Computer Science 351

Application: Randomly Constructed Binary Search Trees

Instructor: Wayne Eberly

Department of Computer Science
University of Calgary

Lecture #24

Learning Goals

Learning Goals:

- Learn about another application of probability theory to the analysis of data structures and algorithms.
- Applying this is somewhat tricky, in this case, because the identification of a ***sample space***, that allows the analysis to be carried out, is somewhat challenging.

Note: It is possible that students will also see a version of this material in CPSC 331.

Randomly Constructed Binary Search Trees

- Let n be a positive integer.
- Consider the binary search trees, with size n, storing the integers $1, 2, \ldots, n$.
- Since these integers can be inserted into an initially empty binary search tree in any order, this experiment can be modelled using a ***sample space***, Ω_n, which includes all ***permutations***

$$(\alpha_1, \alpha_2, \ldots, \alpha_n)$$

  of the set of integers between 1 and n — listing the order in which these integers are inserted.
- It follows that $|\Omega_n| = n!$

One Probability Distribution...

- It is ***assumed*** that all permutations are equally — so that the ***uniform probability distribution***

$$\mathrm{P}_1 : \Omega_n \to \mathbb{R}$$

is used. Then

$$\mathrm{P}_1(\sigma) = \frac{1}{|\Omega_n|} = \frac{1}{n!}$$

for every outcome $\sigma \in \Omega_n$.

... and Another Probability Distribution...

Consider *another* probability distribution

$$\mathrm{P}_2 : \Omega_n \to \mathbb{R}$$

such that, for each element $\sigma \in \Omega_n$, $\mathrm{P}_2(\sigma)$ is the probability that σ is returned by an execution of the following.

1. Choose an integer i such that $1 \leq i \leq n$ — choosing each with probability $\frac{1}{n}$.
2. Choose a permutation

   $$\mu = (\beta_1, \beta_2, \ldots, \beta_{n-1})$$

   uniformly from Ω_{n-1} — so that each permutation, μ, is chosen with probability $\frac{1}{|\Omega_{n-1}|} = \frac{1}{(n-1)!}$.

... and Another Probability Distribution...

3. For $1 \leq j \leq n$, let

$$\gamma_j = \begin{cases} \beta_j & \text{if } 1 \leq \beta_j \leq i-1, \\ \beta_j + 1 & \text{if } i \leq \beta_j \leq n-1, \end{cases}$$

   so that $(\gamma_1, \gamma_2, \ldots, \gamma_{n-1})$ includes the numbers

$$1, 2, \ldots, i, i+1, i+2, \ldots, n \\ = \{j \in \mathbb{N} \mid 1 \leq j \leq n \text{ and } j \neq i\}$$

   in some order (with each of the integers in this set).

4. Return the permutation

$$(i, \gamma_1, \gamma_2, \ldots, \gamma_{n-1}).$$

... and Another Probability Distribution...

- Every permutation $\sigma \in \Omega_n$ corresponds to exactly *one* choice of the integer *i*, at line 1, and exactly *one* choice of the permutation, $\mu \in \Omega_{n-1}$, at line 2,
- This can be used to show that

$$\mathrm{P}_2(\sigma) = \mathrm{P}_1(\sigma) = \frac{1}{n!}$$

for every permutation $\sigma \in \Omega_n$ — so that the probability distributions, P_1 and P_2, are the same.

... and Yet Another Probability Distribution...

Consider *yet another* probability distribution

$$\mathrm{P}_3 : \Omega_n \rightarrow \mathbb{R}$$

such that, for each element $\sigma \in \Omega_n$, $\mathrm{P}_3(\sigma)$ is the probability that σ is returned by an execution of the following.

1. Choose an integer i such that $1 \leq i \leq n$ — choosing each with probability $\frac{1}{n}$.
2. Choose a subset S_L of the set of integers $2, 3, \ldots, n$ with size $i - 1$ — choosing every such subset with the same probability, $\binom{n-1}{i-1}^{-1} = \frac{(i-1)! \times (n-i)!}{(n-1)!}$.
3. Set S_R to be the set of integers between 2 and $n - 1$ that do not belong to S_L — so that S_R is a set, with size $n - i$, such that $S_L \cap S_R = \emptyset$ and $S_L \cup S_R = \{2, 3, \ldots, n\}$.

... and Yet Another Probability Distribution...

4. Choose a permutation

$$\mu = (\beta_1, \beta_2, \ldots, \beta_{i-1})$$

uniformly from Ω_{i-1} — so that each permutation, μ, is chosen with probability $\frac{1}{|\Omega_{i-1}|} = \frac{1}{(i-1)!}$.

Note: μ is a sequence with length zero if $i = 1$, so that $i - 1 = 0$.

5. Choose a permutation

$$\nu = (\gamma_1, \gamma_2, \ldots, \gamma_{n-i})$$

uniformly from Ω_{n-i} — so that each permutation, μ, is chosen with probability $\frac{1}{|\Omega_{n-i}|} = \frac{1}{(n-i)!}$.

Note: μ is a sequence with length zero if $i = n$, so that $n - i = 0$.

... and Yet Another Probability Distribution...

Suppose, now, that

$$S_L = \{k_1, k_2, \ldots, k_{i-1}\} \quad \text{and} \quad S_R = \{\ell_1, \ell_2, \ldots, \ell_{n-i}\}$$

where

$$k_1 < k_2 < \cdots < k_{i-1} \quad \text{and} \quad \ell_1 < \ell_2 < \cdots < \ell_{n-i}.$$

6. Return the permutation $(\alpha_1, \alpha_2, \ldots, \alpha_n) \in \Omega_n$ that is defined as follows:
   - $\alpha_1 = i$.
   - If $2 \leq j \leq n$ and $j \in S_L$ — so that $j = k_h$, for an integer h such that $1 \leq h \leq i-1$, then $\alpha_j = \beta_h$ (for β_h as given at line 4, above).
   - If $2 \leq j \leq n$ and $j \in S_R$ — so that $j = \ell_h$, for an integer h such that $1 \leq h \leq n-i$, then $\alpha_j = n - i + \gamma_h$ (for γ_h as given at line 5, above).

... and Yet Another Probability Distribution...

- Every permutation $\sigma \in \Sigma_n$ corresponds to exactly *one* choice of the integer i at line 1, exactly *one* choice of the subset S_L at line 2, exactly *one* choice of the permutation $\mu \in \Sigma_{i-1}$ at line 4, and exactly *one* choice of the permutation $\mu \in \Sigma_{n-i}$ at line 5.
- This can be used to show that

$$\mathrm{P}_3(\sigma) = \mathrm{P}_2(\sigma) = \mathrm{P}_1(\sigma) = \frac{1}{n!}$$

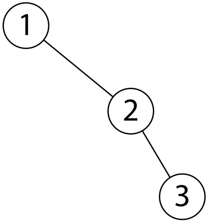
  for every permutation $\sigma \in \Omega_n$ — so that the probability distribution P_3 is the same as the probability distributions P_1 and P_2.

Random Variables of Interest

- For each permutation $\sigma \in \Omega_n$, let T_σ be the binary search tree, storing $1, 2, \ldots, n$, obtained by storing integers into an initially empty binary search tree — in the order given by σ.
- Let $d : \Omega_n \to \mathbb{R}$ such that, for $\sigma \in \Omega_n$, $d(\sigma)$ is the ***depth*** of the binary search tree T_σ.
- Let $xd : \Omega_n \to \mathbb{R}$ such, that, for $\sigma \in \Omega_n$, $xd(\sigma) = 2^{d(\sigma)}$.
- The value of these random variables are shown, for the case that $n = 3$, on the following slides.

Random Variables of Interest

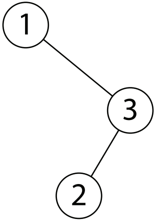
$\sigma = (1, 2, 3)$:



$d(\sigma) = 2$ and $xd(\sigma) = 2^2 = 4$.

Random Variables of Interest

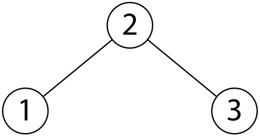
$\sigma = (1, 3, 2)$:



$d(\sigma) = 2$ and $xd(\sigma) = 2^2 = 4$.

Random Variables of Interest

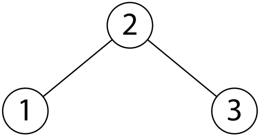
$\sigma = (2, 1, 3)$:



$d(\sigma) = 1$ and $xd(\sigma) = 2^1 = 2$.

Random Variables of Interest

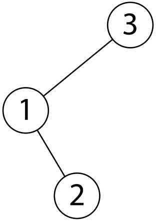
$\sigma = (2, 3, 1)$:



$d(\sigma) = 1$ and $xd(\sigma) = 2^1 = 2$.

Random Variables of Interest

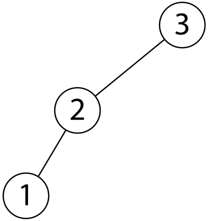
$\sigma = (3, 1, 2)$:



$d(\sigma) = 2$ and $xd(\sigma) = 2^2 = 4$.

Random Variables of Interest

$\sigma = (3, 2, 1)$:



$d(\sigma) = 2$ and $xd(\sigma) = 2^2 = 4$.

Random Variables of Interest

It follows, by the above, that if $n = 3$ then

$$\begin{aligned}
\mathrm{E}[d] &= \sum_{\sigma \in \Sigma_3} d(\sigma) \times \mathrm{P}(\sigma) \\
&= d((1,2,3)) \times \mathrm{P}((1,2,3)) + d((1,3,2)) \times \mathrm{P}((1,3,2)) \\
&\quad + d((2,1,3)) \times \mathrm{P}((2,1,3)) + d((2,3,1)) \times \mathrm{P}((2,3,1)) \\
&\quad + d((3,1,2)) \times \mathrm{P}((3,1,2)) + d((3,2,1)) \times \mathrm{P}((3,2,1)) \\
&= 2 \times \tfrac{1}{6} + 2 \times \tfrac{1}{6} + 1 \times \tfrac{1}{6} + 1 \times \tfrac{1}{6} + 2 \times \tfrac{1}{6} + 2 \times \tfrac{1}{6} \\
&= \tfrac{10}{6} = \tfrac{5}{3}.
\end{aligned}$$

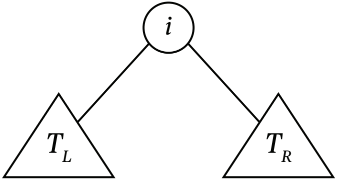
Random Variables of Interest

It also follows, by the above, that if $n = 3$ then

$$
\begin{aligned}
\mathrm{E}[xd] &= \sum_{\sigma \in \Sigma_3} xd(\sigma) \times \mathrm{P}(\sigma) \\
&= xd((1,2,3)) \times \mathrm{P}((1,2,3)) + xd((1,3,2)) \times \mathrm{P}((1,3,2)) \\
&\quad + xd((2,1,3)) \times \mathrm{P}((2,1,3)) + xd((2,3,1)) \times \mathrm{P}((2,3,1)) \\
&\quad + xd((3,1,2)) \times \mathrm{P}((3,1,2)) + xd((3,2,1)) \times \mathrm{P}((3,2,1)) \\
&= 4 \times \tfrac{1}{6} + 4 \times \tfrac{1}{6} + 2 \times \tfrac{1}{6} + 2 \times \tfrac{1}{6} + 4 \times \tfrac{1}{6} + 4 \times \tfrac{1}{6} \\
&= \tfrac{20}{6} = \tfrac{10}{3}.
\end{aligned}
$$

A Recurrence for a Bound

Now let i be an integer such that $1 \leq i \leq n$. Let T_L be a binary search tree storing the integers $1, 2, \ldots, i-1$ and let T_R storing the integers $i+1, i+2, \ldots, n$ — so that one of the binary search trees that stores the integers $1, 2, \ldots, n$ is the binary search tree T that has i at the root, with left subtree T_L and right subtree T_R:



Let $\widehat{T}_R$ be the binary search tree produced by subtracting i from each of the integers stored at nodes — so that $\widehat{T}_R$ stores the integers $1, 2, \ldots, n-i$.

A Recurrence for a Bound

Consider the following values.

- s: The number of permutations in Σ_n that would produce T.
- s_L: The number of permutations in Σ_i that would produce T_L.
- s_R: The number of permutations in Σ_{n-i} that would produce $\widehat{T}_R$.

A Recurrence for a Bound

- p: Probability that T is generated when using the described experiment to produce a binary search tree storing $1, 2, \ldots, n$.
- p_L: Probability that T_L is generated when using the described experiment to produce a binary search tree storing $1, 2, \ldots, i$.
- p_R: Probability that T_R is generated when using the described experiment to produce a binary search tree storing $1, 2, \ldots, n - i$.

A Recurrence for a Bound

Since the ***uniform probability distribution*** is being used in this case,

$$p = \frac{s}{|\Omega_n|} = \frac{s}{n!},$$

$$p_L = \frac{S_L}{|\Omega_i|} = \frac{s_L}{i!},$$

and

$$p_R = \frac{s_R}{|\Omega_{n-i}|} = \frac{s_R}{(n-i)!}.$$

A Recurrence for a Bound

In order to compute s, note the following.

- There is *one* way to choose the first element in an outcome (from Ω_n) — this must always be i, so that i is at the root of the binary search tree that is generated.
- There are exactly $\binom{n-1}{i-1}$ ways to choose the other locations (for the ordering of $1, 2, \ldots, n$ being generated) of integers between 1 and i.
- For each of these, there are (by definition) s_L ways to choose the values placed in these locations, in order for the left subtree generated to be T_L.
- For each of these, there are s_R ways to choose the values placed in the remaining locations, in order for the right subtree to be T_R.

A Recurrence for a Bound

It follows that $s = \binom{n-1}{i-1} \times s_L \times s_R$, so that

$$\begin{aligned} p &= \frac{s}{|\Omega_n|} \\ &= \frac{\binom{n-1}{i-1} \times s_L \times s_R}{n!} \\ &= \frac{\frac{(n-1)!}{(i-1)! \times (n-i)!} \times s_L \times s_R}{n \times (n-1)!} \\ &= \frac{1}{n} \times \frac{s_L}{(i-1)!} \times \frac{s_R}{(n-i)!} \\ &= \frac{1}{n} \times \frac{s_L}{|\Omega_{i-1}|} \times \frac{s_R}{|\Omega_{n-i}|} \\ &= \frac{1}{n} \times p_L \times p_R. \end{aligned}$$

A Recurrence for a Bound

Now, for $i \geq 1$, let $xd_i : \Omega_i \to \mathbb{R}$ be the random variable, defined for the sample space Ω_i, whose value is the exponential depth of the binary search tree (storing the integers $1, 2, \ldots, i$) generated using whatever outcome, from Ω_i, that is being considered.

- It follows by the analysis given above (in which binary search trees storing the integers 1, 2 and 3 were considered) that $xd_3 = \frac{10}{3}$.
- Let us “define” xd_0 to be 0. This will not really change anything, but it will make it easier to produce general formulas for some of what we want to consider.

A Recurrence for a Bound

Suppose n is a positive integer. Consider another sequence of random variables $xd_{n,1}, xd_{n,2}, \ldots, xd_{n,n}$ such that, for every integer i such that $1 \leq i \leq n$ and for every outcome

$$\sigma = (j_1, j_2, \ldots, j_n) \in \Omega_n,$$

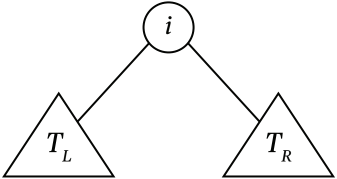
$$xd_{n,i}(\sigma) = \begin{cases} xd_n(\sigma) & \text{if } j_1 = i, \\ 0 & \text{if } j_1 \neq i. \end{cases}$$

Then, for $n \geq 2$, $xd_{n,i}(\sigma) = xd_n(\sigma) > 0$ if and only if i is stored at the root of the binary search tree constructed using insertion order σ — and

$$xd_n = xd_{n,1} + xd_{n,2} + \cdots + xd_{n,n}.$$

A Recurrence for a Bound

Consider, again, a binary search tree T with the form



Once again, let $\widehat{T}_R$ be the binary search tree produced by subtracting i from each of the integers stored at nodes — so that $\widehat{T}_R$ stores the integers $1, 2, \ldots, n - i$.

A Recurrence for a Bound

If the binary search trees T, T_L and $\widehat{T}_R$ have depths d, d_L and d_R respectively, then

$$d = \max(d_L, d_R) + 1.$$

Thus if the *exponential depths* of these trees are $xd = 2^d$, $xd_L = 2^{d_L}$ and $xd_R = 2^{d_R}$, respectively, then

$$\begin{aligned} xd &= 2^d \\ &= 2^{\max(d_L, d_R)+1} \\ &= 2 \times 2^{\max(d_L, d_R)} \\ &= 2 \times \max(2^{d_L}, 2^{d_R}) \\ &= 2 \times \max(xd_L, xd_R) \\ &\leq 2 \times (xd_L + xd_R). \end{aligned}$$

A Recurrence for a Bound

Recall, as well, that if p, p_L and p_R are the probabilities that T, T_L and $\widehat{T}_R$ are obtained (when randomly producing binary search trees with sizes n, $i-1$ and $n-i$, respectively) then

$$p = \frac{1}{n} \times p_L \times p_R.$$

These equations can be applied to establish that

$$\mathrm{E}[xd_{n,i}] = \frac{2}{n} \times (\mathrm{E}[xd_{i-1}] + \mathrm{E}[xd_{n-i}]).$$

Exercise: Establish this bound.

A Recurrence for a Bound

Now, since $xd_n = xd_{n,1} + xd_{n,2} + \cdots + xd_{n,n}$, it follows that

$$\begin{aligned}
\mathrm{E}[xd_n] &= \mathrm{E}\left[\sum_{i=1}^{n} xd_{n,i}\right] \\
&= \sum_{i=1}^{n} \mathrm{E}[xd_{n,i}] \qquad \text{(by Linearity of Expectation)} \\
&\le \sum_{i=1}^{n} \left(\frac{2}{n} \times (\mathrm{E}[xd_{i-1}] + \mathrm{E}[xd_{n-i}]\right) \\
&= \frac{4}{n} \sum_{i=0}^{n-1} \mathrm{E}[xd_i].
\end{aligned}$$

A Recurrence for a Bound

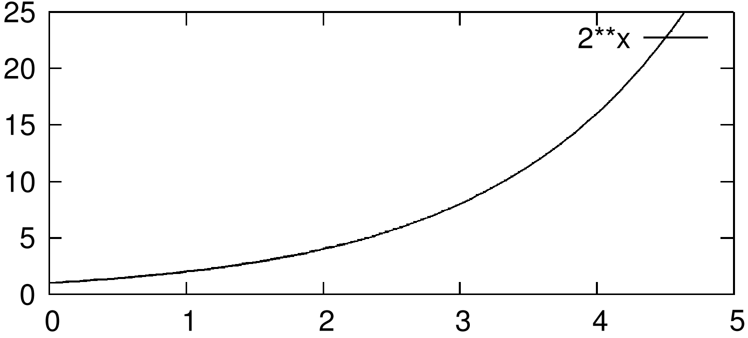
The above inequality can be used to prove — by induction on n — that

$$\mathrm{E}[xd_n] \leq \frac{1}{4}\binom{n+3}{3} \leq n^3$$

for every integer n such that $n \geq 2$.

Bounding Expected Depth

Consider the function $f(x) = 2^x$.



Bounding Expected Depth

This function is **convex**: If $\alpha \geq 0$, $\beta \geq 0$, and $\alpha + \beta = 1$ then

$$f(\alpha x_1 + \beta x_2) \leq \alpha f(x_1) + \beta f(x_2)$$

for real numbers x_1 and x_2 such that $x_2 > x_1 \geq 0$. This can be used to prove the following.

> ***Theorem (Jensen's Inequality):*** If f is a convex function then, for every integer $m \geq 1$ and for all positive values $x_1, x_2, \ldots, x_m$,
>
> $$f\left(\tfrac{1}{m}(x_1 + x_2 + \cdots + x_m)\right) \leq \frac{1}{m}\left(f(x_1) + f(x_2) + \cdots + f(x_m)\right).$$

Bounding Expected Depth

Applying this, with $m = |\Omega_n|$,

$$\Omega_n = \{\sigma_1, \sigma_2, \dots, \sigma_m\}$$

(for some ordering of this set) and $x_i = d_n(\sigma_i)$ for $1 \le i \le m$, we obtain the inequality

$$2^{\mathrm{E}[d_n]} \le \mathrm{E}[xd_n] \le n^3$$

which implies that

$$\mathrm{E}[d_n] \le 3 \log_2 n.$$

This — if a binary search tree with size n by starting with an empty tree and inserting keys, using a “uniformly and randomly chosen” insertion order, then the expected value of the depth of the resulting tree is at most $3 \log_2 n$.

Tail Bounds

Suppose, now, that k is a positive integer and consider a binary search tree, with size n, that is "randomly" generated as described above.

- The depth of this tree is greater than or equal to $3\log_2 n + k$ if and only if the *exponential depth* of this tree is greater than or equal to $2^k \times n^3 \geq 2^k \times \mathrm{E}[xd_n]$.
- *Markov's Inequality* can be applied to show that the probability of this is at most 2^{-k}.
- Thus the probability that a randomly constructed binary search has a depth, that is significantly larger than $3\log_2 n$, is very small.

Remember That Assumption!

Please note that, like every other "average case analysis", *this analysis depends on an assumption that might not be satisfied*.

- In this case the assumption concerns how binary search trees with size n are generated (which is used to obtain an assumption about the shapes of these trees).
- If the assumption is not satisfied then, while the analysis is still technically "correct", it might also be completely ***irrelevant*** — and the depts of binary search trees seen, under whatever circumstances you are considering, might be very different than what this analysis suggests.