ln n} = 0 \implies \ln\ln\ln n = 1$$

In tale punto, $\ln\ln\ln n/\ln\ln n = 1/e$, quindi

$$P\left(\max_i X_i \geq \frac{b\ln n}{\ln\ln n}\right) \leq \exp\left\{(1-b)\ln n + \frac{b}{e}\ln n\right\} = \frac{1}{n^{b(1-1/e)-1}}$$

□

Inoltre per n sufficientemente grande $\ln\ln\ln n/\ln\ln n \leq 1/3$, e per $b = 3$ si ottiene

$$P\left(\max_i X_i \geq \frac{3\ln n}{\ln\ln n}\right) \leq \frac{1}{n}$$

Possiamo inoltre utilizzare le approssimazioni di Poisson per ottenere un limite inferiore al carico massimo:

Teorema 24. *Quando n palline sono lanciate indipendentemente fra loro e in modo casuale uniforme in n scatole, la probabilità che il carico massimo sia minore di $\ln n/\ln\ln n$ è al più $1/n$ per n sufficientemente grande.*

Dimostrazione. Nell'approssimazione di Poisson la probabilità che la prima scatola abbia almeno k palline è:

$$P(Y_1 \geq k) = \sum_{i=k}^{\infty} \frac{e^{-m/n}(m/n)^i}{i!}$$

Per $m = n$ si ha:

$$P(Y_1 \geq k) = \sum_{i=k}^{\infty} \frac{1}{ei!} \geq \frac{1}{ek!}$$

Poiché ogni scatola è indipendente dalle altre nell'approssimazione di Poisson, la probabilità che nessuna scatola abbia k o più palline è

$$P(\max_i Y_i < k) \leq \left(1 - \frac{1}{ek!}\right)^n \leq e^{-n/(ek!)}$$

Nel caso esatto la disuguaglianza diventa

$$P(\max_i X_i < k) \leq e\sqrt{n}e^{-n/(ek!)}$$

Per ottenere una probabilità minore di $1/n$ possiamo cercare i valori di k tali che $e^{-n/(ek!)} \leq 1/n^2$, ovvero tali che $k! \leq n/2e \ln n$. Usando i logaritmi questa condizione diventa

$$\ln k! \leq \ln n - \ln\ln n - \ln(2e)$$

Riutilizziamo il Lemma 8

$$k! \leq e\sqrt{k}\left(\frac{k}{e}\right)^k \leq k\left(\frac{k}{e}\right)^k$$

per $k \geq e^2$. Posto $k = \ln n/\ln\ln n$, si ha

$$k! \leq \frac{\ln n}{\ln\ln n}\left(\frac{\ln n}{e\ln\ln n}\right)^{\frac{\ln n}{\ln\ln n}}$$

Usando il logaritmo:

$$\begin{aligned}
\ln k! &\leq \ln\ln n - \ln\ln\ln n + \left(\frac{\ln n}{\ln\ln n}\right)[\ln\ln n - 1 - \ln\ln\ln n] \\
&= \ln\ln n - \ln\ln\ln n + \ln n - \frac{\ln n}{\ln\ln n} - \frac{\ln n}{\ln\ln n}\ln\ln\ln n \\
&= \ln n - \frac{\ln n}{\ln\ln n} + \ln\ln n - \ln\ln\ln n(1 + \frac{\ln n}{\ln\ln n}) \\
&\leq \ln n - \frac{\ln n}{\ln\ln n} \\
&\leq \ln n - \ln\ln n - \ln(2e)
\end{aligned}$$

Dove nelle ultime due disequazioni è stato sfruttato il fatto che $\ln\ln n = o(\ln n/\ln\ln n)$. □

Sia per il Teorema 23 che per il Teorema 24 è possibile estendere facilmente i risultati in modo da ridurre la probabilità a $1/(2n)$. Nel primo caso, è sufficiente scegliere b in modo che $b(1-1/e)-1 > 1$, e per n sufficientemente grande è possibile ottenere il nuovo limite. Ad esempio, per $b = 3e/(e-1) \simeq 4.75$ si ottiene

$$P\left(\max_i X_i \geq \frac{3e\ln n}{(e-1)\ln\ln n}\right) \leq \frac{1}{n^2} \leq \frac{1}{2n} \quad \text{per } n \geq 2$$

Nel secondo caso si osservi che il teorema ottiene i valori di k tali che

$$P(\max_i X_i < k) \leq e\sqrt{n}\frac{1}{n^2} = \frac{e}{n^{3/2}} \leq \frac{1}{2n} \quad \text{per } n \geq 4e^2$$

In conclusione, unendo questi due risultati si ottiene il seguente teorema.

Teorema 25. *Quando n palline sono lanciate indipendentemente fra loro e in modo casuale uniforme in n scatole, per n sufficientemente grande*

$$P\left(\frac{\ln n}{\ln\ln n} \leq \max_i X_i \leq \frac{3e}{e-1}\frac{\ln n}{\ln\ln n}\right) \geq 1 - \frac{1}{n}$$

Carico massimo tramite le disuguaglianze di Chernoff Sfruttando le disuguaglianze di Chernoff sulle singole scatole possiamo individuare degli intervalli in cui il carico massimo risiede con elevata probabilità per $m > n$.

Teorema 26. *Se $m < (n\log_2 n)/3$, allora*

$$P\left(\max_i X_i \geq 2\log_2 n\right) \leq \frac{1}{n}$$

Se $m \geq (n\log_2 n)/3$, allora

$$P\left(\max_i X_i \geq 6\frac{m}{n}\right) \leq \frac{1}{n}$$

Dimostrazione. Per il limite dell'unione, la probabilità che il carico massimo sia superiore a k è $P(\max X_i > k) \leq nP(X_1 > k)$. Utilizzando il Teorema 21:

$$P\left(\max X_i > k\right) \leq \frac{n}{2^k} \quad \text{per } k \geq \frac{6m}{n}$$

Con $k = 6m/n$, e ponendo $P(\max_i X_i > k) \leq 1/n$

$$\frac{n}{2^{6m/n}} \leq \frac{1}{n}$$
$$2^{6m/n} \geq n^2$$
$$m \geq \frac{1}{3}n\log_2 n$$

Per $m \leq (n\log_2 n)/3$ e $k = 2\log_2 n$ il Teorema 21 è ancora applicabile e si ottiene:

$$P\left(\max_i X_i > 2\log_2 n\right) \leq \frac{n}{n^2} = \frac{1}{n}$$

□

Questo significa che, per n sufficientemente grande, lanciando $m \geq (n \log_2 n)/3$ palline il carico massimo è con alta probabilità vicino al carico medio delle scatole, mentre per un numero inferiore di palline il carico massimo non supera comunque $2 \log_2 n$. Limiti più ristretti possono essere ottenuti usando la prima equazione del Teorema 21. Per $m \leq n$ valgono le considerazioni sul carico massimo per $m = n$, per cui il carico massimo è $O(\ln n / \ln \ln n)$. Con $m = n^\epsilon$, $0 < \epsilon < 1$ è possibile limitare il carico massimo ad una funzione di ϵ:

Teorema 27. *Per $m = n^\epsilon$ con $0 < \epsilon < 1$ e con $k \geq 2/(1-\epsilon) \wedge k \geq e$*

$$P\left(\max_i X_i \geq k\right) \leq \frac{1}{n}$$

Dimostrazione. Sia $\delta = kn/n^\epsilon - 1$. Ponendo $r = n/n^\epsilon$

$$P\left(\max_i X_i \geq (1+\delta)\frac{m}{n}\right) = P\left(\max_i X_i \geq k\right) \leq n\left(\frac{e^{kr-1}}{(kr)^{kr}}\right)^{1/r}$$
$$= n\left(\frac{e^{k-1/r}}{(kr)^k}\right)$$

Troviamo i valori di k che verificano la tesi:

$$\left(\frac{e^{k-1/r}}{(kr)^k}\right) \leq \frac{1}{n^2}$$

Applicando il logaritmo ad entrambi i membri:

$$k - \frac{1}{r} - k(\ln k + \ln r) \leq -2 \ln n$$

$$k(1 - \ln k - \ln r) \leq \frac{1}{r} - 2 \ln n$$

Poiché $1/r > 0$, è possibile trascurarlo senza aggiungere nuove soluzioni per k. Esplicitando $\ln r$ si ottiene

$$k[1 - \ln k - (1-\epsilon)\ln n] \leq -2 \ln n$$

$$k\left(1 - \epsilon + \frac{\ln k - 1}{\ln n}\right) \geq 2$$

Il secondo fattore al primo membro è positivo per $k \geq e$, quindi

$$k \geq \frac{2}{\left(1 - \epsilon + \frac{\ln k - 1}{\ln n}\right)}$$

Per $k \geq 2/(1-\epsilon) \wedge k \geq e$, la disequazione è verificata. □

È interessante osservare il comportamento dell'ultima disuguaglianza per $\epsilon \to 1$, infatti si ottiene

$$k \ln k - k \geq 2 \ln n$$
$$\left(\frac{e}{k}\right)^k \leq \frac{1}{n^2}$$
$$n \left(\frac{e}{k}\right)^k \leq \frac{1}{n}$$

Il primo membro della disequazione coincide con l'espressione della probabilità di un carico massimo superiore a k nel Teorema 23, e di conseguenza è verificata per $k \geq 3 \ln n / \ln \ln n$ per n sufficientemente grande. In conclusione, per $\epsilon = 1$, l'applicazione delle disuguaglianze di Chernoff porta alla stessa conclusione del Teorema 23.

2.2.5 Numero di scatole vuote

Un evento di interesse particolare è l'assenza di scatole vuote. Il calcolo del numero atteso di scatole vuote è semplice, grazie alle proprietà dell'aspettazione.

Teorema 28. *Il numero atteso di scatole vuote è*

$$n \left(1 - \frac{1}{n}\right)^m \approx n e^{-m/n}$$

Per n sufficientemente elevato.

Dimostrazione. La probabilità che una determinata scatola sia vuota è

$$P[X_i = 0] = \left(1 - \frac{1}{n}\right)^m$$

Definiamo la variabile indicatrice ξ_i come

$$\xi_i = \begin{cases} 1 & \text{se } X_i = 0 \\ 0 & \text{altrimenti} \end{cases}$$

Allora $E[\sum_{i=0}^{n} \xi_i]$ è il numero atteso di scatole vuote e

$$E\left[\sum_{i=0}^{n} \xi_i\right] = n E[\xi_1] = n \left(1 - \frac{1}{n}\right)^m$$

□

Esempio Per $m = n \ln n$, il numero atteso di scatole vuote tende ad 1.

Con lo stesso procedimento si può ottenere il momento del secondo ordine:

$$\begin{aligned} E\left[\left(\sum_{i=0}^{n} \xi_i\right)^2\right] &= nE\left[\xi_1^2\right] + (n^2 - n)E[\xi_1 \xi_2] \\ &= n\left(1 - \frac{1}{n}\right)^m + (n^2 - n)\left(1 - \frac{2}{n}\right)^m \\ &\approx ne^{-m/n} + (n^2 - n)e^{-2m/n} \end{aligned}$$

Tuttavia, per calcolare la probabilità di avere un certo numero di scatole vuote nel caso esatto è necessario considerare le dipendenze tra le variabili X_i, in particolare per la probabilità di non avere scatole vuote (in [2] sono presenti alcuni risultati ottenuti tramite il principio di inclusione-esclusione).

È possibile sfruttare l'approssimazione di Poisson per eliminare la dipendenza tra le variabili e ricavare espressioni molto semplici. La probabilità che la singola scatola nel caso di Poisson sia vuota è

$$P[Y_1 = 0] = \frac{e^{-\mu}\mu^0}{0!} = e^{\mu}, \quad \mu = \frac{m}{n}$$

Poiché le v.a. Y_i sono indipendenti tra loro, allora il numero di scatole vuote segue semplicemente una distribuzione binomiale, con $p = e^{-m/n}$:

$$p(k) = \binom{n}{k}\left(e^{-m/n}\right)^k\left(1 - e^{-m/n}\right)^{n-k}$$

La probabilità di avere 0 scatole vuote è quindi

$$p(0) = \left(1 - e^{-m/n}\right)^n$$

Posto $m = n \ln n + cn$, si ottiene:

$$p(0) = \left(1 - e^{-(\ln n + c)}\right)^n = \left(1 - \frac{e^{-c}}{n}\right)^n \approx e^{-e^{-c}}$$

dove l'approssimazione è giustificata dal limite per $n \to \infty$. Questa probabilità tende rapidamente ad 1 all'aumentare di c.

Nel caso esatto, é facile notare che la probabilità che non ci siano scatole vuote è monotona crescente. Quindi , utilizzando il Corollario 13 si può concludere che entro $n \ln n + cn$ lanci tutte le scatole si riempiono con

probabilità al più $2e^{-e^{-c}}$. È possibile mostrare con ulteriori passaggi che l'approssimazione di Poisson è accurata per $n \to \infty$, e quindi ottenere una probabilità di $e^{-e^{-c}}$ anche per il caso esatto.

In [4] è presente una analisi più approfondita sul numero di scatole vuote dopo m lanci, in particolare si dimostra che il numero atteso di lanci affinché nessuna scatola sia vuota è $n \ln n + O(n)$. Approssimando la distribuzione del numero di lanci impiegati ad una v.a. di Poisson, si raggiunge un risultato simile, ma con passaggi più complessi.

Dato che il numero di scatole vuote nell'approssimazione di Poisson ha una distribuzione binomiale, possiamo sfruttare ancora le disuguaglianze di Chernoff.

Teorema 29. *Sia ξ il numero di scatole vuote nel caso di Poisson. Allora, per $1 < \alpha \leq n/\log_2 n$ ed $m \geq n \ln(6\alpha)$*

$$P\left[\xi \geq \frac{n}{\alpha}\right] \leq \frac{1}{n}$$

Dimostrazione. Per il Teorema 21, con $R \geq 6\mu = 6ne^{-m/n}$

$$P[\xi \geq R] \leq \frac{1}{2^R}$$

Con $R = n/\alpha$, nella condizione su R si ottiene:

$$\frac{n}{\alpha} \geq 6ne^{-m/n}$$

$$e^{-m/n} \leq \frac{1}{6\alpha}$$

La condizione è soddisfatta per $m \geq n \ln(6\alpha)$. Infine

$$\frac{1}{2^R} \leq \frac{1}{n} \implies \frac{n}{\alpha} \geq \log_2 n \implies \alpha \leq \frac{n}{\log_2 n}$$

□

Si può osservare come nel caso precedente che $P\left[\xi \geq \frac{n}{\alpha}\right]$, che rappresenta la probabilità che ci siano almeno n/α scatole vuote, è monotona decrescente in m sia nel caso di variabili aleatorie di Poisson sia nel caso esatto, quindi vale il Corollario 13 e $P\left[\xi \geq \frac{n}{\alpha}\right] = O(1/n)$ anche nel caso esatto.

Esempio Con $\alpha = 100$ e $n \geq 1000$, dopo $m \geq 6.4n$ lanci, con alta probabilità al più l'1% delle scatole risultano ancora vuote.

2.3 Variante: Il minimo tra due scatole

Una variante in grado di migliorare l'uniformità delle scatole nel processo Balls and Bins cerca un compromesso tra lanciare le palline in modo casuale e lanciarle nella scatola più vuota.

La variante consiste nello scegliere a caso d scatole come destinazioni candidate per ogni pallina, e di piazzare la pallina nella scatola più vuota tra queste. Rispetto al processo tradizionale, la variante rende inutilizzabili i metodi di analisi utilizzati precedentemente, in quanto la probabilità che una pallina finisca in una scatola dipende dall'occupazione delle scatole, e quindi dalle palline precedenti. Si può comunque mostrare come, con $d \geq 2$ costante, il carico massimo diventi $\Theta(\ln \ln n)$ con alta probabilità per $m = n$.

Teorema 30. *Si supponga che n palline siano inserite in sequenza in n scatole con il seguente criterio: per ogni pallina, tra $d \geq 2$ scatole scelte casualmente con probabilità uniforme e con reinserimento, la pallina viene inserita nella scatola più vuota, e nel caso in cui più scatole abbiano il minor numero di palline, in una di queste scelta casualmente. Dopo aver inserito tutte le palline, il carico massimo è al più* $\ln \ln n / \ln d + O(1)$ *con probabilità* $1 - o(1/n)$.

Il modo di gestire il caso in cui più scatole abbiano il numero minimo di palline tra le d scatole scelte è indifferente, purché non ci siano informazioni sulle d scatole scelte dalle palline inserite successivamente. Infatti, qualunque criterio di scelta in caso di parità conduce alla stessa distribuzione del carico, e quindi non influenza le possibili distribuzioni del carico nelle d scatole del lancio successivo. Induttivamente, le possibili distribuzioni del carico non sono influenzate per nessuna delle palline successive.

Si discute soltanto una traccia della dimostrazione, che si basa sul trovare per induzione un limite superiore approssimato al numero di scatole con *almeno* i palline (La dimostrazione completa si trova in [1] e in [10]). Per trovare questo limite superiore, anziché fare riferimento direttamente al numero di scatole, si suppone che le palline siano impilate all'interno di una scatola, e che ogni pallina abbia una altezza, definita come il numero di palline sotto di essa più uno. Il numero di scatole con almeno i palline è chiaramente limitato dal numero di palline con altezza almeno i.

Innanzitutto, analizziamo il processo istante per istante: l'istante $t \in \mathbb{N}$ si riferisce al momento successivo all'inserimento della t-esima pallina.

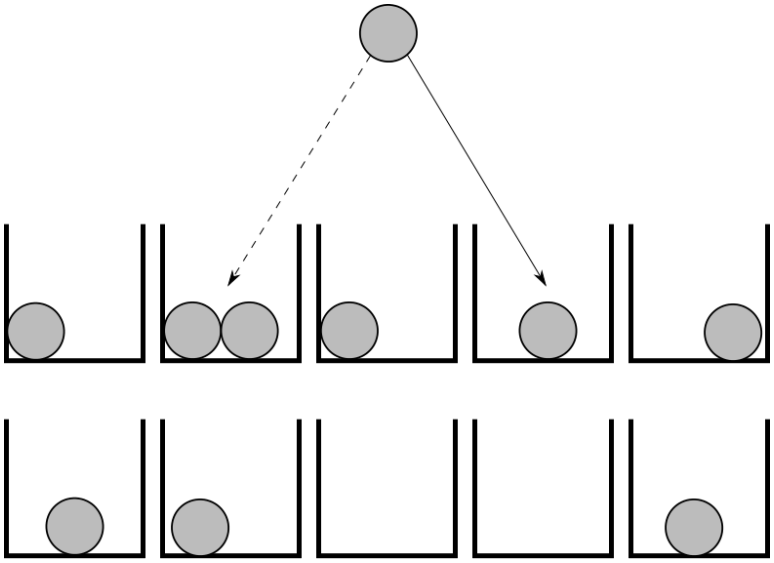

Figura 2.2: Lancio di una pallina utilizzando la variante. Le frecce rappresentano le $d = 2$ destinazioni candidate, la freccia non tratteggiata la destinazione finale.

Definiamo le seguenti variabili:

$h(t)$	altezza della t-esima pallina
$\nu_i(t)$	numero di scatole all'istante t con almeno i palline
$\mu_i(t)$	numero di palline all'istante t di altezza almeno i

Una pallina ha altezza $h(t) \geq i + 1$ se e solo se, al momento del suo lancio, tutte le d scatole candidate hanno già almeno i palline. Supponiamo che, per tutta la durata del processo, esiste una sequenza β_i tale che $\nu_i(t) \leq \beta_i$ con alta probabilità (usando come base del procedimento $\beta_4 = n/4$, infatti non possono esserci più di $n/4$ scatole con 4 palline). Allora per ciascuna pallina t, la sua altezza è almeno $i + 1$ se e solo se tutte e d le scatole candidate hanno altezza almeno i. La probabilità di scegliere queste d scatole è

$$P(h(t) \geq i + 1) = \left(\frac{\nu_i(t-1)}{n}\right)^d \leq \left(\frac{\beta_i}{n}\right)^d$$

Tramite una disuguaglianza di Chernoff, si può mostrare che il numero di palline con altezza al più $i + 1$ è al più $2n(\beta_i/n)^d$ con alta probabilità:

$$\mu_{i+1}(t) \leq 2n \left(\frac{\beta_i}{n}\right)^d$$

Chiaramente, $\nu_{i+1}(t) \leq \mu_{i+1}(t)$, quindi $\mu_{i+1}(t)$rappresenta un candidato per β_{i+1}:

$$\frac{\beta_{i+1}}{n} \leq 2\left(\frac{\beta_i}{n}\right)^d$$

Analizzando attentamente la ricorsione, si può mostrare che β_j è $O(\ln n)$ per $j = \ln\ln n/\ln d + O(1)$. Partendo da questo risultato, si può arrivare con ulteriori accorgimenti alla tesi.

Con un procedimento simile si può arrivare ad ottenere anche un limite inferiore al carico massimo:

Teorema 31. *Si supponga che n palline siano inserite in sequenza in n scatole con il seguente criterio: per ogni pallina, tra $d \geq 2$ scatole scelte casualmente con probabilità uniforme e con ripetizione, la pallina viene inserita nella scatola più vuota. Dopo aver inserito tutte le palline, il carico massimo è almeno $\ln\ln n/\ln d - O(1)$ con probabilità $1 - o(1/n)$.*

La dimostrazione di questo teorema procede in modo simile al precedente, stavolta cercando una sequenza γ_i tale che $\nu_i(n(1 - 1/2^i)) \geq \gamma_i$, con base $\gamma_0 = n$ (per $i = 0$ si ha $\nu_0(0) \geq n$, che è verificata dato che tutte le scatole hanno sempre un numero di palline non negativo). Per trovare questa sequenza, è necessario osservare ciò che succede nell'intervallo $[n(1 - 1/2^i), n(1 - 1/2^{i+1})]$.

In conclusione:

Teorema 32. *Si supponga che n palline siano inserite in sequenza in n scatole con il seguente criterio: per ogni pallina, tra $d \geq 2$ scatole scelte casualmente con probabilità uniforme e con ripetizione, la pallina viene inserita nella scatola più vuota. Dopo aver inserito tutte le palline, il carico massimo è $\ln\ln n/\ln d + \Theta(1)$ con probabilità $1 - o(1/n)$.*

Ciò implica che già con $d = 2$ scatole si ottiene una riduzione esponenziale nel carico massimo rispetto al processo standard, ma con costanti ancora maggiori si ottiene solo una riduzione del carico massimo per una costante.

Capitolo 3

Applicazioni di Balls and Bins

3.1 Bucket sort

Bucket sort è un algoritmo di ordinamento non basato su confronti, per il quale non è applicabile il lower bound di $n \log n$. Si suppone di voler ordinare n elementi, con chiavi di ordinamento intere comprese in un intervallo limitato. L'algoritmo inizialmente prepara un insieme di k "secchi" (bucket), che partizionano l'intervallo in modo ordinato, ovvero gli elementi nel primo secchio saranno sicuramente minori degli elementi del secondo secchio, e così via. Successivamente l'algoritmo piazza ogni elemento da ordinare nel rispettivo secchio, e alla fine ordina gli elementi all'interno di ciascun secchio (ad esempio con un algoritmo con caso peggiore $O(n^2)$). Concatenando gli elementi ordinati di ciascun secchio si ottiene l'array degli elementi ordinati.

La complessità di questo algoritmo, trascurando il tempo impiegato ad inserire gli elementi nei bucket, dipende fortemente da come sono distribuiti gli elementi nei secchi: se tutti gli elementi finiscono nello stesso secchio, il tempo impiegato per l'ordinamento è $O(n^2)$, infatti non otteniamo alcun vantaggio dalla suddivisione in bucket. Tuttavia, se gli elementi sono ben distribuiti, si possono ottenere dei vantaggi in base al numero di secchi utilizzati.

Se X_i è il numero di elementi nell' i-esimo secchio, il tempo atteso di ordinamento dell'i-esimo secchio è al più cX_i^2 per qualche costante c. Il tempo totale atteso è la somma del tempo impiegato per ogni secchio, e la

sua aspettazione è

$$E\left[\sum_{i=1}^{k} cX_i^2\right] = c\sum_{i=1}^{k} E\left[X_i^2\right]$$

Se, nell'insieme da ordinare, ogni elemento ha la stessa probabilità di finire in ciascun secchio, allora

$$E\left[\sum_{i=1}^{k} cX_i^2\right] = ckE\left[X_1^2\right]$$

Dove X_1 è ora una v.a. binomiale di parametri $(n, 1/k)$, quindi

$$E[X_1^2] = np(1-p) + (np)^2 = \frac{n}{k}\left(1 - \frac{1}{k}\right) + \left(\frac{n}{k}\right)^2 = \frac{n}{k}\left(1 + \frac{n-1}{k}\right)$$

Il tempo atteso è dunque

$$E\left[\sum_{i=1}^{k} cX_i^2\right] = cn\left(1 + \frac{n-1}{k}\right)$$

Supponendo di usare $k = n$ secchi, si ottiene $E[X_1^2] = (1-1/n)+1 = 2-1/n$, quindi

$$cnE\left[X_i^2\right] = 2cn - c \leq 2cn$$

Quindi, con tanti bucket quanti elementi e nell'ipotesi di distribuzione uniforme degli elementi, il tempo atteso di ordinamento è $O(n)$.

3.2 Hash table con concatenazione

Una hash table è una struttura dati che implementa un dizionario, ovvero un tipo di dati astratto composto da una collezione di chiavi, dove per ciascuna chiave è associato un valore. Le operazioni tipiche su un dizionario sono l'aggiunta o la rimozione di elementi e la ricerca del valore di un elemento del dizionario (se è presente) tramite la sua chiave.

L'implementazione tramite hash table consiste nell'utilizzare una *funzione hash* per suddividere gli elementi. Questa funzione associa ad ogni possibile valore della chiave (che può appartenere ad insiemi infiniti, per esempio stringhe di lunghezza qualsiasi) un valore (*hash*) scelto da un insieme limitato di n elementi. La tabella di hash è costituita da n celle destinate ad ospitare gli elementi secondo il rispettivo valore della funzione hash.

Poiché la funzione hash è chiaramente non iniettiva, possono esserci degli elementi con lo stesso hash. Se questo avviene tra gli elementi inseriti in una tabella di hash, si tratta di una *collisione*.

Sono presenti varie soluzioni per la gestione delle collisioni in una tabella di hash. La soluzione che più si avvicina al processo Balls and Bins è la concatenazione: ad ogni cella è associata una lista contenente tutti gli elementi con quell'hash. Quando si effettua la ricerca un elemento all'interno della tabella (per lettura o per eliminazione), è necessario scorrere la lista associata al rispettivo hash. La ricerca è l'operazione tipicamente più importante in una tabella di hash, e il tempo impiegato dipende sia dal numero di elementi inseriti nella tabella di hash, sia dalla loro distribuzione. La distribuzione uniforme degli hash è una proprietà molto desiderata della funzione di hash, e la progettazione di funzioni di hash che distribuiscano gli elementi in modo più vicino possibile ad una distribuzione casuale degli elementi è molto importante. Si prende in considerazione una ipotetica funzione hash uniforme, cioè una funzione tale che ogni hash ha la stessa probabilità di essere associato ad un elemento.

Si suppone di aver inserito m elementi in una tabella di hash con n celle con concatenazione, e di voler cercare in essa un elemento a partire dalla sua chiave (e quindi il suo hash). Se questo elemento non è presente nella tabella, è necessario scorrere tutta la lista della cella associata all'hash; altrimenti si deve scorrere la lista fino a quando si incontra la chiave cercata, quindi il tempo impiegato dipende dalla posizione occupata dall'elemento nella lista. In entrambi i casi il tempo impiegato per la ricerca è $O(l)$, dove l è la lunghezza della lista.

La lunghezza della lista equivale al numero di palline in una determinata scatola nel caso di Balls and Bins. Nel caso medio, il numero di palline in una scatola è m/n. Quindi, con un numero di elementi proporzionale al numero di liste della hash table, il tempo medio è $O(1)$. Tuttavia, nel caso in cui la lista in cui si effettua la ricerca è la più lunga, il tempo di ricerca potrebbe aumentare sensibilmente, fino a $O(m)$ nel caso sfavorevole di avere una sola lista non vuota. Tuttavia, utilizzando il Teorema 23, con $m = n$ si ottiene che il tempo di ricerca è $O(\ln n / \ln \ln n)$ con alta probabilità . Pur essendo più veloce di una ricerca binaria semplice, il tempo massimo è sensibilmente più alto della media, e in alcune applicazioni può essere una limitazione. Inoltre con $m = n$ può esserci uno spreco di memoria dovuto alle celle rimaste vuote. Una scelta ottimale per l'occupazione delle celle è quella di avere $m = n \ln n$, ma a costo di aumentare il tempo medio di ricerca a $O(\ln n)$.

Si possono ottenere risultati più interessanti tramite la variante introdot-

ta nella Sezione 2.3. Utilizzando due funzioni hash ed inserendo ogni elemento nella lista più vuota tra le due che corrispondono ai valori delle due funzioni hash, si ottiene un carico massimo per $m = n$ di $\Theta(\ln \ln n)$. Tuttavia, poiché la ricerca si svolge su due liste, il tempo medio di ricerca raddoppia. È possibile mitigare questo incremento effettuando la ricerca in parallelo sulle due liste.

3.3 Insieme approssimato

Un altro impiego delle funzioni hash è la costruzione di un insieme approssimato. L'obiettivo è quello di fornire una struttura dati che rappresenti un insieme, in modo che sapere se un elemento è presente o meno nell'insieme richieda il minor tempo possibile, senza tuttavia occupare memoria in modo eccessivo. Per raggiungere questi obiettivi, si accetta la possibilità di eventuali errori nella risposta.

Un modo per realizzare un insieme approssimato è quello di immagazzinare solamente gli hash degli elementi. Per sapere se un elemento appartiene all'insieme si verifica la presenza del suo hash (ad esempio tramite una ricerca binaria). Il vantaggio di utilizzare solo gli hash è la possibilità di ridurre drasticamente lo spazio occupato dall'insieme: tramite una funzione hash è possibile mappare elementi di dimensione arbitraria a stringhe binarie di lunghezza fissa. Esiste tuttavia la possibilità che questo insieme fornisca dei falsi positivi, nel caso in cui l'elemento da verificare abbia un hash coincidente con un hash di un altro elemento già inserito nell'insieme. Se non è ammessa la rimozione un elemento dall'insieme, allora non sono possibili falsi negativi.

Utilizzando una funzione hash uniforme, se l'insieme ha m elementi e la funzione hash restituisce una stringa di b bit, la probabilità di un falso positivo corrisponde alla probabilità che l'hash dell'elemento da verificare collida con uno degli hash inseriti finora, pur non avendo inserito l'elemento nell'insieme. Questa probabilità è

$$1 - \left(1 - \frac{1}{2^b}\right)^m \geq 1 - e^{-m/2^b}$$

Se si desidera che questa probabilità sia inferiore ad una costante c, si può ottenere un numero minimo di bit necessari per soddisfare questa richiesta

$$e^{-m/2^b} \geq 1 - c$$

$$-m \geq 2^b \ln(1 - c)$$

$$b \geq \log_2 m - \log_2 \ln\left(\frac{1}{1-c}\right) = \log_2 m + f(c)$$

Quindi, è necessario che $b = \Omega(\log_2 m)$. Tuttavia, anche solo con $b = 2\log_2 m$, sfruttando il binomio di Newton si ottiene:

$$\begin{aligned}
1 - \left(1 - \frac{1}{m^2}\right)^m &= 1 - \left[\sum_{k=0}^{m} \binom{m}{k} \frac{(-1)^k}{m^{2k}}\right] \\
&= 1 - \left[1 - \binom{m}{1}\frac{1}{m^2} + o\left(\frac{1}{m}\right)\right] \\
&= \frac{1}{m} - o\left(\frac{1}{m}\right) \\
&\leq \frac{1}{m}
\end{aligned}$$

In entrambi i casi, lo spazio occupato per elemento è solo di b bit per un array ordinato degli hash, al posto della dimensione media degli elementi. Inoltre, grazie all'ipotesi di uniformità della funzione hash, possiamo ottenere ulteriori miglioramenti nel tempo di ricerca immagazzinando gli hash in una struttura simile a quella di bucket sort. In questo modo, per effettuare la ricerca di un elemento, è sufficiente verificare il contenuto del bucket associato al valore di hash ottenuto. Con un numero di bucket proporzionale al numero di elementi da inserire, il tempo medio di ricerca diventa O(1).

3.4 Bloom Filter

3.4.1 Descrizione

Il Bloom filter è una struttura dati ideata da Burton Bloom [7] che permette di ottenere dei trade-off più interessanti per gli insiemi approssimati tra lo spazio impiegato e la probabilità di falsi positivi.

Il Bloom filter consiste di un array A di n bit, inizialmente tutti a 0, e di k funzioni hash uniformi e indipendenti tra loro $h_1, h_2, \ldots, h_k$ (In [6] si afferma che è sufficiente una famiglia di funzioni nella forma $g_i(x) = h_1(x) + ih_2(x), 1 \leq i \leq k$) . L'inserimento di un elemento x nel Bloom filter consiste nel porre $A[h_i(x)] = 1$, con $1 \leq i \leq k$. Per verificare se l'elemento è presente nell'insieme S degli elementi inseriti, se per qualche i si ha $A[h_i(x)] = 0$, allora chiaramente l'elemento non fa parte dell'insieme. Se $A[h_i(x)] = 1$ per ogni i, allora assumiamo che l'elemento appartenga all'insieme. È possibile tuttavia che tutti i k bit associati siano ad 1 per effetto di elementi diversi da x, che potrebbe quindi non essere nell'insieme.

3.4.2 Probabilità di falsi positivi

Dopo l'inserimento di m elementi, la probabilità che una specifica cella sia ancora a 0 è

$$p = \left(1 - \frac{1}{n}\right)^{km} \approx e^{-km/n}$$

Quindi, il numero di celle a 0 segue una distribuzione binomiale con parametri $(n, p \approx e^{-km/n})$. Per semplificare l'analisi assumiamo che il numero di celle a 0 sia uguale alla media, e che quindi p sia anche la frazione di celle a 0 sul totale. La probabilità di un falso positivo, ovvero la probabilità che un elemento non appartenente all'insieme abbia le celle corrispondenti a 1 è quindi

$$f = P\left[A[h_i(x)] = 1 \,\forall i \,|\, x \notin S\right] = (1-p)^k = (1 - e^{-km/n})^k$$

Fissati n ed m, al crescere di k la probabilità p che una cella sia a 0 diminuisce, rendendo più probabile la collisione di una singola funzione di hash, ma allo stesso tempo con k aumenta il numero di collisioni necessarie per un falso positivo e quindi contribuirebbe ad una diminuzione di f se p fosse indipendente da k. Per trovare il minimo, usiamo $g = \ln f = k \ln(1 - e^{-km/n})$ e troviamo il minimo in funzione di k derivando:

$$\begin{aligned}
\frac{dg}{dk} &= \ln(1 - e^{-km/n}) + \frac{k}{(1 - e^{-km/n})}\left(-e^{-km/n}\right)\left(-\frac{m}{n}\right) \\
&= \ln(1 - e^{-km/n}) + \frac{km}{n}\frac{e^{-km/n}}{1 - e^{-km/n}}
\end{aligned}$$

e ponendo la derivata a 0:

$$\begin{aligned}
\frac{dg}{dk} = 0 &\implies (1 - e^{-km/n})\ln(1 - e^{-km/n}) = -\frac{km}{n}e^{-km/n} \\
&\implies 1 - e^{-km/n} = e^{-km/n} \\
&\implies k = (\ln 2)\frac{n}{m}
\end{aligned}$$

Questo è un punto di minimo globale (trascurando il fatto che k deve essere intero), e corrisponde ad una probabilità di falsi positivi di $f = (1/2)^k = (1/2)^{(\ln 2)n/m} \approx (0.6185)^{n/m}$. Questa probabilità è esponenziale decrescente nel numero di celle del Bloom filter per ogni elemento. Se si desidera che questa probabilità sia inferiore ad una costante c, è sufficiente porre

$$n \geq m\frac{\ln c}{|\ln 0.6185|}$$

Una probabilità di falsi positivi di circa 1% può essere soddisfatta per $n/m \approx 9.58, k \approx 6.64$. Confrontato questi risultati con le bit string nella sezione 3.3, anziché usare $\Omega(\log_2 m)$ bit per elemento si utilizzano solamente un numero costante di bit per elemento in un Bloom filter per ottenere una probabilità costante di falsi positivi. Inoltre, il tempo impiegato per verificare la presenza di un elemento è costante (escluso il tempo per calcolare i k hash).

Si noti inoltre che per $k = (\ln 2)n/m$ si ha $p = 1/2$. Un Bloom filter ottimale è quindi riempito in modo tale da avere approssimativamente lo stesso numero di celle a 1 e a 0.

3.4.3 Utilizzo

L'efficienza sia in termini di spazio che di tempo di ricerca rendono il Bloom Filter un ottimo filtro per evitare accessi costosi a strutture di dati. È il caso di BigTable di Google, un DBMS proprietario di Google, che impiega i Bloom Filter per evitare accessi al disco quando si richiedono righe o colonne non esistenti [8]. Anche le cache dei proxy Squid sfruttano i Bloom Filter per poter stabilire rapidamente se un oggetto è presente nella cache del proxy tramite il suo URL.

I Bloom Filter possono inoltre essere utilizzati per verificare se una stringa fa parte di un particolare insieme . Per verificare l'esistenza di un match, si verifica se la stringa è presente nel Bloom Filter in cui sono stati inseriti gli elementi dell'insieme. Ad esempio, può essere utilizzato per verificare l'appartenenza di un URL ad una blacklist di siti malevoli, senza utilizzare l'intera blacklist. Se il riscontro è positivo, si può procedere ad una ricerca vera e propria, ad esempio interrogando un server che ospiti la blacklist.

Infine, nella sincronizzazione di dati tra più entità, i Bloom Filter possono essere utilizzati per confrontare i dati posseduti da un'entità con quelli delle altre entità, e procedere all'invio dei dati mancanti [9].

3.5 Load balancing

Il processo Balls and Bins è utilizzabile anche nel load balancing. Nello scenario attuale, infatti, è molto frequente trovarsi in situazioni in cui si rendono disponibili molti server per la gestione dei vari compiti inviati dai client, con l'obiettivo di suddividere il carico totale di compiti fra più server anziché gestirli centralmente. Tuttavia è necessario garantire che il carico dei singoli server sia piuttosto uniforme per poterli sfruttare in modo efficiente.

Un modo per distribuire il carico tra i server è quello di avere un load balancer centralizzato che si occupi sia di tenere traccia del carico dei server (e di eventuali fattori determinanti per migliorare le prestazioni), sia di ricevere i compiti e di assegnarli in modo da distribuire efficientemente il carico. In questo modo è possibile ottenere risultati ottimi, ma richiede un lavoro di coordinazione che talvolta può essere proibitivo.

In alternativa, è possibile assegnare i compiti in modo casuale, eventualmente in base alle prestazioni di ciascun server. Nel caso in cui i server abbiano tutti le stesse prestazioni, il processo Balls and Bins è in grado di mostrare vantaggi e limiti della randomizzazione.

Rispetto ad un load balancing deterministico, è possibile avere situazioni in cui un server è sensibilmente più carico rispetto alla media. In compenso si è visto che per $m = n$, il carico massimo è con alta probabilità al più $O(\log n / \log \log n)$, che tuttavia può essere sensibilmente più elevato del carico medio, che è solamente 1. Con la variante del minimo tra le due scatole, è possibile portare il carico massimo ad una soglia inferiore di $O(\log \log n)$, in cambio di una piccola verifica del carico su due server anziché n. Se invece $m \geq n \log n$, si è visto che il carico massimo è contenuto a 6 volte rispetto al carico medio, che può comunque essere rilevante in certe condizioni.

Un altro evento indesiderato è la presenza di server senza compiti, corrispondenti ad una scatola vuota nel processo. Oltre a costituire uno spreco di risorse, un server poco carico o addirittura vuoto implica un aumento del carico medio degli altri server. Mentre un load balancing deterministico può essere in grado di ottenere zero scatole vuote per $m = n$, per il Teorema 28 un criterio casuale fornisce in media un numero di scatole vuote pari a circa n/e. Gli altri server di conseguenza saranno più carichi: supponendo che il numero di scatole vuote sia proprio n/e, i server restanti avranno un carico di circa il 58% in più rispetto alla media. Anche in questo caso la situazione migliora per $m \geq n \log n$, ottenendo un numero medio di server inattivi inferiore a 1.

Nel caso in cui vi sia abbondanza di server rispetto alle richieste ($m < n$) e sia critico avere il minor numero di collisioni possibili, come si è visto per il paradosso del compleanno è necessario che $m \leq \sqrt{2n \ln 2}$ perché la probabilità di avere carico massimo 1 sia maggiore di $1/2$.

Bibliografia

[1] Michael Mitzenmacher, Eli Upfal, *Probability and Computing*, Cambridge University Press, 2005.

[2] N. Johnson, S. Kotz, *Urn Models and Their Applications*, Wiley , 1977.

[3] William Feller, *An Introduction to Probability Theory and Its Applications*, Wiley, 1968.

[4] V. F. Kolchin, B. A. Sevsat'yanov, and V. P. Chistyakov, *Random Allocations*, V. H. Winston & Sons, 1978.

[5] Martin Raab, Angelika Steger, *"Balls into Bins" - A Simple and Tight Analysis*, in *Proceedings of the Second International Workshop on Randomization and Approximation Techniques in Computer Science (RANDOM '98)*, 1998.

[6] Adam Kirsch, Michael Mitzenmacher, *Less hashing, same performance: Building a better Bloom filter*, in *Lecture Notes in Computer Science*, Volume 4168/2006, pp. 456-467,

[7] Burton H. Bloom, *Space/Time Trade-offs in Hash Coding with Allowable Errors*, in *Communications of the ACM*, Volume 13, Number 7, 1970.

[8] Fay Chang et al., *Bigtable: A Distributed Storage System for Structured Data*, in *OSDI'06: Seventh Symposium on Operating System Design and Implementation*, 2006.

[9] J.W.Byers et al., *Informed content delivery across adaptive overlay networks*, in *IEEE/ACM Transactions on Networking*, Volume 12, Issue 5, 2004.

[10] Y. Azar, A. Broder, A. Karlin, e E. Upfal. *Balanced allocations*, in *Proceedings of the 26th ACM Symposium on the Theory of Computing*, pagg.593-602, 1994.