Probabilistic Method and Random Graphs

Lecture 6. Poisson Approximation with Application [1]

Xingwu Liu

Institute of Computing Technology
Chinese Academy of Sciences, Beijing, China

[1] The slides are mainly based on Chapter 5 of the textbook *Probability and Computing* and Lecture 12 of Ryan O'Donnell's lecture notes of *Probability and Computing*.

Preface

Questions, comments, or suggestions?

A recap of Lecture 5

Poisson Convergence (LSN)

Assume that $X_n \sim Bin(n, p_n)$ with $\lim_{n\to\infty} np_n = \lambda$. For any fixed k, $\lim_{n\to\infty} \Pr(X_n = k) = \frac{e^{-\lambda}\lambda^k}{k!}$.
Valid when weakly dependent.
Almost Valid when strongly dependent (Stein-Chen Theorem).

Joint distribution of bin loads

Max load: $L(n,n) < 3\frac{\ln n}{\ln\ln n}$ with high probability.
$\Pr(X_1 = k_1, ...X_n = k_n) = \frac{m!}{k_1!k_2!\cdots k_n!n^m}$

Poisson approximation theorem

- $(X_1^{(m)}, X_2^{(m)}, ...X_n^{(m)}) \sim (Y_1^{(\mu)}, Y_2^{(\mu)}, ...Y_n^{(\mu)} | \sum Y_i^{(\mu)} = m)$

Application: coupon collector's problem

$\lim_{n\to\infty} \Pr(X > n\ln n + cn) = 1 - e^{-e^{-c}}$

Poisson approximation is nice but ...

Hard to use due to conditioning.

Can we remove the condition?

Condition-free Poisson Approximation

Notation

$X_i^{(m)}$: the load of bin i in (m,n)-model.
$Y_i^{(m)}$: independent Poisson r.v.s with expectation $\frac{m}{n}$.

Theorem

For any non-negative n-ary function f, we have
$\mathbb{E}[f(X_1^{(m)},...X_n^{(m)})] \leq e\sqrt{m}\mathbb{E}[f(Y_1^{(m)},...Y_n^{(m)})]$.

Remark

Unlike $(X_1^{(m)}, X_2^{(m)}, ...X_n^{(m)}) \sim (Y_1^{(\mu)}, Y_2^{(\mu)}, ...Y_n^{(\mu)} | \sum Y_i^{(\mu)} = m)$, the mean of the Poisson distribution is $\frac{m}{n}$, not arbitrary.
Condition-freedom at the cost of approximation.

Proof

$$\begin{aligned}
&\mathbb{E}[f(Y_1^{(m)},...Y_n^{(m)})] \\
&\quad = \sum_k \mathbb{E}[f(Y_1^{(m)},...Y_n^{(m)})| \sum_i Y_i^{(m)} = k] \Pr(\sum_i Y_i^{(m)} = k) \\
&\quad \geq \mathbb{E}[f(Y_1^{(m)},...Y_n^{(m)})| \sum_i Y_i^{(m)} = m] \Pr(\sum_i Y_i^{(m)} = m) \\
&\quad = \mathbb{E}[f(X_1^{(m)},...X_n^{(m)})] \Pr(\sum_i Y_i^{(m)} = m).
\end{aligned}$$

$\sum_i Y_i^{(m)} \sim Poi(m) \Rightarrow \Pr(\sum_i Y_i^{(m)} = m) = \frac{m^m e^{-m}}{m!} \geq \frac{1}{e\sqrt{m}}$ since $m! < e\sqrt{m}(me^{-1})^m$.

Remark

$\mathbb{E}[f(X_1^{(m)},...X_n^{(m)})] \leq 2\mathbb{E}[f(Y_1^{(m)},...Y_n^{(m)})]$ if f is monotonic in m

In Terms of Probability

Any event that takes place with probability p in the independent Poisson coupling takes places in Bins&Balls setting with probability at most $pe\sqrt{m}$

If the probability of an event in Bins&Balls is monotonic in m, it is at most twice of that in the independent Poisson coupling

Remark

Use approximate experiments to bound the exact-case probability. Powerful in bounding the probability of rare events in Bins&Balls.

Lower bound of max load in (n,n)-model

Asymptotically, $\Pr(\mathcal{E}) \leq \frac{1}{n}$, where $\mathcal{E}$ is the event that the max. load in the (n,n)-Bins&Balls model is smaller than $\frac{\ln n}{\ln\ln n}$.

Remark: In fact, the max. load is $\Theta\left(\frac{\ln n}{\ln\ln n}\right)$ w.h.p.

Proof

In the Poisson approximation, a bin has at least $M = \frac{\ln n}{\ln\ln n}$ balls with probability at least $\frac{1}{eM!} \Rightarrow \Pr(\mathcal{E}') \leq \left(1 - \frac{1}{eM!}\right)^n \leq e^{-\frac{n}{eM!}}$.

$M! \leq e\sqrt{M}(e^{-1}M)^M \leq M(e^{-1}M)^M$
$\Rightarrow \ln M! \leq \ln n - \ln\ln n - \ln(2e) \Rightarrow M! \leq \frac{n}{2e\ln n}$.

Altogether, $\Pr(\mathcal{E}) \leq e\sqrt{n}\Pr(\mathcal{E}') \leq \frac{e\sqrt{n}}{n^2} \leq \frac{1}{n}$.

Application: Hashing

Used to look up records, protect data, find duplications ...

Membership problem: password checker

Binary search vs Hashing

Hash table (1953, H. P. Luhn @IBM)

Hash functions: efficient, deterministic, uniform, non-invertible

Random: coin tossing, SUHA

SHA-1 (broken by Wang et al., 2005)

Bins&Balls model

Efficiency

Search time for m words in n bins: expected vs worst.

Space: $\geq 256m$ bits if each word has 256 bits.

Potential wasted space: $\frac{1}{e}$ in the case of $m = n$.

Trade space for time. Can we improve space-efficiency?

Information Fingerprint

Fingerprint

Succinct identification of lengthy information

Fingerprint hashing

Fingerprinting $\rightsquigarrow$ sorting fingerprints (rather than original data)
$\rightsquigarrow$ binary search.

Trade time for space

Performance

False positive: due to loss of information

No other errors

Partial correction using white lists

False positive

Probability of a false positive: m words, b bits

Fingerprint of an acceptable differs from that of a bad: $1 - \frac{1}{2^b}$.
Probability of a false positive: $1 - \left(1 - \frac{1}{2^b}\right)^m \geq 1 - e^{-\frac{m}{2^b}}$.

Determine b

For a constant c, false positive $< c \Rightarrow e^{-\frac{m}{2^b}} \geq 1 - c$.
So, $b \geq \log_2 \frac{-m}{\ln(1-c)} = \Omega(\ln m)$.
If $b \geq 2\log_2 m$, false positive $< \frac{1}{m}$.

2^{16} words, 32-bit fingerprints, false positive $< 2^{-16}$.
Save a factor of 8 if each word has 256 bits.

Can more space be saved while getting more time-efficient?

Bloom Filter

1970, CACM, by Burton H. Bloom.

Used in Bigtable and HBase.

Basic idea

Hash table + fingerprinting
Illustration

False positive is the only source of errors.

False positive: m words, n-bit array, k mappings

A specific bit is 0 with probability $\left(1 - \frac{1}{n}\right)^{km} \approx e^{-\frac{km}{n}} \triangleq p$.
Resonable to assume that a fraction p of bits are 0.
  By Poisson approximation and Chernoff bounds.
False positive probability: $f \triangleq \left(1 - \left(1 - \frac{1}{n}\right)^{km}\right)^k \approx \left(1 - e^{-\frac{km}{n}}\right)^k$

Determine k for fixed m, n

Objective

Minimize f.
Dilemma of k: chances to find a 0-bit vs the fraction of 0-bits.

Optimal k

$\frac{d \ln f}{dk} = \ln\left(1 - e^{-\frac{km}{n}}\right) + \frac{km}{n}\frac{e^{-\frac{km}{n}}}{1-e^{-\frac{km}{n}}}$.
$\frac{d \ln f}{dk}\Big|_{k=\frac{n}{m}\ln 2} = 0$.
$f\big|_{k=\frac{n}{m}\ln 2} = 2^{-k} \approx 0.6185^{n/m}$.
$f < 0.02$ if $n = 8m$, and $f < 2^{-16}$ if $n = 23m$, saving 1/4 space

Remark

Fix n/m, the #bits per item, and get a constant error probability.
In fingerprint hashing, $\Omega(\ln m)$ bits per item guarantee a constant error probability

Reference

Lecture 12 of the CMU lecture notes by Ryan O'Donnell.