Tópicos Especiais em Algoritmos
Árvores de Segmentos e Árvores de Fenwick
Editorial

Daniel Saad Nogueira Nunes

Codeforces 313B: Ilya and Queries

Este problema pode ser resolvido através de uma técnica de soma de prefixos. Dada a string de entrada $S[0, n-1]$, a ideia é construir um vetor $V[0, n-1]$ da seguinte forma:

$$V[i] = \begin{cases} 0, & i = 0 \\ V[i] + 1, & i > 0 \land S[i] = S[i-1] \\ V[i], & i > 0 \land S[i] \neq S[i-1] \end{cases}$$

Para responder uma consulta sobre um intervalo $[i, j]$, basta tomar basta tomar $V[j] - V[i]$ no vetor de soma de prefixos.

Complexidade

$\langle O(n), O(1) \rangle$: $O(n)$ para processar V e $O(1)$ por consulta.

Resolução com Fenwick Trees

É possível resolver o problema de maneira similar através de Fenwick Trees. Para isto, a definição do vetor é um pouco diferente:

$$V[i] = \begin{cases} 0, & i = 0 \\ 1, & i > 0 \land S[i] = S[i-1] \\ 0, & i > 0 \land S[i] \neq S[i-1] \end{cases}$$

Para responder uma consulta sobre um intervalo $[i, j]$, basta realizar a operação $sum(i, j) = sum(j) - sum(i-1)$ através de uma Fenwick Tree construída sobre V. Observe que se $i = 0$, o resultado é simplesmente $sum(j)$.

Complexidade

$\langle O(n), O(\lg n)\rangle$: $O(n)$ para processar V e $O(\lg n)$ por consulta.

AtCoder ABC223F: Parenthesis Checking

Seja $S[0, n-1]$ a *string* de parênteses de entrada. Podemos definir de maneira similar a S um vetor V de soma de prefixos que soma o valor anterior com 1, se $S[i] =)$ e -1 caso contrário. Para ser mais preciso:

$$V[i] = \begin{cases} 1, & S[i] = (\text{ e } i = 0 \\ -1, & S[i] =) \text{ e } i = 0 \\ V[i-1]+1, & S[i] = (\text{ e } i > 0 \\ V[i-1]-1, & S[i] =) \text{ e } i > 0 \end{cases}$$

Seja $e(i, j)$ o excedente de parênteses abertos em relação aos fechados de acordo com $S[i, j]$. Isto é, $e(i, j) = V[j] - V[i-1]$. Caso $i = 0$, temos que $e(i, j) = V[j]$ apenas. Também tome $rmq(i, j)$ o valor mínimo em $V[i, j]$. Uma observação crucial para resolver o problema é perceber que, uma sequência $S[i, j]$ é dita balanceada se e somente se:

- $e(i, j) = 0$
- $rmq(i, j) = 0$.

Podemos construir uma árvore de segmentos que, para cada intervalo $[l, r]$ representado na árvore, armazena o par $(e(l, r), rmq(l, r))$. Um nó v desta árvore, cujos nós filhos são v_l e v_r pode ser atualizado/inicializado tomando:

$$\begin{aligned} &v.e = v_l.e + v_r.e \\ &v.rmq = \min(v_l.rmq, v_l.e + v_r.rmq) \end{aligned}$$

Já uma folha v sobre o intervalo $[l, l]$ recebe:

$$\begin{aligned} &v.e = 1, \text{ se } S[l] =) \\ &v.e = -1, \text{ se } S[l] = (\\ &v.rmq = 0 \end{aligned}$$

Um aspecto importante desta questão é que é possível atualizar a entrada na forma de troca de conteúdos de diferentes posições (swap). Mas a definição dos nós internos e folha não muda, bastando implementar a atualização sobre as árvores de segmentos.

Complexidade

$\langle O(n), O(\lg n), O(\lg n)\rangle$: $O(n)$ para construção da árvore, $O(\lg n)$ por consulta e $O(\lg n)$ por atualização.