

谨慎使用STL



lxfeng (/u/62c6f75ec9d5) [+ 关注](#)
2016.08.15 21:56* 字数 1117 阅读 831 评论 0 喜欢 1 阅读 831 评论 0 喜欢 1 (/u/62c6f75ec9d5)

谨慎使用STL

最近解决了一个因为大量使用STL造成的严重内存泄漏问题，再次记录下。

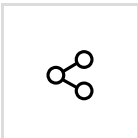
上周上线了基于用户tag的推荐，没天服务器会自动生成新的tag数据，然后scp到指定目录下，推荐服务中，对文件做了监控，如果改变就会重新加载解析。

函数代码如下：

```
void ReposManager::ReLoadUidTagData(const std::string& file_name) {
    std::string file_content;
    bool ret = file::ReadFileToString(file_name, &file_content);
    std::unordered_map<int, std::unordered_map<std::string, double> > uid_tags_map;
    if (ret == false) return;
    std::vector<std::string> content;
    SplitString(file_content, '\n', &content);
    ifstream fin(file_name);
    for (auto line: content)
    {
        std::vector<std::string> str;
        std::vector<std::string> vstr;
        SplitString(line, '=', &str);
        if (str.size() < 2) continue;
        double weight = StringToDouble(str[1]);
        SplitString(str[0], ':', &vstr);
        if (vstr.size() < 2) continue;
        int uid = StringToInt(vstr[0]);
        std::string tag = vstr[1];
        std::unordered_map<std::string, double>& tags_map = uid_tags_map[uid];
        tags_map[tag] = weight;
    }
    for (auto it = uid_tags_map.begin(); it != uid_tags_map.end(); ++it) {
        int uid = it->first;
        std::unordered_map<std::string, double>& uid_tags = uid_tags_map[uid];
        std::vector<std::string> tags;
        for (auto it = uid_tags.begin(); it != uid_tags.end(); ++it) {
            tags.push_back(it->first);
        }
        auto cmp = [&](const std::string& a, const std::string& b) {
            return uid_tags[a] > uid_tags[b];
        };
        std::sort(tags.begin(), tags.end(), cmp);
        uid_tags_[uid] = tags;
    }
}
```

文件存储格式为uid:tag=weight 有很多行，共220M左右，函数的功能是对每行进行分割，然后存到成员变量uid_tags_中。函数初看之下没什么问题，但上线后第二天内存涨到3.6G，线上服务器内存很大也不能这么浪费，我记得刚启动加载完毕才2.0G，怎么涨了这么多，考虑原因，估计是凌晨的文件变更再次解析导致的，重启改动文件测试，果然是这个。

文件220M，全部加载进内存，解析过程中算上额外的数据结构的确会引起内存增长，但这些内存消耗只是暂时的，局部变量函数结束后，内存就释放了，显然这里的一些局部变量没有释放掉，这个情况还的确没遇到过。



google了解到STL的内存分配器，分多级，对于申请交大的内存，他会在堆上去申请。我们理解的局部变量出了作用域就会释放掉，这是因为大部分的局部变量都在函数栈里，函数执行完了，整个栈都会释放掉。周末在家试了下，尝试手动通过vector的swap以及map的clear，手动释放，在本机有一定的效果，内存没有继续增长，以为解决了。

周一到公司，放到测试服务器上，发现还是没有解决了，内存还是会出现较大增长，同样的二进制文件，为何会有差异，考虑再三，估计是虚拟机和测试服的配置不同，虚拟机内存不大，所以内存释放得会快点，测试服内存十几个G，内存不是很紧张，所以长时间得不到释放。(看来保持测试环境和线上环境的统一，这个还是很有必要的，保证了运行环境的一致，便于排查问题)。

在知乎看到有人回答malloc_trim（0） ，google了到使用malloc_trim(0),后来发现无效，然后看C语言的malloc只是提供了接口，具体的实现在各个平台不一样，linux下是glibc的里的malloc，了解到可以用google的tcmalloc替换掉linux默认的malloc，替换后内存从原来高达3.6G到2.6G，看来效果还是杠杆的! 回头自己对代码做了下检查，觉得存储的文件可以再修改下，

```
uid:tag1=weight
uid:tag2=weight
...
```

改为：

```
uid:tag1=weight,tag2=weight,tag3=weight....
```

将每个uid的所属tag整合到一起，原来220M的文件变味了150M左右，然后修改文件解析函数：

```

void ReposManager::ReLoadUidTagData(const std::string& file_name) {

    std::string file_content;
    bool ret = file::ReadFileToString(file_name, &file_content);
    if (ret == false) return;
    std::vector<std::pair<std::string, double>> tag_list;
    using namespace std;
    auto cmp = [](const pair<string, double>& a, const pair<string, double>& b) {
        return a.second > b.second;
    };
    std::vector<std::string> content;
    SplitString(file_content, '\n', &content);

    for (auto line: content) {
        if (line.length() == 0) continue;
        std::vector<std::string> str;
        SplitString(line, ':', &str);
        if (str.size() < 2) continue;
        int uid = StringToInt(str[0]);
        std::vector<std::string> tags;
        SplitString(str[1], ',', &tags);
        tag_list.clear();
        for (int i = 0; i < tags.size(); i++) {
            std::vector<std::string> tmp;
            SplitString(tags[i], '=', &tmp);
            if (tmp.size() < 2) continue;
            std::string tag = tmp[0];
            double weight = StringToDouble(tmp[1]);
            tag_list.push_back(std::pair<string, double>(tag, weight));
        }
        std::sort(tag_list.begin(), tag_list.end(), cmp);
        uid_tags_[uid].clear();
        for (int i = 0; i < tag_list.size(); ++i) {
            uid_tags_[uid].push_back(tag_list[i].first);
        }
    }
}

```

相比于修改前的，减少了一个 STL局部变量。

std::unordered_map<int, std::unordered_map<std::string, double> > uid_tags_map;

修改完毕，放到测试服务器测试，内存占用稳定到1.5G左右，多次加载不会出现增长，问题搞定，上线部署。

代码都会写，如何写出高性能，高质量的代码，这就是个技术活。

总结：


1. malloc()/free()作为C标准，ANSI C并没有指定它们具体应该如何实现。各个平台上（windows, mac, linux等等），调用这两个函数时，实现不一样。
2. 在linux下，malloc()/free()的实现是由**glibc库**负责的。STL的内存释放，有时候并没有直接返还给os，只是返还给了分配器。
3. 针对大量数据，谨慎大量使用STL局部变量，虽然栈上分配的，但它维护的队列是分配在heap上的，它操作的内存不一定能够即使释放，可能产生碎片。

stackoverflow 问题 ([https://link.jianshu.com?](https://link.jianshu.com?t=http://stackoverflow.com/questions/10943907/linux-allocator-does-not-release-small-chunks-of-memory)

t=<http://stackoverflow.com/questions/10943907/linux-allocator-does-not-release-small-chunks-of-memory>)

ptmalloc理解 ([https://link.jianshu.com?](https://link.jianshu.com?t=http://blog.csdn.net/phenics/article/details/777053)

t=<http://blog.csdn.net/phenics/article/details/777053>)



lxfeng (/u/62c6f75ec9d5)

写了 24512 字，被 14 人关注，获得了 23 个喜欢
(/u/62c6f75ec9d5)写了 24512 字，被 14 人关注，获得了 23 个喜欢

+ 关注

C/C++, 后端, Java 高并发,分布式系统

喜欢 | 1







更多分享

(http://cwb.assets.jianshu.io/notes/images/5288852/w



下载简书 App ▶

随时随地发现和创作内容



(/apps/redirect?utm_source=note-bottom-click)



(/sign_in?utm_source=desktop&utm_medium=not-signed-in-comment-form)

评论

智慧如你，不想发表一点想法 (/sign_in?utm_source=desktop&utm_medium=not-signed-in-nocomments-text)
咩~

推荐阅读

[更多精彩内容](#) > (/)

Chapter 49.你快躲到被子里去 (/p/7838473e988e?utm_campaign=mal...

坐在自家客厅里的萨拉警长莫名觉得阴风阵阵，疑心是自己坐太久的关系决定去趟洗手间活动活动，出来的时候刚好看到卡嘉莉挽着母亲下来。随后他就发现，卡嘉莉看向自己的眼神明显变了。“怎么了？”他挑

醋溜土豆丝不加辣 (/u/8ff67168e4f1?
utm_campaign=maleskine&utm_content=user&utm_medium=pc_all_hots&utm_source=recommendation)

佛说今生 (/p/5bb8983d4cf5?utm_campaign=male... (/p/5bb8983d4cf5?

我倚着海市蜃楼，吐呐着千年的风。我看见，在心的横切面上，长睡着一朵莲花。佛说：你要静静等待，等待花开的时刻，等待风拂过花萼，以唤醒前世种
utm_campaign=maleskine&utm_content=note&utm_m

割钦妈妈 (/u/2bf1d6f722a1?

utm_campaign=maleskine&utm_content=user&utm_medium=pc_all_hots&utm_source=recommendation)

被关注是一种幸福 (/p/a37ef414e10f?utm_campaign=maleskine&utm_content=note&utm_medium=seo_notes&utm_source=recommendation)

作者:任争气 我们都希望被关注 那是一种心理的自我满足 是幸福感的源泉 我们也害怕被关注 害怕暴露于光天化日之下 所以 总有两个我在头脑里掐架 我们

醉美长安 (/u/7886cd7da46e?

utm_campaign=maleskine&utm_content=user&utm_medium=pc_all_hots&utm_source=recommendation)

大腹便便 (/p/9801ce363938?utm_campaign=maleskine&utm_content=note&utm_medium=seo_notes&utm_source=recommendation)

水中游沙里爬寄在石头的篱下万般具样样应夏 白灼盐焗炆炖炒不在话丁 补充微量元素钙磷铁钾海草滚汤补碘佳脚勤手快口不停不长肥膘才假罢了又罢

蒋光头jL94430 (/u/3fa0ac4f3559?

utm_campaign=maleskine&utm_content=user&utm_medium=pc_all_hots&utm_source=recommendation)

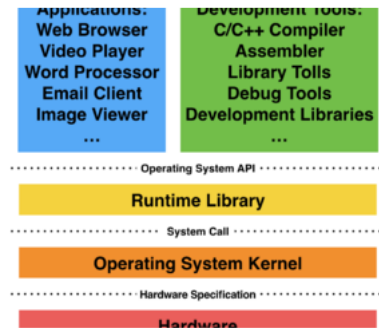
人在旅途 (/p/c12eb4e3b2c4?utm_campaign=maleskine&utm_content=note&utm_medium=seo_notes&utm_source=recommendation)

2018年7月3日 星期二 雨 我们的人生总是太匆匆，如果老是在一个地方混着日子的话，那么我们的人生没有波澜，像一潭死水，一眼可以望到头。历史上许

精进的医生 (/u/61d710a4a0ce?

utm_campaign=maleskine&utm_content=user&utm_medium=pc_all_hots&utm_source=recommendation)


(/p/3795dc97dc61?



utm_campaign=maleskine&utm_content=note&utm_medium=seo_notes&utm_source=recommendation)

iOS 程序员的自我修养 — 读《程序员的自我修养-链接、装载与库》 (/p/3795dc97dc61?utm_campaign=maleskine&utm_content=note&utm_medium=seo_notes&utm_source=recommendation)

2016年国庆假期终于把此书过完，整理笔记和体会于此。 关于书名 书名源于俄罗斯的演员斯坦尼斯拉夫斯基创作的《演员的自我修养》， 作者为了写这本书前前后后修改了三十年之久，临终前才同意不在修

 李剑飞的简书 (/u/0428bf888c54?

utm_campaign=maleskine&utm_content=user&utm_medium=seo_notes&utm_source=recommendation)

iOS面试题300+ (/p/0d2713164646?utm_campaign=maleskine&utm_content=note&utm_medium=seo_notes&utm_source=recommendation)


*面试心声:其实这些题本人都没怎么背,但是在上海 两周半 面了大约10家 收到差不多3个offer,总结起来就是把基础的东西弄好,复杂的东西了解就ok了! *此题库是2015－2106年北上广深杭各大小公司面试题。 *

 Dove_iOS (/u/feadf9b4f0b1?

utm_campaign=maleskine&utm_content=user&utm_medium=seo_notes&utm_source=recommendation)

C总结 (/p/8f276e206752?utm_campaign=maleskine&utm_content=note&utm_medium=seo_notes&utm_source=recommendation)

C语言笔记 一终端命令 ls -l显示当前工作路径下的所有的文件及文件信息 d开头文件夹 -开头文件 r读 w写 x执行 - -当前用户的权限 - -其他用户的权限 > pwd查看终端程序的工作路径 >cd切换工作路径 >clear清理

 eric007shine (/u/6aea5963decd?

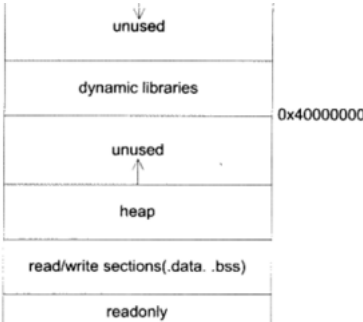
utm_campaign=maleskine&utm_content=user&utm_medium=seo_notes&utm_source=recommendation)

读书笔记 - 《程序员的自我修养》 (/p/78a1e8d85e5f?utm_campaign=maleskine&utm_content=note&utm_medium=seo_notes&utm_source=recommendation)

一、温故而知新 1. 内存不够怎么办 内存简单分配策略的问题地址空间不隔离内存使用效率低程序运行的地址不确定 关于隔离： 分为 虚拟地址空间 和 物理地址空间 分段： 把一段程序所需要的内存空间大小映射

 SeanCST (/u/7283ea6ff5b6?utm_campaign=maleskine&utm_content=user&utm_medium=seo_notes&utm_source=recommendation)

(/p/cc1bdada166f?



utm_campaign=maleskine&utm_content=note&utm_medium=seo_notes&utm_source=recommendation)
C++ 面试知识点总结 (/p/cc1bdada166f?utm_campaign=maleskine&ut...

1. C++基础知识点 1.1 有符号类型和无符号类型 当我们赋给无符号类型一个超出它表示范围的值时，结果是初始值对无符号类型表示数值总数取模之后的余数。当我们赋给带符号类型一个超出它表示范围的值

 Mr希灵 (/u/ccb6e3e26ec3?utm_campaign=maleskine&utm_content=user&utm_medium=seo_notes&utm_source=recommendation)

第二十七章 (/p/fd3b3dbb05ea?utm_campaign=maleskine&utm_conte...

看来不用等到明天了，半夜子默在离他们休息的地方的不远处修炼，忽然听到一个脚步声远远的过来，其实只是脚步一点一点的御风术，只是在黑夜里所以才这般的小心翼翼，只是太过悉心才被子默发现。于是

 闹别扭的鱼 (/u/8cb6d3919071?utm_campaign=maleskine&utm_content=user&utm_medium=seo_notes&utm_source=recommendation)

11月5日 (/p/0b17b43f78f8?utm_campaign=maleskine&utm_content=...

感恩今天好天气，一早带妈妈和芮芮去植物园，感恩哥哥不请自来，在我们出发前来看妈妈，也一起陪妈妈玩，还带了两瓶泡椒开。感恩植物园对老年人的优惠政策，一看妈妈就连证件都不查就给免门票了。植

 顺路99 (/u/924308454e7b?utm_campaign=maleskine&utm_content=user&utm_medium=seo_notes&utm_source=recommendation)

iOS ARC 和 MRC 混合开发(注意事项) (/p/1e4cefb71723?utm_campaign...

//联系人:石虎QQ: 1224614774昵称:嗡嘛呢叭咪哄 ARC & MRC混合开发 在项目开发中，遇到使用MRC开发的第三方库怎么办？例如：ASI1>尝试使用Xcode的转换工具（失败率比较高）2>在编译选项中，为

 石虎132 (/u/ff32105744ef?utm_campaign=maleskine&utm_content=user&utm_medium=seo_notes&utm_source=recommendation)

分手故事其十（男版）#敬往事一杯酒，再爱也不回头# (/p/cfdf0d3ba5e...

联系上南瓜的时候，他告诉我他打算离开深圳回老家了。是啊，留在这个城市的目的，是为了和你好好在一起，多年的风雨都经历了。最终却可笑的败给了平淡生活的琐碎。 不知道为何，听到这个消息我突然觉

 奋斗中的小Li子 (/u/842f159b212f?utm_campaign=maleskine&utm_content=user&utm_medium=seo_notes&utm_source=recommendation)

幽丝 (/p/128710169b4b?utm_campaign=maleskine&utm_content=not...

幽丝幽丝歌如丝 静无语 淑宁更害羞 年年朝朝长挽头 飘逸在心中

 王尚千岁 (/u/fbf3e3cefcf7?utm_campaign=maleskine&utm_content=user&utm_medium=seo_notes&utm_source=recommendation)