Exercise sheet 6.

Advanced Algorithms WiSe 2020/21

Instructor: László Kozma

Due 12:00, December 18th, 2020

Exercise 1 Splay trees *2+4+4 Points*

(a) Try to get some familiarity with the behavior of splay trees. Draw a binary search tree with 5-10 nodes and work out a few splay operations on paper. You can also try some interactive demonstrations of splay tree on the internet. Try to force splay to make costly operations. What happens?

(b) A possible intuition for the efficiency of splay trees is that when searching for x, the depths of *most* nodes on the search path of x are reduced. Prove a statement that makes this intuition precise, showing that if a node on the search path has depth d before splaying, then it has depth at most $a \cdot d + b$ after splaying, for suitable constants $a, b > 0$.

Hint: consider a node at depth d on the search path and see how it is affected by all the zig-zig, zig-zag, and zig operations.

(c) Consider a "simpler version" of splay, called move-to-root. In move-to-root, instead of the zig-zig and zig-zag operations, we simply rotate the accessed element up using normal rotations until it becomes the root.

Show that move-to-root can be very inefficient: construct an initial tree and an arbitrarily long search sequence that has high cost per search if move-to-root is used instead of splay. Is the choice of initial tree essential in your example?

Exercise 2 Hashing *4 × 4 Points*

Recall the definition from class: a family $\mathcal{H}$ of hash functions $U \to T$ (where $|T| = n$, and $|U| = m$) is 2-*universal*, if for all $x, y \in U$, $x \neq y$,

$$\mathrm{Pr}_{h\in\mathcal{H}}\Big[h(x) = h(y)\Big] \leq \frac{1}{n}.$$

(a) Let $U = \{a, b, c, d, e\}$. Give an explicit 2-universal family of four hash functions $U \to \{0, 1\}$ (write the hash family as a 4×5 table with entries 0 and 1), and verify that the family is 2-universal, using the definition. Give an example of the same size that is not universal.

(b) A counter-intuitive aspect of hash families is that they can be quite good even if some of the individual hash functions they contain are bad. Give a small example of a 2-universal family $\mathcal{H}$ of functions $U \to T$, say with $|U| = |\mathcal{H}| = 4$, $|T| = 2$, such that $\mathcal{H}$ contains the all-zero function $h(x) = 0$.

(c) A family $\mathcal{H}$ of hash functions is 2-*independent*, if for all $x, y \in U$, $x \neq y$, and all $a, b \in T$,

$$\mathrm{Pr}_{h\in\mathcal{H}}\Big[h(x) = a \quad\text{and}\quad h(y) = b\Big] \leq \frac{1}{n^2}.$$

(Intuitively, on every *pair* of items in U the function h looks like a perfectly random function; observe that "$\leq$" could be replaced by "$=$". Do you see why?)

Show that independence is stronger than universality, in the following sense: if $\mathcal{H}$ is 2-independent, then it is 2-universal.

(d) Show that the converse is not true by constructing a small example (say $U = \{a, b, c, d\}$, $T = \{0, 1\}$, and $|\mathcal{H}| = 4$) that is 2-universal, but not 2-independent.

(e) (4 bonus points) Show that the definition of 2-universality cannot be strengthened significantly, in the following sense: for every family $\mathcal{H}$ of hash functions, there are values $x, y \in U$ such that

$$\mathrm{Pr}_{h\in\mathcal{H}}\Big[h(x) = h(y)\Big] \geq \frac{1}{n} - \frac{1}{m}.$$

Total: 26 points. Have fun with the solutions!

From: Yumeng Li and Thore Brehmer

Exercise 1 Splay trees *2+4+4 Points*

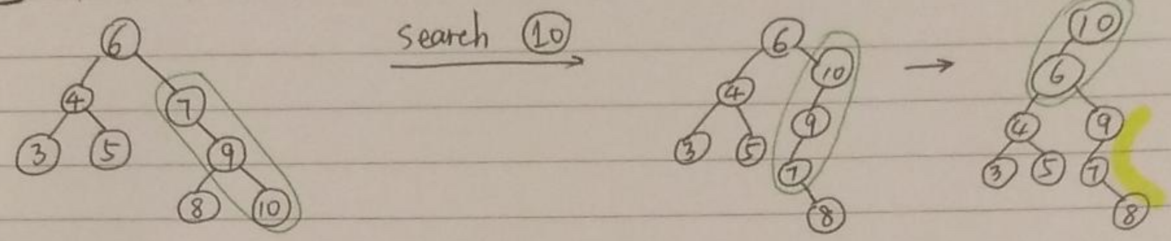
(a) Try to get some familiarity with the behavior of splay trees. Draw a binary search tree with 5-10 nodes and work out a few splay operations on paper. You can also try some interactive demonstrations of splay tree on the internet. Try to force splay to make costly operations. What happens?

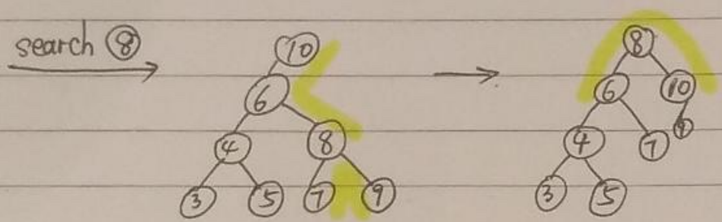
(b) A possible intuition for the efficiency of splay trees is that when searching for x, the depths of *most* nodes on the search path of x are reduced. Prove a statement that makes this intuition precise, showing that if a node on the search path has depth d before splaying, then it has depth at most $a \cdot d + b$ after splaying, for suitable constants $a, b > 0$.

Hint: consider a node at depth d on the search path and see how it is affected by all the zig-zig, zig-zag, and zig operations.

NO.
DATE L6.

1a).

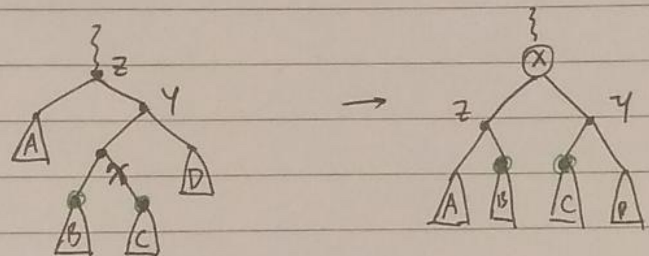




1b).

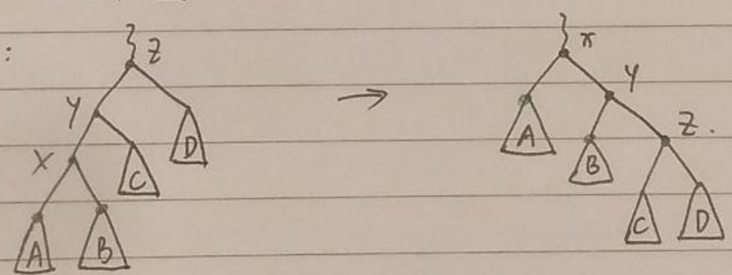
$a = \frac{1}{2}$
$b = 3$.
i.e. $d' \le \frac{d}{2} + 3$

zigzag:



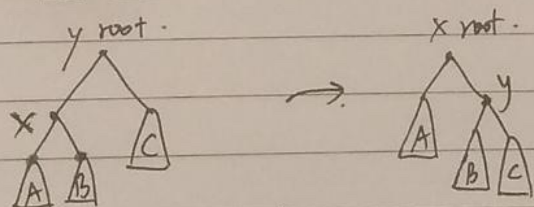
depth: $d_x \to d_x - 2$.
x_children depth -1
$d_y \to d_y$
$d_z \to d_z + 1$.

zigzig:



$d_x \to d_x - 2$
x_childre depth reduce at least 1.
$d_y \to d_y$.
$d_z \to d_z + 2$.

zig:



$d_x \to d_x - 1$
$d_y \to d_y + 1$.
x_children ~~remains~~ don't increase depth.

$\#\{\text{zigzag / zigzig}\} = \lfloor \frac{d_x}{2} \rfloor$
notice that each node below x reduces its depth by at least 1 in a zigzig / zigzag operation.

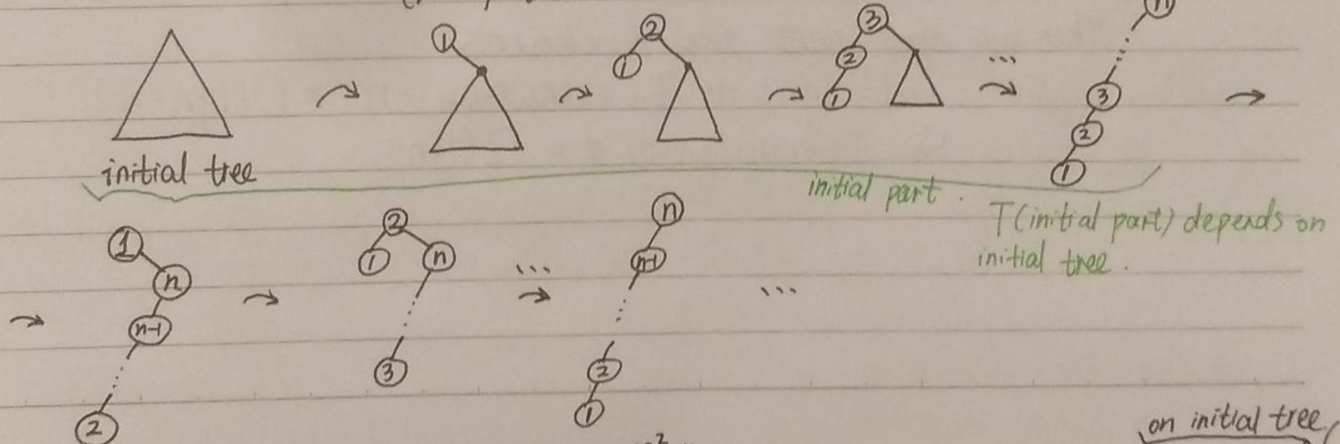
After zigzag / zigzig involving x and y, y will be below x,
So $\#\{\text{zigzag/zigzig that } y \text{ is below } x\}$ is $\lfloor \frac{d_y}{2} \rfloor$, in the end, depth of y would be $\le d_y - \lfloor \frac{d_y}{2} \rfloor \le \frac{d_y}{2} + 1$.

start: $d_z = d_x - 2$. after zigzig/zigzag involving x and z, z will be below x.
So $\#\{\text{zigzag/zigzig that } z \text{ is below } x\}$ is $\lfloor \frac{d_z}{2} \rfloor$, in the end depth of z would be less than $d_z + 2 - \lfloor \frac{d_z}{2} \rfloor \le \frac{d_z}{2} + 3$. . since in zig, all children of x remain their depth, proof complete.

(c) Consider a “simpler version” of splay, called move-to-root. In move-to-root, instead of the zig-zig and zig-zag operations, we simply rotate the accessed element up using normal rotations until it becomes the root.

Show that move-to-root can be very inefficient: construct an initial tree and an arbitrarily long search sequence that has high cost per search if move-to-root is used instead of splay. Is the choice of initial tree essential in your example?

c). sequence: $\underbrace{1, 2, \cdots, n, 1, 2, \cdots, n, 1, 2, \cdots, n, \cdots\cdots}_{(1+m)\text{- times}}$



Cost: $T(\text{initial part}) + m \cdot \frac{n^2-n}{2}$ ⇒ cost per operation is $O(n)$, doesn't depend on initial tree

Exercise 2 Hashing *4 × 4 Points*

Recall the definition from class: a family $\mathcal{H}$ of hash functions $U \to T$ (where $|T| = n$, and $|U| = m$) is 2-*universal*, if for all $x, y \in U$, $x \neq y$,

$$\Pr_{h\in\mathcal{H}}\Big[h(x) = h(y)\Big] \leq \frac{1}{n}.$$

(a) Let $U = \{a, b, c, d, e\}$. Give an explicit 2-universal family of four hash functions $U \to \{0, 1\}$ (write the hash family as a 4×5 table with entries 0 and 1), and verify that the family is 2-universal, using the definition. Give an example of the same size that is not universal.

a) $U = \{a, b, c, d, e\}$ $\mathcal{H} = \{h_1, h_2, h_3, h_4\}$ $T = \{0, 1\}$

e.g. 1

	a	b	c	d	e
h_1	0	0	1	1	1
h_2	1	1	0	0	1
h_3	0	1	1	0	0
h_4	1	0	0	1	0

$$Pr[h(a) = h(b)] = \underset{(h_2)}{\frac{1}{4}} + \underset{(h_3)}{\frac{1}{4}} = \frac{1}{2}$$

$Pr[h(a) = h(c)] = 0$

$Pr[h(a) = h(d)] = \frac{1}{4}$

$Pr[h(a) = h(e)] = \frac{1}{2}$

$Pr[h(b) = h(c)] = \frac{1}{4}$

$Pr[h(b) = h(d)] = 0$

$Pr[h(b) = h(e)] = \frac{1}{2}$

$Pr[h(c) = h(d)] = \frac{1}{2}$

$Pr[h(c) = h(e)] = \frac{1}{2}$

$Pr[h(d) = h(e)] = \frac{1}{2}$

$\Rightarrow$ 2-universal

e.g. 2

	a	b	c	d	e
h_1	0	0	0	0	0
h_2	0	0	0	0	1
h_3	0	0	0	1	0
h_4	0	0	1	0	0

$Pr[h(a) = h(b)] = 1$

$\Rightarrow$ not 2-universal

(b) A counter-intuitive aspect of hash families is that they can be quite good even if some of the individual hash functions they contain are bad. Give a small example of a 2-universal family $\mathcal{H}$ of functions $U \to T$, say with $|U| = |\mathcal{H}| = 4$, $|T| = 2$, such that $\mathcal{H}$ contains the all-zero function $h(x) = 0$.

b) $U = \{a, b, c, d\}$ $\mathcal{H} = \{h, h_2, h_3, h_4\}$ $T = \{0, 1\}$

e.g.

	a	b	c	d
h	0	0	0	0
h_2	0	0	1	1
h_3	0	1	1	0
h_4	1	0	1	0

$$Pr[h(a) = h(b)] = \underset{(h)}{\frac{1}{4}} + \underset{(h_3)}{\frac{1}{4}} = \frac{1}{2}$$

$$Pr[h(a) = h(c)] = \frac{1}{2}$$

$$Pr[h(a) = h(d)] = \frac{1}{2}$$

$$Pr[h(b) = h(c)] = \frac{1}{2}$$

$$Pr[h(b) = h(d)] = \frac{1}{2}$$

$$Pr[h(c) = h(d)] = \frac{1}{2}$$

$\Rightarrow$ 2-universal

(c) A family $\mathcal{H}$ of hash functions is 2-*independent*, if for all $x, y \in U$, $x \neq y$, and $\forall$ $a, b \in T$,

$$\Pr_{h\in\mathcal{H}}\Big[h(x) = a \text{ and } h(y) = b\Big] \leq \frac{1}{n^2}. \qquad n = |T|$$

(Intuitively, on every *pair* of items in U the function h looks like a perfectly random function; observe that "$\leq$" could be replaced by "$=$". Do you see why?)

Show that independence is stronger than universality, in the following sense: if $\mathcal{H}$ is 2-independent, then it is 2-universal.

$\forall a, b \in T \quad h(x), h(y) \in T \quad \forall x, y \in U, \ x \neq y$

Show: $h(x) = a \wedge h(y) = b \overset{?}{\Rightarrow} h(x) = h(y)$

for $a = b$

$\hookrightarrow h(x) = b \wedge h(y) = b \Rightarrow h(x) = h(y)$

$h(y) = h(x) \Rightarrow h(x) = h(y)$

$\square$

(d) Show that the converse is not true by constructing a small example (say $U = \{a, b, c, d\}$, $T = \{0, 1\}$, and $|\mathcal{H}| = 4$) that is 2-universal, but not 2-independent.

$U = \{a, b, c, d\}$ $\quad \mathcal{H} = \{h_1, h_2, h_3, h_4\}$ $\quad T = \{0, 1\}$

$Pr_i :=$ Pr for 2-independent

$Pr_u :=$ Pr for 2-universal

$h(x) = \alpha \wedge h(y) = b$

e.g. From b)

	a	b	c	d
h_1	0	0	0	0
h_2	0	0	1	1
h_3	0	1	1	0
h_4	1	0	1	0

$\Rightarrow$ 2-universal

$Pr_i[h(a) = 0 \wedge h(b) = 0] = \underset{(h_1)}{\frac{1}{4}} + \underset{(h_2)}{\frac{1}{4}} = \frac{1}{2}$

$Pr_i[h(a) = 0 \wedge h(b) = 1] = \frac{1}{4}$

$Pr_i[h(a) = 1 \wedge h(b) = 0] = \frac{1}{4}$

$Pr_i[h(a) = 1 \wedge h(b) = 1] = 0$

$\vdots$

$\Rightarrow$ not 2-independent