2025/10/17 03:14 1/2 个人总结

个人总结

陈铭煊 Max.D.

本周是比较摸的一周,主要学习了一下广义后缀自动机(不敢说自己已经会了),稍微写了下FWT的知识库。

龙鹏宇 Hardict

肖思炀 MountVoom

这周是特别摸的一周,补了补题,没学啥新东西。

摸了一套div.4和一套div.3[doge]□等明天再摸一套div.2□

摸了.jpg[]

本周推荐

陈铭煊 Max.D.

主要介绍一下rope这个容器[]rope并不在C++标准库中,而是在GNU g++的ext扩展库系列中,和pb_ds是属于平级关系[]C++98环境也一样可以使用(注意:如果不是g++编译器自然无法使用)。

rope的主要目标,是对一个经过多次拼切、修改、替换而来的超长字符串进行维护。当然也可以用于普通的序列维护(为了体现字符串的地位,rope<char>被特化成了crope类型)。

rope对于序列的插入、删除、替换、拼接、切片等操作快速到了\$\log n\$级别。这是vector或者string所不具备的,因为后者的这些操作,基本上都要对元素进行整体移位。可想而知,其内部是使用了某种BST的结构[]BST的每一个结点,代表了一个序列区间,而子节点则进一步划分成了两个子区间——我们自己手写像Splay[]Treap这样的BST[]也能比较轻松地达到rope的要求。

rope的另外一大能力,是进行**可持久化**,或者说"写时复制"——在rope进行复制或者部分复制时□BST会尽量利用原序列已有的部分,在对新串进行修改时,才新建部分节点。这种可持久化,能够帮助我们实现一些手写比较困难的数据结构,例如洛谷3369 **可持久化平衡树**,代码详见个人主页。

但rope的缺点也是显而易见的——没有办法对区间添加额外的域进行维护(比如没有办法快速求区间和,没有办法进行快速区间翻转,这里快速是指的\$\log n\$级别),另外常数也是比手写的平衡树大了不少。 这些场合还是建议手写(当然□pb_ds中的BST可以弥补这一点,但是额外的代码不必手写简单多少)。

一些常见的操作如下:

R.push back(a) //往后插入 , 也可以用+=

update: 2020-2021:teams:alchemist:weekly_digest_2 https://wiki.cvbbacm.com/doku.php?id=2020-2021:teams:alchemist:weekly_digest_2&rev=1589528751

- R.insert(pos,a)//在pos位置插入a[pos是一个迭代器。
- R.erase(pos,n)//在pos位置删除n个元素。
- R.replace(pos,x)//*从pos开始替换成x*
- R.substr(pos,x)//*从pos开始提取x个。*

其他还有相当多操作不便于列举。大家可以参看网址SGI Rope (没错□rope是来源于SGI STL的, g++基本上是将其拷贝过来了)。其中实现的原理,参考维基百科

龙鹏宇 Hardict

肖思炀 MountVoom

不知道推荐什么好,就推荐一道数据结构题吧。

题解

From:

https://wiki.cvbbacm.com/ - CVBB ACM Team

Permanent link:

 $https://wiki.cvbbacm.com/doku.php?id=2020-2021:teams:alchemist:weekly_digest_2\&rev=158952875126.$

Last update: 2020/05/15 15:45



https://wiki.cvbbacm.com/ Printed on 2025/10/17 03:14