Lecture 10: Sampling from Databases Final Review

Tuesday, June 2nd, 2009

Outline

- Sampling from databases
  - Not on the final, but useful anyway…
- Final review

Bernoulli Distribution

Consider the following random variable X

- X = 0 with probability 1-p
- X = 1 with probability p

What are the atomic events ?

What is the expected value of X ?

Bernoulli Distribution

Consider the following random variable X

- $X = 0$ with probability $1-p$
- $X = 1$ with probability p

What are the atomic events ?

- A: $\{0, 1\}$, $p_0 = 1-p$, $p_1 = p$

What is the expected value of X ?

- A: $E[X] = p$

Binomial Distribution

- Let n independent and identically distributed (iid) Bernoulli variables $X_1, \ldots, X_n$
- Define the random variable
  $X = X_1 + \ldots + X_n$
- Or their average:
  $Y = (X_1 + \ldots + X_n)/n$

Binomial Distribution

$X = X_1 + \ldots + X_n$

What are the atomic events ?

What is the expected value of X ?

Binomial Distribution

$X = X_1 + \ldots + X_n$

What are the atomic events ?

- A: set of atomic events is $\{0,1\}^n$

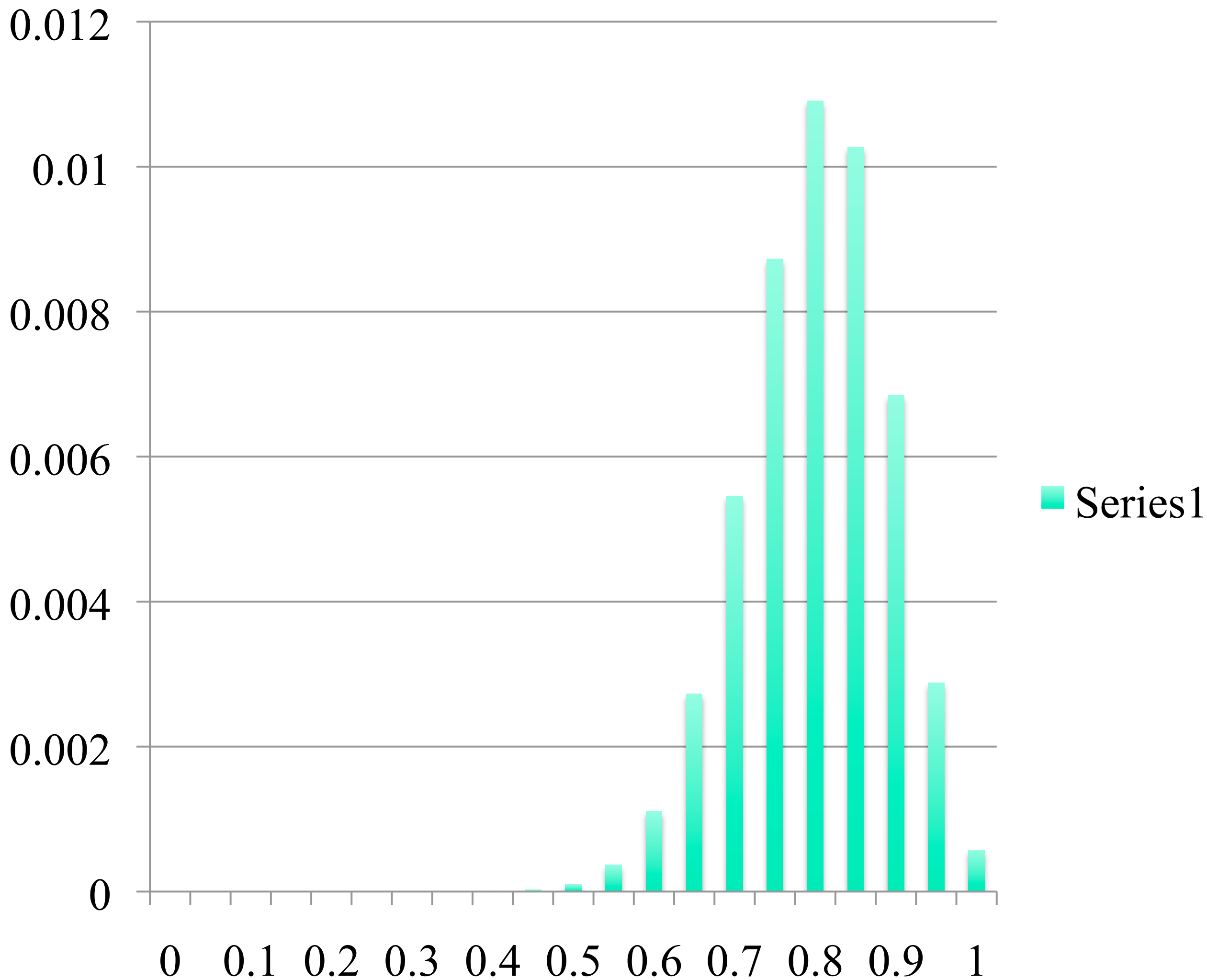
What is the expected value of X ?

- $E[X] = np$, assuming $X_1 \ldots X_n$ are identical and $E[X_1] = \ldots = E[X_n] = p$

Example: Binomial Distribution

A compute the *density* of $X = X_1 + \ldots + X_n$:

- $P[X=0] = \binom{n}{0}(1-p)^n$
- $P[X=1] = \binom{n}{1}p(1-p)^{n-1}$
- …
- $P[X=k] = \binom{n}{k}p^k(1-p)^{n-k}$
- …
- $P[X=n] = \binom{n}{n}p^n$



Density of Y = (X1 + … +Xn) / n, when p=0.8

Random Sampling from Databases

- Given a relation $R = \{t_1, \ldots, t_n\}$

- Compute a sample S of R

Random Sample of Size 1

- Given a relation $R = \{t_1, \ldots, t_n\}$
- Compute random element s of R

Q: What is the probability space ?

Random Sample of Size 1

- Given a relation $R = \{t_1, \ldots, t_n\}$
- Compute random element s of R

Q: What is the probability space ?

A: Atomic events: $t_1, \ldots, t_n$,
Probabilities: $1/n, 1/n, \ldots, 1/n$

Random Sample of Size 1

```
Sample(R) {
   r = random_number(0..2^32–1);
   n = |R|;
   s = “the (r % n)’th element of R”
   return s;
}
```

Random Sample of Size 1

Sequential scan

```
Sample(R) {
   forall x in R do {
         r = random_number[0..1];
         if (r ≤  ???) s = x;
   }
   return s;
}
```

Fill in the ??? Note the challenge: we don't use the size of R

Random Sample of Size 1

Sequential scan

```
Sample(R) {  k = 1;
   forall x in R do {
         r = random_number[0..1];
         if (r ≤ 1/k++) s = x;
   }
   return s;
}
```

Note: need to scan R fully. How can we stop early ?

Random Sample of Size 1

Sequential scan: use the size of R

```
Sample(R) {  k = 0;
   forall x in R do { k++;
         r = random_number[0..1];
         if (r ≤ 1/(n - k +1) return x;
    }
   return s;
}
```

Binomial Sample

In practice we want a sample > 1

```
Sample(R) {  S = emptyset;
   forall x in R do {
         r = random_number[0..1];
         if (r ≤ p) insert(S,x);
   return S;
}
```

What is the problem with binomial sample ?

Binomial Sample

- The size of the sample S is not fixed
- Instead it is a random binomial variable of expected size pn
- In practice we want a guarantee on the sample size, i.e. we want the sample size = m

Fixed Size Sample

Problem:

- Given relation R with n elements
- Given m > 0
- Sample m distinct values from R

What is the probability space ?

Fixed Size Sample

Problem:

- Given relation R with n elements
- Given m > 0
- Sample m distinct values from R

What is the probability space ?

A: all subsets of R of size m, each has probability $1/\binom{n}{m}$

Reservoir Sampling: known population size

Here we want a sample S of fixed size m from a set R of known size n

```
Sample(R) {  S = emptyset; k = 0;
   forall x in R do { k++;
         p = (m-|S|)/(n-k+1)
         r = random_number[0..1];
         if (r ≤  p) insert(S,x);
   return S;
}
```

Reservoir Sampling: unknown population size

```
Sample(R) {  S = emptyset; k = 0;
   forall x in R do
         p = |S|/k++
         r = random_number[0..1];
         if (r ≤ p) if (|S|=m) remove a random
                                           element from S;
                        insert(S,x);}
   return S;
}
```

Question

- What is the disadvantage of not knowing the population size ?

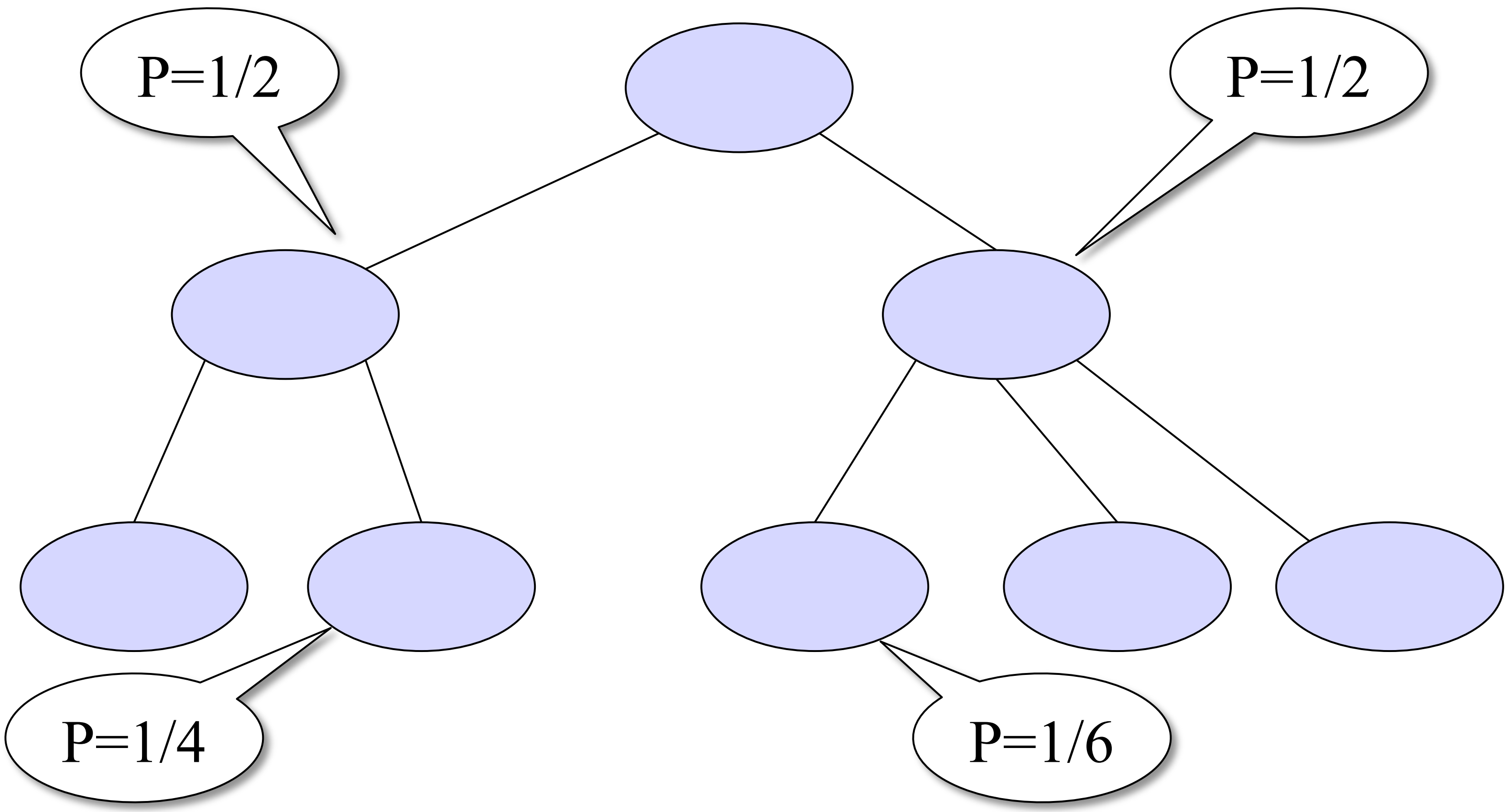
Sampling from a B+ Tree

- **Sample a single record s** from the leaves of the B+ tree
  - Make sure each record has the same probability !
- **Sample a set of records S** from the leaves of the B+ tree
  - Same idea, but more complicated
  - Omitted in class

Sampling from a B+ Tree

- Start from the root node x_1
- If x_i has fanout f_i, choose one child at random
  - Each child has probability $1/f_i$
- If x_h is a leaf with f_h records, choose a record at random
  - Each record has probability $1/f_h$

A Problem…



Leaves have different probabilities ! This is a problem..

A Problem…

- Consider a record s in a leaf, and let f_1, f_2, …, f_h be the fanouts from the root to that record
- The probability that this leaf record is selected is:

$$\boxed{p(s) = 1/f_1 f_2 \ldots f_h}$$

We want this probability to be independent on the path !

A Solution !

- Use rejection sampling !
- Let f_{max} = maximum fanout
- At each node x_i:
  - With probability f_i/f_{max} accept the choice of the child, and continue
  - With remaining probability reject, and start all over

Why This Works

- The probability that a record s is selected is:

$$\boxed{p(s) = 1/f_1 f_2 \ldots f_h}$$

- The probability that this path is accepted (not rejected) is:

$$\boxed{accept(s) = f_1/f_{max} \times f_2/f_{max} \times \ldots \times f_h/f_{max}}$$

After multiplying them → independent on the path

Sampling from a B+ Tree

- Rejection sampling needs multiple trials to return one sample
  - The expected number of trials: $1/\text{accept}(s) \approx f_{max}/f_1 \times f_{max}/f_2 \times \ldots \times f_{max}/f_h$
- Improvements: if we knew the number of records in each subtree then we could use *weighted sampling*
  - Why don't we store the number of records in each subtree of a B+ tree ?

Summary of this Course

Roles we played:

- Data manager / administrator:
  - SQL, database design, tuning
- Application writer
  - JDBC, Transactions
- Systems developer
  - Implementation, query processing
- General-purpose data user:
  - XML, sampling

What We Have Not Covered

- Parallel databases
  - Old stuff: parallel operators (joins, groupby)
  - Hot stuff: map/reduce, Scope, Dryad…
- Database as a service
  - Bottom line: less functionality for less cost
- Lots of adjacent topics:
  - Data mining, data privacy, uncertain/ probabilistic data

The Final

- Open books, open notes, access to the computer.
- No communication/collaborations allowed with your colleagues.
- Questions ? Send email

The Final

- Posted: Tuesday, June 2nd, 9:30pm.
- Turn in by: Thursday, June 4th, 11:59pm.
- https://catalysttools.washington.edu/collectit/dropbox/bhushan/5598
- What to turn in: text file, or Word file.
- WRITE YOUR NAME !

Problem 1: Relational Model

- SQL ! Both schema design and queries
- Same level of difficulty as homework
- Note: you don't need to test your SQL queries

Problem 2: FDs and DB Design

- Review the theory of FDs
- Lecture notes should suffice here

Problem 3: Transactions

- Concurrency control
  - Use lecture notes and/or book
- Recovery
  - Note: the book has an excellent description of ARIES

Problem 4: Indexes

- A little, fun question on a clever use of an index…

Problem 5: Query Execution/ Optimization

- From SQL → Relational Algebra
- Make sure you understand how to compute the cost of a plan
  - Lecture notes are helpful here
- Algebraic identities

Problem 6: XML/XPath/XQuery

- You will have to write some simple XPath, XQuery expressions

The End