Zaprojektuj strukturę danych dla dynamicznego zbioru liczb całkowitych S umożliwiającą wydajne wykonywanie następujących operacji:

Ini(S) :: $S := \emptyset$ // wykonywana raz na początku // obliczeń

Add(S, [a,b]) :: $S := S \cup \{a, a+1, \ldots, b\}$

Remove(S, [a,b]) :: $S := S \setminus \{a, a+1, \ldots, b\}$

Size(S) :: return $|S|$

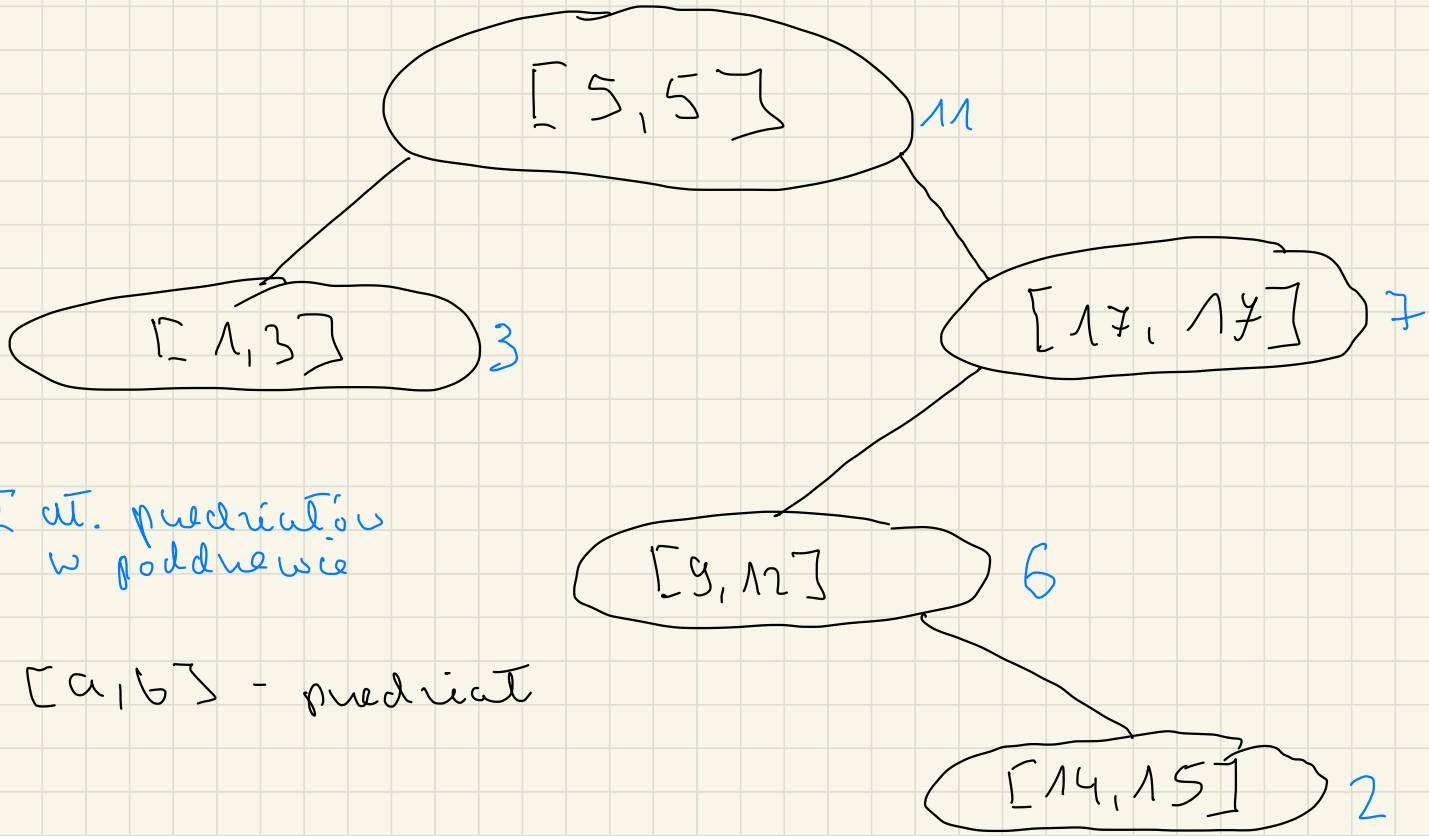
Rozwiązanie

Zbiór S będziemy reprezentować jako zbiór maksymalnych, domkniętych przedziałów liczb całkowitych, które sumują się (teoriomnogościowo) do S.

Przykład

$$S = \{1, 2, 3, 5, 9, 10, 11, 12, 14, 15, 17\}$$
$$= [1,3] \cup [5,5] \cup [9,12] \cup [14,15] \cup [17,17]$$

Zbiór S implementujemy jak drzewo wyszukiwań typu Splay. Przedziały są uporządkowane względem lewych końców. Dodatkowo w każdym węźle pamiętamy sumę długości przedziałów w poddrzewie.



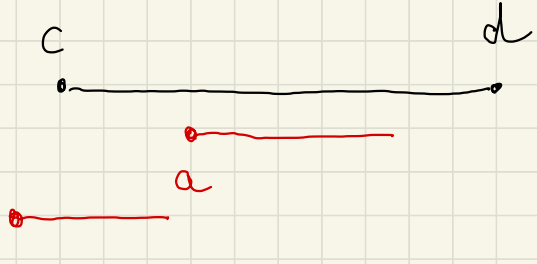
Σ wt. predykatów w poddrzewie

[a,b] - predykat

Proste ćwiczenie – pojedynczą rotację łatwo zaimplementować tak, żeby lokalnie w czasie stałym zachować sumy długości przedziałów w poddrzewie

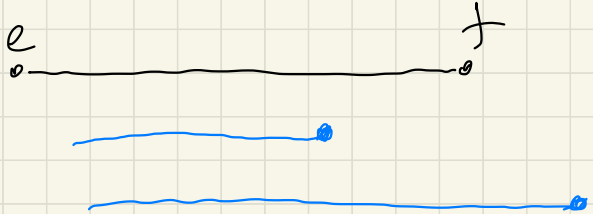
Pokażemy, w jaki sposób wykonać operację Add(S, [a, b]). Idea z dokładnością do operacji brzegowych:

- Szukamy w S przedziału $[c,d]$ z najmniejszym $d \geq a$. Mamy dwa przypadki:



1. $a \geq c$
2. $a < c$

- Szukamy w S przedziału $[e,f]$ z z największym $e \leq b$. Znowu mamy dwa przypadki:



1. $b \leq f$
2. $b > f$

Ze zbioru S usuwamy wszystkie przedziały od $[c,d]$ do $[e,f]$ włącznie i dodajemy nowy przedział $[x = \min(a,c), y = \max(b,f)]$.

Musimy jeszcze sprawdzić, czy prawy koniec przedziału $[x',y']$, bezpośrednio poprzedzającego $[x,y]$ w S, nie jest równy $x-1$. Jeśli tak usuwamy ten przedział z S, natomiast x zmieniamy na x'.
Podobnie robimy z prawym końcem.
Szukamy przedział $[x'',y'']$ bezpośrednio za $[x,y]$ w S. Jeśli $x'' = y+1$, to usuwamy z S $[x'',y'']$ i y zmieniamy na y''.

W jaki sposób usunąć szybko przedziały od $[c,d]$ do $[e,f]$. W tym celu wykorzystamy operacje Split i Join.

Na drzewie S (drzewo reprezentujące zbiór S) wykonujemy $Split(S,[c,d])$ i dostajemy drzewa S' z przedziałami $< [c,d]$ oraz drzewo S'' z przedziałami $\geq [c,d]$.

Niech $[e',f']$ będzie przedziałem bezpośrednio po $[e,f]$. Na S'' wykonujemy $Split(S'',[e',f'])$ i dostajemy drzewa $(S'')'$ i $|S''|''$. Na koniec robimy $Join(S', (S'')'')$.

Operacje Split, Join wykonujemy
w czasie zamortyzowanym $O(\log |S|)$
na drzewach typu Splay. Stosunkowo
łatwo je zaimplementować na 2-3-4-drzewach
(czerwono-czarnych) w pesymistycznym czasie
$O(\log |S|)$. Możliwa jest implementacja
w takim czasie na AVL-drzewach, ale
stosunkowo trudna.
Operację Remove wykonujemy podobnie
do Add - samodzielne ćwiczenie!