云风的 BLOG

思绪来得快去得也快，偶尔会在这里停留

« Unity3D asset bundle 格式简析 | 返回首页 | 近日工作记录 »

STM 的简单实现

STM 全称 Software transactional memory 。

在前年的项目里，我制作了一个类似的东西。随着 skynet 的日趋完善，我希望找到一个更为简单易用的方法实现类似的需求。

对于不常更新的数据，我在 skynet 里增加了 sharedata 模块，用于配置数据的共享。每次更新数据，就将全部数据打包成一个只读的树结构，允许多个 lua vm 共享读。改写的时候，重新生成一份，并将老数据设置脏标记，指示读取者去获取新版本。

这个方案有两个缺点，不适合实时的数据更新。其一，更新成本过大；其二，新版本的通告有较长时间的延迟。

我希望再设计一套方案解决这个实时性问题，可以用于频繁的数据交换。（注：在 mmorpg 中，很可能被用于同一地图上的多个对象间的数据交换）

一开始的想法是做一个支持事务的树结构。对于写方，每次对树的修改都同时修改本地的 lua table 以及被修改 patch 累计到一个尽量紧凑的序列化串中。一个事务结束时，调用 commit 将快速 merge patch 。并将整个序列化串共享出去。相当于快速做一个快照。

读取者，每次读取时则对最新版的快照增加一次引用，并要需反序列化它的一部分，变成本地的 lua table 。

我花了一整天实现这个想法，在写了几百行代码后，意识到设计过于复杂了。因为，对于最终在 lua 中操作的数据，实现一个复杂的数据结构，并提供复杂的 C 接口去操作它性能上不会太划算。更好的方法是把数据分成小片断（树的一个分支），按需通过序列化和反序列化做数据交换。

既然序列化过程是必须的，我们就不需要关注数据结构的问题。STM 需要管理的只是序列化后的消息的版本而已。这一部分（尤其是每个版本的生命期管理）虽然也不太容易做对，但结构简单的多。

我在 skynet 的 dev 分支上提交了叫做 stm 的 lua C 模块。

obj = stm.new(msg,sz) 可以用来生成一个用于数据交换的消息对象。通常，可以使用 skynet.pack(...) 来得到这个 msg 指针，和 sz 长度。

如果想把这条消息传递给别的服务，可以先用 copy = stm.copy(obj) 获得一个 copy 。这个 copy 是一个lightuserdata ，把它发送出去即可。

获得这个 copy 的一方，使用 reader = stm.newcopy(copy) 就能拿到这个对象了。

使用 reader(function(msg, sz) ... end) 可以把内含的 msg/sz 解出来。通常用 reader(skynet.unpack) 即可。

reader 返回的第一个参数为 true 时，成功获取了数据。之后是解码函数的返回值。

若第一个参数为 false 。stm 对象中可能没有数据，也可能版本没有更新。

这里提到的版本指：生成数据的一方，可以用 obj(msg,sz) 更新对象里面的数据。而读取方能正确感知数据的更新。

---

test/teststm.lua 是一个简单的范例。

云风 提交于 August 12, 2014 02:38 PM | 固定链接

COMMENTS

云风, 好好管管你们的策划和测试员工阿, 陌陌争霸要被他们整黄了.

Posted by: 五名 | (4) August 23, 2014 01:40 PM

这个目前没用到。

Posted by: 麦子 | (3) August 21, 2014 11:29 AM

不是很明白 这是门心语言吗

Posted by: 钢管调直机 | (2) August 14, 2014 11:22 PM

或许可以参考 Linux 源代码中的 RCU。

Posted by: Grissiom | (1) August 13, 2014 07:09 PM

POST A COMMENT

非这个主题相关的留言请到：留言本

名字：

Email 地址：

为了验证您是人类，请将六加一的结果（阿拉伯数字七）填写在下面：

URL:

记住我的信息？

留言：
（不欢迎在留言中粘贴程序代码）

提交