Idea:

- Reprezentovaná data/vstup/... rozdělit na bloky velikosti $\sqrt{n}$.
- Předpočítat pomocná data pro bloky.
- Intervalové dotazy/změny provádět na $O(\sqrt{n})$ celých blocích plus na $O(\sqrt{n})$ zbylých prvcích.

Příklad: Na posloupnosti $a_1, \ldots, a_n$ provádějte akce

- `add(f,t,d)`: K $a_f, a_{f+1}, \ldots, a_t$ přičti d
- `smaller(f,t,b)`: Kolik z čísel $a_f, a_{f+1}, \ldots, a_t$ je menších než b?

Nechť $s = \lceil\sqrt{n}\rceil$, B_i bude reprezentovat $a_{si+1}, \ldots, a_{si+s-1}$:

- Hodnota d_i.
- Setříděné pole s uloženými čísly a'_j tž. $a_j = a'_j + d_i$.
- Mapování $j \mapsto$ pozice a'_j.

`add(f,t,d)`:

- Pro každý blok B_i celý obsažený mezi f a t: $d_i += d$.
- Pro nanejvýš dva bloky částečně překrývající $[f, t]$, pro $f \le j \le t$: $a'_j += d$, setřídíme.

Časová složitost $O(n/s + s \log n) = O(\sqrt{n} \log n)$.

`smaller(f,t,b)`:

- Pro každý blok B_i celý obsažený mezi f a t: p $+=$ počet čísel menších než $b - d_i$ (půlení intervalů).
- Pro nanejvýš dva bloky částečně překrývající $[f, t]$: projdeme všechny prvky z $[f, t]$ v těchto blocích a počet menších než b přičteme k p.

Časová složitost $O(n/s \cdot \log n + s) = O(\sqrt{n} \log n)$.

Existuje i $O(\log^2 n)$ řešení (intervalový strom). Pro $n = 100000$:

- $\sqrt{n} \approx 320$
- $\log n \approx 16$
  - Složitější operace.
  - Více paměti s horším využitím cache.

Je těžké nastavit časové limity tak, aby tato řešení odlišila.

Příklad: Máme posloupnost $a_1, \ldots, a_n$, množiny indexů $S_1, \ldots, S_m \subseteq \{1, \ldots, n\}$ celkové velikosti t. Akce:

- `add(i,d)`: Ke všem a_j tž. $i \in S_j$ přičti d.
- `sum(i)`: Vrať $\sum_{j \in S_i} a_j$.

S_i je velká jestliže $|S_i| \geq s$. Pro každé i tž. S_i je velká:

- Předpočítáme $|S_i \cap S_j|$ pro každé j – $O((m+t) \cdot t/s)$.
- Udržujeme v $O(t/s)$ na každou operaci `add`:
  - $\Sigma_i = \sum_{j \in S_i} a_j$
  - d_i: součet hodnot přičtených k S_i.

Dále v $O(s)$ na operaci udržujeme

- a'_j: j-tý prvek s přičtenými změnami pro malé množiny.

`sum(i)` pro malou S_i je

$$\left(\sum_{j \in S_i} a'_j\right) + \sum_{S_k \text{ velká}} |S_k \cap S_i| d_k.$$

Časová složitost: $O(t/s + s) = O(\sqrt{t})$ na operaci pro $s = \lceil \sqrt{t} \rceil$.

Příklad: Je zadána posloupnost $a_1, \dots, a_n$ přirozených čísel. Odpovídejte na m dotazů `distinct(f,t)`: Počet navzájem různých čísel mezi $a_f, \dots, a_t$.

- Po pro aktuální interval $[f, t]$ si pamatujeme strom $a \mapsto$ počet výskytů a mezi $a_f, \ldots, a_t$ (pouze pro a tž. tento počet je nenulový).
- Odpověď: počet vrcholů stromu.
- Strom pro $[f, t+1]$, $[f, t-1]$, $[f+1, t]$, $[f-1, t]$ lze vytvořit v čase $O(\log n)$.

- Dotazy setřídíme dle $(\lfloor f/s \rfloor, t)$.
- Přechod mezi dotazy se stejným $\lfloor f/s \rfloor$:
  - $O(s \log n)$ levá hranice na každý přechod
  - $O(n \log n)$ pravá hranice celkově
- Přechod mezi dotazy s různým $\lfloor f/s \rfloor$: $O(n \log n)$

Časová složitost $O(ms \log n + n^2 \log n / s) = O(n\sqrt{m} \log n)$ pro $s = \lceil n/\sqrt{m} \rceil$.

Příklad: Máme graf G s n vrcholy (na začátku prázdný), posloupnost m operací `add_edge(u,v)`, `delete_edge(u,v)`, dotazů na počet komponent.

Operace rozdělíme do bloků po s. Pro každý blok:

- $G' = G -$ všechny `delete_edge(u,v)` z bloku.
- Inicializujeme DFU pro G'.
- Průběžně udržujeme G (ale ne DFU).
- Pro každý dotaz:
  - Do DFU přidej všechny hrany $E(G) \setminus E(G')$.
  - Odpověz dotaz.
  - Vrať DFU do původního stavu.

$O(m/s \cdot m \log n + ms \log n) = O(m^{3/2} \log n)$ pro $s = \lceil \sqrt{m} \rceil$.