

- Předpočet pro všechny podintervaly délky 2^i .
 - Čas/paměť $O(n \log n)$.
- Dotaz: interval délky ℓ pokryjeme dvěma int. délky $2^{\lfloor \log_2 \ell \rfloor}$.
 - Čas $O(1)$.

Vs. intervalový strom:

- Předpočet: podintervaly délky 2^i na pozicích dělitelných 2^i .
 - Čas/paměť $O(n)$.
 - Lze dynamicky upravovat v $O(\log n)$.
- Dotaz: interval pokryjeme $O(\log n)$ disjunktními.
 - Čas $O(\log n)$.
 - Žádný překryv.

Minimum z intervalu

Předpočet:

```
for (i = 0; i < n; i++) m[0][i] = vstup[i];
for (d = 1; (1 << d) <= n; d++)
    for (i = 0; i <= n - (1 << d); i++)
        m[d][i] = min (m[d-1][i],
                       m[d-1][i + (1 << (d-1))]);
```

Dotaz (minimum z intervalu $[od, \dots, od + len - 1]$):

```
int d = floor(log2(len));
return min (m[d][od], m[d][od + len - (1 << d)]);
```

Zadání

Pro zadaný řetězec odpovídejte na dotazy “Je podřetězec $r(p_1, \ell)$ délky ℓ od pozice p_1 lexikograficky menší než podřetězec $r(p_2, \ell)$?”

Předpočet: Pro $d \leq \log_2 n$ určíme $c_{i,d} \in \{0, \dots, n-1\}$ tž.

$$c_{i,d} < c_{j,d} \text{ právě když } r(i, 2^d) < r(j, 2^d).$$

- Setřídíme podřetězce $r(i, 2^d)$ a očíslovíme dle pořadí.
- $r(i, 2^d) < r(j, 2^d)$ právě když $(c_{i,d-1}, c_{i+2^{d-1},d-1}) < (c_{j,d-1}, c_{j+2^{d-1},d-1})$.
- Bucket sort.

Čas: $O(n \log n)$

Zadání

Pro zadaný řetězec odpovídejte na dotazy “Je podřetězec $r(p_1, \ell)$ délky ℓ od pozice p_1 lexikograficky menší než podřetězec $r(p_2, \ell)$?”

Dotaz:

- $d = \lfloor \log_2 \ell \rfloor$
- $r(i, \ell) < r(j, \ell)$ právě když $(c_{i,d}, c_{i+\ell-2^d,d}) < (c_{j,d}, c_{j+\ell-2^d,d})$

Čas: $O(1)$

Lexikografické setřídění všech suffixů:

- Za řetězec doplníme spec. znaky s nejmenší hodnotou.
- Číslování dle pořadí pro podřetězce délky $2^0, 2^1, \dots, 2^{\lceil \log_2 n \rceil}$.
- Pro délku $2^{\lceil \log_2 n \rceil}$: Pořadí suffixů.

b a n a n a \emptyset \emptyset \emptyset \emptyset ...

1: 2 1 3 1 3 1 0
b a n a n a \emptyset ...

2: 3 2 4 2 4 1 0
2 b 1 a 3 n 1 a 3 n 1 a ~~6~~
1 2 3 n 1 2 3 n 1 2 ~~6~~ ~~0~~

4: 4 3 6 2 5 1 0
3 b 2 a 4 n 2 a 4 n 1 a \emptyset \emptyset
4 n 2 n 4 n 1 \emptyset 0 \emptyset 0 \emptyset

8: 4 3 C 2 5 1
b a n a n a
a n n n
n n n n
n n n n

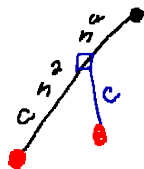
| | | | | | |
|---|-----|---------|-----------|-----|-------|
| 5 | 3 | 1 | 0 | 4 | 2 |
| a | n n | b n n n | b n n n n | n n | n n n |

Nejdelší společný prefix od pozic p_1 a p_2 :

```
d = ceil(log2(n));
spolecny = 0;
while (d >= 0):
    if c[p1][d] == c[p2][d]:
        spolecny += 1 << d;
        p1 += 1 << d;
        p2 += 1 << d;
    d--
```

Konstrukce suffixového stromu

- Vytvořit suffixové pole.
- Postupně přidáváme suffixy do stromu:
 - Sestup na výšku společného prefixu s předchozím suffixem.
 - Je-li to potřeba, vytvoř nový vrchol podrozdělující hranu.
 - Přidej list pro nový suffix.



b a h a h c;

| | | | | | |
|---|---|---|---|---|---|
| 1 | 3 | 0 | 5 | 2 | 4 |
| a | a | b | c | h | h |
| h | h | a | | a | c |
| a | c | h | | h | |
| h | | a | | c | |
| c | | h | | | |
| | | c | | | |