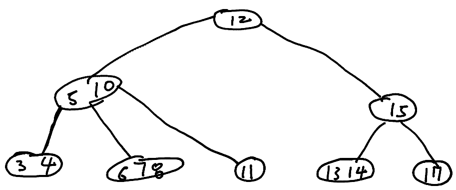
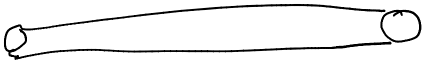
# Augmented data structures

2017년 10월 25일 수요일
오후 7:52



Given a communication channel,
process a sequence of request.

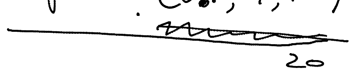
. let a request be represented as
tuple $(x, t_1, t_2)$ where
$x$ $(0 < x \le 1)$ is portion of
channel from time $t_1$ to $t_2$ $0 \le t_1 \le t_2$

Algo structure

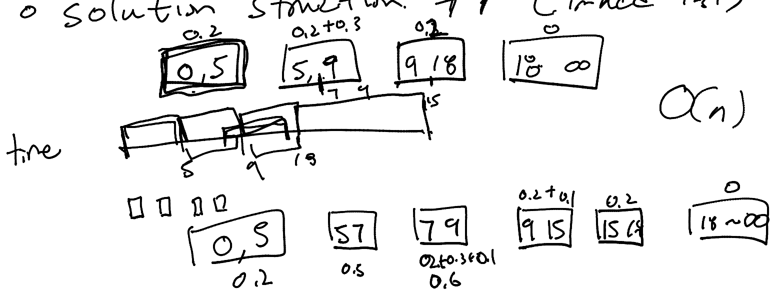
<u>if</u> portion $x$ is available from $t_1$ to $t_2$
<u>then</u> schedule the request
<u>else</u> respond with "no"
<u>endif</u>

o example
① ( 0.2 , 0 , 18 )
② ( 0.3 , 5 , 9 )

request (0.1, 7, 15)




o solution structure #1 (linked list)

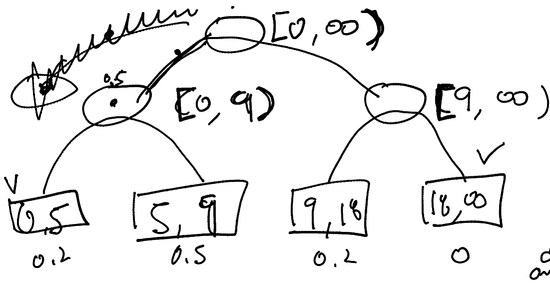


$O(n)$

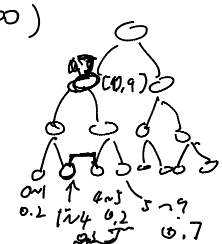
2nd solution structure

2-3-4 trees

$\Rightarrow$ leaves hold unoverlapping time intervals




$O(n)$




(4, 6) 0.6 (0.6 ~ 20)

(0.6, 3~5)

* step on Augmented DS

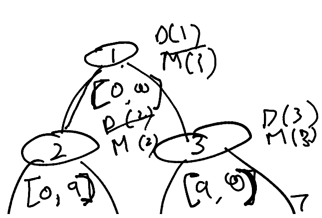
① Choose an underlying DS

② Determine additional information to maintain in the underlying DS

③ Verify that we can maintain the additional info for the basic modified operations the under DS

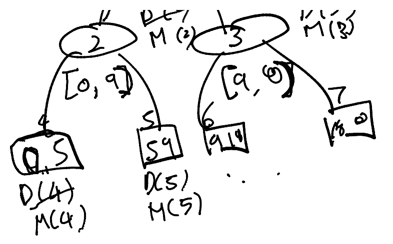
④ Develop new operations

Idea : add infor x to certain ~~intermediate~~ interval nodes that cover the req. interval.




① add to each node v field

D(v) : delta ($\Delta$) 등 change




D(v) : delta (Δ) ≡ change
M(v) : Max amount of capacity used at any time for v's interval

- leaves $[t', t'')$

1) true portion of channel in use during time $[t', t'')$ is

$$\sum_{u \in P} D(u)$$

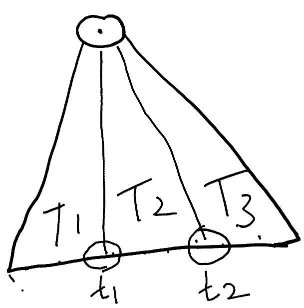
P is path from root to the leaf

2) $M(v) = \max_{P} \sum_{u \in P} D(u)$

where P is ~~the~~ any path from v down to leaf.

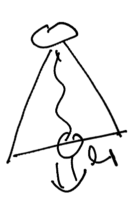
max $\left\{ \sum_{u \in P_1} D(u), \; \sum_{u \in P_2}, \; \sum_{u \in P_3} \right.$

- test request processing
$(x, \underline{t_1}, \underline{t_2})$



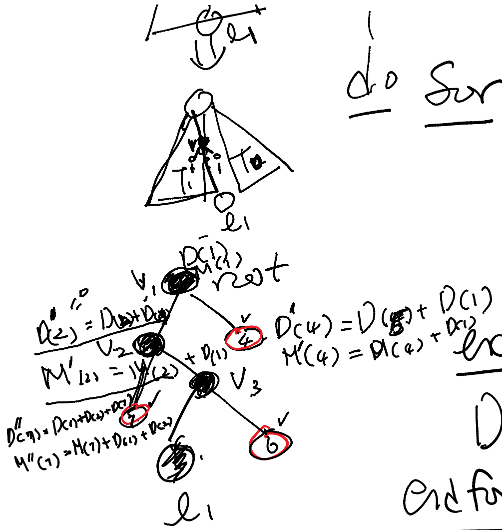

1) find leaf $l_1$ that can contain $t_1$ in it's interval

2) split tree into $T_1, l_1, T_0$ as follows



for each node $v \neq l_1$ on the path from root to $l_1$ in order starting from root.

do for each child w of v do




do for each child $u$ of $v$ do

$D(u) \leftarrow D(u) + D(v)$

$M(u) \leftarrow M(u) + D(v)$

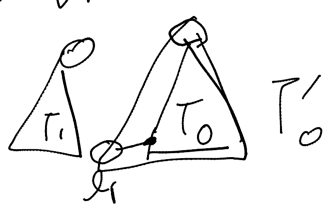
end for

$D(v) \leftarrow 0$

end for

Perform the split, updating the M values as fragments are merged.

3) union $l_1$ to $T_0$ to get $T_0'$




4) find the right most leaf $l_2$ in $T_0'$ containing the time $\leq t_2$

5) similar to 2) split $T_0'$ to $T_2, l_2, T_3$


6) union the $l_2$ and $T_2$ to get $T_2'$

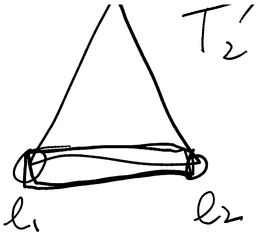
$D(r) = 0.1$

$M(r) = y$ $\quad 0.5$ $\quad (x, t_1, t_2)$

$T_2$ let $y = M$(root of $T_2'$)

if channel available.

$T_2'$ let $y = M$(root of $T_2'$)




if channel available.
$(x \leq 1 - y)$

then request can be processed

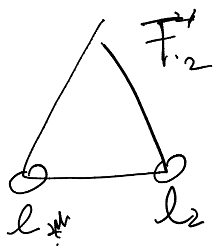
$(t_1, t_2]$
$t_1 \leq t < t_2$

else // cannot be proccessed.

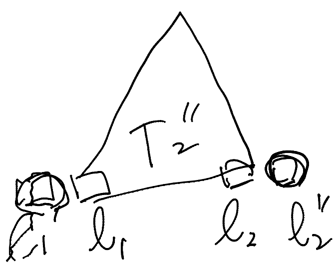
If request cannot be processed

merge $T_1, T_2', T_3$

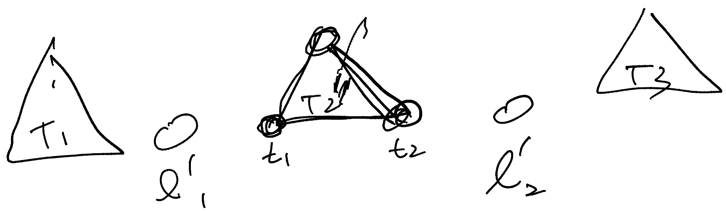
elseif if req. satisfied do




- from $l_1$ with interval $[t_1', t_1'']$ split off $l_1'$ with $(t_1', t_1]$, $[t_1, t'']$
- from $l_2$ ~~with interval~~ do likewise $[t_2', t_2''] \to (t_2', t_2], (t_2, t_2'']$




request, $(x, t_1, t_2)$

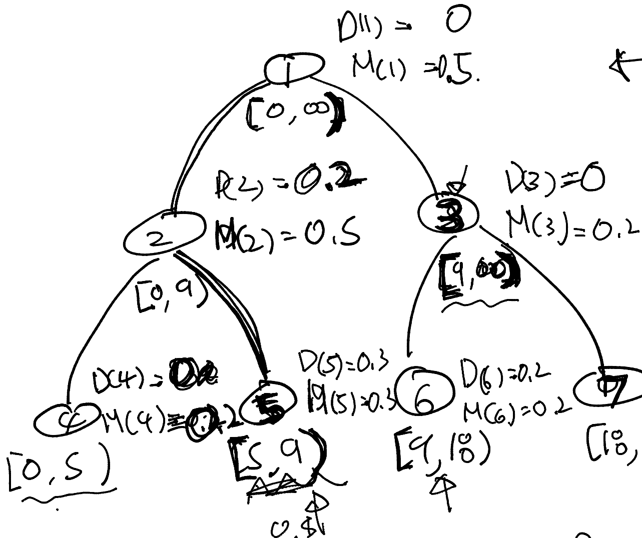



- add $x$ to $D$(root of $T_2''$)
- add $x$ to $M$(root of $T_2''$)
- merge $T_1, l_1', T_2'', l_2', T_3$
- Take care of the $M$ values.

end if

$D(1) = 0$

$\Gamma(0.2, 0, 18)$




D(7) = 0
M(7) = 0.

$\leftarrow \left[ \begin{array}{l} (0.2,\ 0,\ 18) \\ (0.3,\ \boxed{[5,9)}) \end{array} \right.$ $\qquad [t_1, t_2)$

req : (0.1, 7, 15)

$D(1) + D(2) + D(4) = 0.2$ ← path

$D(1) + D(2) + D(5) = 0.5$

def D $\quad D(1) + D(3) + D(6) = 0.2$

$D(4) + D(3) + D(7) = 0$

def M. $\quad M(v) = D(v) + \mathrm{Max}\{M(u)\}$

u child of v