15-499: Parallel Algorithms **Lecturer:** Guy Blelloch
Topic: Randomization **Date:** February 3, 2009
Scribe: Jonathan Coens

6.1 Today

- Finish pointer jumping for prefix sums
- Probability + randomized parallel algorithms

6.2 Pointer Jumping

do log(n) times
for i=[1:n] in parallel
v[i] = v[i] + v[p[i]]
p[i] = p[p[i]]

Depth:
O(log n) since each step is constant depth and done (log n) times

Work:
Each step does O(n) work to compute each pointer jump. This coupled with doing (log n) steps yields total work of O(n log n)

6.3 Probability 101++

- Define Sample space to be a set, Ω, of elementary events
  For example, an elementary event could be a single coin flip;

- Define Probability Measure to be a function $p(e) : \Omega \rightarrow [0, 1]$ such that $\sum_{e \epsilon \Omega} p(e) = 1$.
  In other words, the sum of all the probabilities must equal 1.

- Define a Random Variable to be a function $X : \Omega \rightarrow \Re$, mapping events to real numbers.

- Define an Event, E, to be a subset of Ω, e.g. all throws that sum to an even number

- The event notation $(x = a)$ is equivalent to saying $\{e \epsilon \Omega | x(e) = a\}$. Both of these denotes the event when the random variable x is a.

- Define Expectation to be $E[x] = \sum_{e \epsilon \Omega} p(e) * x(e)$ or $\sum_i ip(x = i)$. This is the weighted sum based on value and probability.

- Linearity of expectations: $E[X + Y] = E[X] + E[Y]$, if X and Y are independent.

- The Union Bound says $Pr[\bigcup_{i \epsilon I} E_i] \leq \sum_{i \epsilon I} Pr[E_i]$

- Define an Indicator Random Variable to be a function $X : \Omega \rightarrow \{0, 1\}$. An indicator random variable is essentially an indicator of certain property, mapping to 1 if the property (such as even-ness of a number) holds and to 0 otherwise.

- Define the variance for some random variable X to be $Var[X] = E[(X - E[X])^2]$

6.4 High probability bounds

While expectation of a random variable is about averages, we usually want to bound the distance (deviation) of an event from it's average.

Example: Let's say that an algorithm needs to toss a biased coin (HEAD occurs with probability p) till a HEAD occurs and we want to make sure that this will not take more than a certain number of attempts with *high probability* in terms of the input size n. Let us define the random variable X to be the number of coin tosses (of an unbiased coin) till we get a HEAD. The probability that the first HEAD happens on the k-th coin toss is exactly $(1 - p)^{k-1} \cdot p$. This means that the expected number of coin tosses till the first HEAD occurs is just $\sum_{i=0}^{\text{inf}} i \cdot (1 - p)^{i-1}.p = 1/p$. However, the probability that there is no HEAD in the first $(c \log n)$ flips is exactly $(1 - p)^{c \log n}$. If p is a constant like $1/2$, this latter bound which says that $X < 2 \log n$ with probability $1 - 1/n^c$ for some positive constant c is a *high probability* bound.

Some inequalities that relate the probability that a random variable deviates from mean by a certain quantity to the expected value of the random variable:

- Markov Inequality: $Pr[X > a] < \frac{E[x]}{a}$

- Chebychev Inequality: $Pr[(x - E[x]) > a] < \frac{Var[X]}{a^2}$

These are both bounds on the first and second moment respectively, but are not general enough.

6.5 Chernoff Bounds

Chernoff Bounds are more general equation and capture higher moments of the distribution. They are very powerful when used properly. For example, you can use Chernoff Bounds to find the probability that a random algorithm will go to a certain depth.
The equations are as follow:
Given a set $X_1, X_2, ...X_n$ of indicator random variables for independent trials with $Pr[X_i] = p_i$, $X = \sum X_i$, $\mu = E[X]$, $\delta > 0$:

$$Pr[X > (1+\delta)\mu] < \left(\frac{e^\delta}{(1+\delta)^{(1+\delta)}}\right)^\mu . \tag{6.5.1}$$

When $0 < \delta < 1$,

$$Pr[X > (1+\delta)\mu] < e^{\frac{-\mu\delta^2}{3}} \tag{6.5.2}$$

$$Pr[X < (1-\delta)\mu] < e^{\frac{-\mu\delta^2}{2}} \tag{6.5.3}$$

6.6 Quicksort Depth

Theorem 6.6.1 *Recursion depth of Randomized Quicksort is bounded by* $d = (10/\log_e 4/3)\log n$ *with probability* $\frac{1}{n}$

Proof: To bound the depth of randomized quicksort to less than a depth of d, we will use Chernoff Bounds. Consider the leaf of the longest branch in randomized quicksort, looking at each split / chosen pivot on the way down this branch. Associate a random variable X_i with each split. $X_i = 1$ if the split is better (more balanced) than 75-25 (call such splits good splits), and 0 otherwise. With probability $1/2$, the pivot lands in the middle 50% of the list, i.e., $X_i = 1$ with probability $1/2$. To cover worst case, assume that each good split split the list 75:25, and this branch got the larger section. We need at least $\log_{4/3} n$ good splits before the problem size goes down to a constant. Since out of d successive splits, we expect at least $d/2 = (5/\log_e 4/3)\log_n = 5\log_{4/3} n$ splits to be good, we show that it can not be the case that there are less than $\log_{4/3} n$ good splits with high probability. For this, let X be the sum of the k independent variables X_i. $\mu = E[X] = k/2$. Using Chernoff bounds ($\mu = 5log_{4/3}n, \delta = 4/5$),

$$\mathbf{Pr}\Big[X < \log_{4/3} n\Big] = \mathbf{Pr}\Big[X < (1-\frac{4}{5})\mu\Big] < e^{-\delta^2\mu/3} \leq 1/n. \tag{6.6.4}$$

■

6.7 Random mate

Theorem 6.7.1 *Probability that random mate removes less than* $\frac{n}{8}$ *elements is* $< e^{(-\frac{n}{32})}$
To be continued...