CSC 226

Algorithms and Data Structures: II

Rich Little
rlittle@uvic.ca

Randomized Algorithms

- When analyzing the ***average-case runtime*** of a deterministic algorithm we determine the runtime over all possible input distributions.
  - In practice this can be difficult and sometimes impossible
- On the other hand we can use randomness as a tool in our algorithms
  - ***Randomized algorithms***
- ***Expected-case runtime*** dependent on randomness in the algorithm as opposed to randomness in the input

Basic Probability

- Section 1.2.4 – Tamassia and Goodrich
- A ***sample space***, S, is the set of all possible outcomes of some experiment.
- **Ex 1:** Flip a coin five times and record.
  - $2^5 = 32$ possible outcomes
  - $S = \{HHHHH, HHHHT, HHHTH, \ldots, TTTTT\}$
- **Ex 2:** Flip a coin until it is tails.
  - Infinite sample space
  - $S = \{T, HT, HHT, HHHT, \ldots\}$

Probability Space

- Each subset A of a sample space S is called an ***event***.
- A ***probability space*** is a sample space, S, with a ***probability distribution*** (***function***), $\Pr: A \subseteq S \rightarrow [0,1]$, such that,
  1. $\Pr(\emptyset) = 0$
  2. $\Pr(S) = 1$
  3. $0 \leq \Pr(A) \leq 1$, for any $A \subseteq S$
  4. If $A, B \subseteq S$ and $A \cap B = \emptyset$, then

$$\Pr(A \cup B) = \Pr(A) + \Pr(B)$$

Properties of Probabilities

- If $A \subseteq B$, then $\Pr(A) \leq \Pr(B)$
- Let $\bar{A} = S - A$, the ***complement***, then $\Pr(\bar{A}) = 1 - \Pr(A)$
- For any $A, B \subseteq S$,

$$\Pr(A \cup B) = \Pr(A) + \Pr(B) - \Pr(A \cap B)$$

Discrete Probability Distributions

- A probability distribution is ***discrete*** if it is defined over a finite or countably infinite sample space.
- For any event A, $\Pr(A) = \sum_{s \in A} \Pr(s)$
  - s are elementary events of S.
- If S is finite, then for every $s \in S$,

$$\Pr(s) = \frac{1}{|S|}$$

- This is a ***uniform probability distribution*** on S.

Independence

- Two events A and B are independent if

$$\Pr(A \cap B) = \Pr(A) \cdot \Pr(B)$$

- A collection of events $\{A_1, \ldots, A_n\}$ is mutually independent if

$$\Pr\left(A_{i_1} \cap \cdots \cap A_{i_k}\right) = \Pr\left(A_{i_1}\right) \cdot \cdots \cdot \Pr\left(A_{i_k}\right)$$

for any subset $\left\{A_{i_1}, \ldots, A_{i_k}\right\}$.

Example 3

- Let S be the experiment of flipping a coin two times.
  - $S = \{HH, HT, TH, TT\}$
- $\Pr(H) = \Pr(T) = 1/2$
- Let A be the event that heads are flipped first, $\mathrm{A} = \{HH, HT\}$. What is $\Pr(A)$?
  - $\Pr(HH) = \Pr(H \cap H) = \Pr(H) \cdot \Pr(H) = 1/4$
  - $\Pr(HT) = \Pr(H \cap T) = \Pr(H) \cdot \Pr(T) = 1/4$
  - $\Pr(A) = \Pr(HH \cup HT)$
    $= \Pr(HH) + \Pr(HT) = 1/2$

Expectation

- A ***random variable*** is a function X that maps outcomes from S to real numbers

$$X: S \to \mathbb{R}$$

- The ***expected value*** of a discrete random variable X is defined as

$$E(X) = \sum_{x} x \cdot \Pr(X = x)$$

for all possible values of X.

- **Theorem – Linearity of Expectation**

Let X and Y be two random variables, then

$E(X + Y) = E(X) + E(Y)$

Example 4

- Flip two coins, let X be the number of heads flipped.
  - $S = \{HH, HT, TH, TT\}$
- Here the possible values of X are 0, 1, and 2
- So,

$$E(X)$$
$$= 0 \cdot \Pr(X = 0) + 1 \cdot \Pr(X = 1) + 2 \cdot \Pr(X = 2)$$
$$= (0)\left({}^{1}\!/_{4}\right) + (1)\left({}^{1}\!/_{2}\right) + (2)\left({}^{1}\!/_{4}\right)$$
$$= 0 + {}^{1}\!/_{2} + {}^{1}\!/_{2} = 1$$

Randomized Quicksort

- Even though the worst case running time of Quicksort is quadratic, in practice Quicksort is a very efficient sorting algorithm.

- Consider the expected running time of "Randomized Quicksort" where the index of the pivot is chosen randomly.

Randomized QuickSort

Algorithm quickSort(S):
Input: Sequence S containing n elements
Output: Sorted sequence S

if S.length() = 1 **then**
    return element of S
Let L, E, G be empty sequences
$p \leftarrow$ pickRandomPivot(S)
partition(L, E, G, S, p)
quickSort(L)
quickSort(G)
concatenate(S,L,E,G)
return S

Randomized Quicksort

- ***Theorem*.** The **expected** running time of randomized Quicksort on a sequence of size n is $O(n \log n)$.
- Randomized means choosing a pivot randomly from the set of elements to be sorted.
- How can we prove this theorem?
- To obtain $O(n \log n)$ **expected** time, we need to split up at least a fraction of n of all the elements.
- Suppose we can show that we can split up ¼ n elements not every time, but every other time we choose a pivot randomly, then we are done.

Random Pivot Selection

- Suppose our set of *n* elements is sorted

- A "good" pivot is one that is in the red range
- How can we choose a pivot from the red range when the array is not sorted yet?
- Let us select a pivot randomly from the input set
- What are the chances that the pivot is in the red range?
  - 50 %
  - Probability ½
  - Basic coin toss
- Thus, every other time we choose a "good pivot" if we choose one randomly

Proof

- Now we have to estimate the height of the recursion tree, given that we we split up at least ¼ elements every other time.
- Suppose that we split up ¼ elements every time

$$\frac{1}{4}|S| \leq |L| \leq \frac{3}{4}|S| \qquad \frac{1}{4}|S| \leq |G| \leq \frac{3}{4}|S|$$

- Then the Quicksort recursion-tree is bounded in height by $\log_{4/3} n$
- Since we only choose a good pivot every other time the Quicksort recursion-tree is bounded in height by $2\log_{4/3} n$

Proof

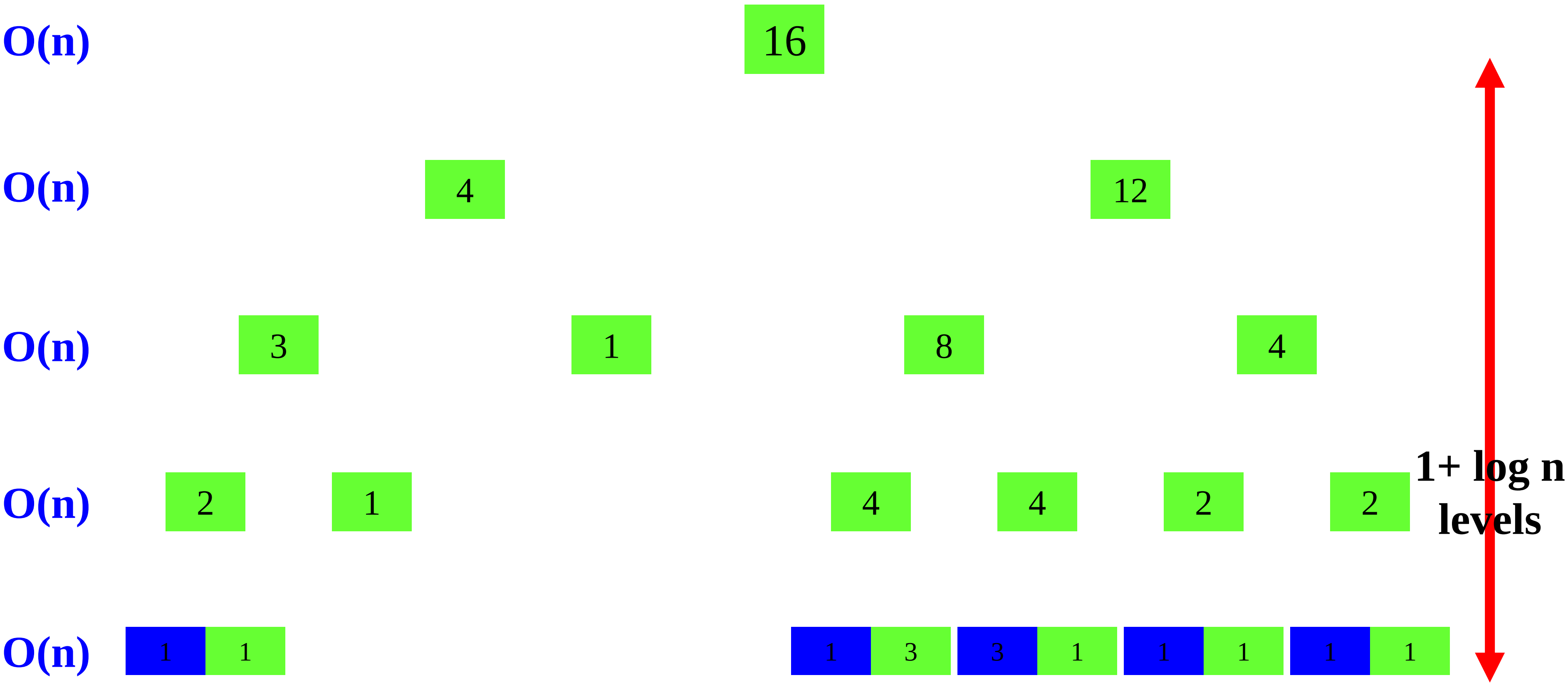
- How many pivots do you have to pick to get $\log_{4/3} n$ good ones? $2\log_{4/3} n$
- What is the probability to pick a good pivot? $\frac{1}{2}$
- How many good pivots exist? $\frac{1}{2}n$

Proof

- Thus the tree has an expected bound in height of $2\log_{4/3} n$

- **Thus, the resulting expected running time for Randomized Quicksort is $O(n \log n)$**

Height of Recursion Tree



$$T(n) = n \log_{4/3} n = n \frac{\log_2 n}{\log_2 4/3} = cn \log_2 n \in O(n \log n)$$

Randomized QuickSelect

Algorithm QuickSelect(S,k):
Input: Sequence S containing n elements, integer $k \leq n$
Output: k^{th} smallest element in sorted sequence S

if S.length() = 1 **then**
    return element of S
Let L, E, G be empty sequences
$p \leftarrow$ pickRandomPivot(S)
partition(L, E, G, S, p)
if $k \leq$ L.length() **then return**
    QuickSelect(L, k)
else if $k \leq L$.length() + E.length() **then**
    return p
else
    return QuickSelect(G, k − L.length() − E.length())

Expected Time Analysis of Randomized QuickSelect

- Reuse the analysis for randomized Quicksort
- We split up ¼ *n* elements every time
- Thus, we have to continue partitioning at most ¾ *n* elements
- Thus, the height of the QuickSelect tree is at most $2\log_{4/3} n$
- How much work do we do at each level?

Expected Time Analysis of Randomized QuickSelect

$$T(n)=\begin{cases} b & \text{if } n=1 \\ cn+T(\frac{3}{4}n) & \text{otherwise} \end{cases}$$

$$T(n)\in O(n)$$

- Show by repeated substitution

Expected Time Analysis of Randomized QuickSelect

$$T(n) = cn + T\left(\frac{3}{4}n\right)$$

$$= cn + \frac{3}{4}cn + T\left(\left(\frac{3}{4}\right)^2 n\right)$$

$$= cn + \frac{3}{4}cn + \left(\frac{3}{4}\right)^2 cn + \cdots + T\left(\left(\frac{3}{4}\right)^i n\right)$$

Here $i = 2\log_{4/3} n$ to get to $T(1)$

$$T(n) = b + \left(\frac{3}{4}\right)^0 cn + \left(\frac{3}{4}\right)^1 cn + \left(\frac{3}{4}\right)^2 cn + \cdots + \left(\frac{3}{4}\right)^{2\log_{4/3} n - 1} cn$$

$$= b + cn\left[\frac{1-\left(\frac{3}{4}\right)^{2\log_{4/3} n}}{1-\frac{3}{4}}\right]$$

$$= b + 4cn\left[1-\left(\frac{3}{4}\right)^{2\log_{4/3} n}\right] \leq b + 4cn \in O(n)$$

Expected Time Analysis of Randomized QuickSelect

- **Theorem.**
  Expected time of Randomized QuickSelect is $O(n)$.

Worst-case Analysis

- **Theorem.**
  The worst-case $T(n)$ of Quicksort is $O(n^2)$.
- **Theorem.**
  The expected-case $T(n)$ of Randomized Quicksort is $O(n \log n)$.
- **Theorem.**
  The expected-case $T(n)$ of Randomized QuickSelect is $O(n)$.
- **Theorem.**
  The worst-case $T(n)$ of QuickSelect is $O(n^2)$.
- **Theorem.**
  The worst-case $T(n)$ of LinearSelect is $O(n)$.