

golang hashmap的使用及实现



icexin (/u/737a4b893f51) [+ 关注](#)

2016.05.12 21:38* 字数 1254 阅读 1958 评论 0 喜欢 5 阅读 1958 评论 0 喜欢 5

(/u/737a4b893f51)

基本语法

定义hashmap变量

由于go语言是一个强类型的语言，因此hashmap也是有类型的，具体体现在key和value都必须指定类型，比如声明一个key为string，value也是string的map，需要这样做

```
var m map[string]string // 声明一个hashmap，还不能直接使用，必须使用make来初始化
m = make(map[string]string) // 初始化一个map
m = make(map[string]string, 3) // 初始化一个map并附带一个可选的初始bucket（非准确值，只是有提示意义）

m := map[string]string{} // 声明并初始化

m := make(map[string]string) // 使用make来初始化
```

大部分类型都能做key，某些类型是不能的，共同的特点是：**不能使用==**来比较，包括:slice, map, function

get,set,delete

```
m := map[string]int
m["a"] = 1

fmt.Println(m["a"]) // 输出 1

// 如果访问一个不存在的key，返回类型默认值
fmt.Println(m["b"]) // 输出0

// 测试key是否存在
v, ok := m["b"]
if ok {
    ...
}

// 删除一个key
delete(m, "a")
```

迭代器

```
// 只迭代key
for k := range m {
    ...
}

// 同时迭代key-value
for k, v := range m {
    ...
}
```

在迭代的过程中是可以对map进行删除和更新操作的，规则如下：

- 迭代是无序的，跟插入是的顺序无关
- 迭代的过程中删除一个key，无论遍历还是没有遍历过都不会再遍历到

- 迭代的过程中添加一个key，不确定是否能遍历到
- 未初始化的map也可以迭代

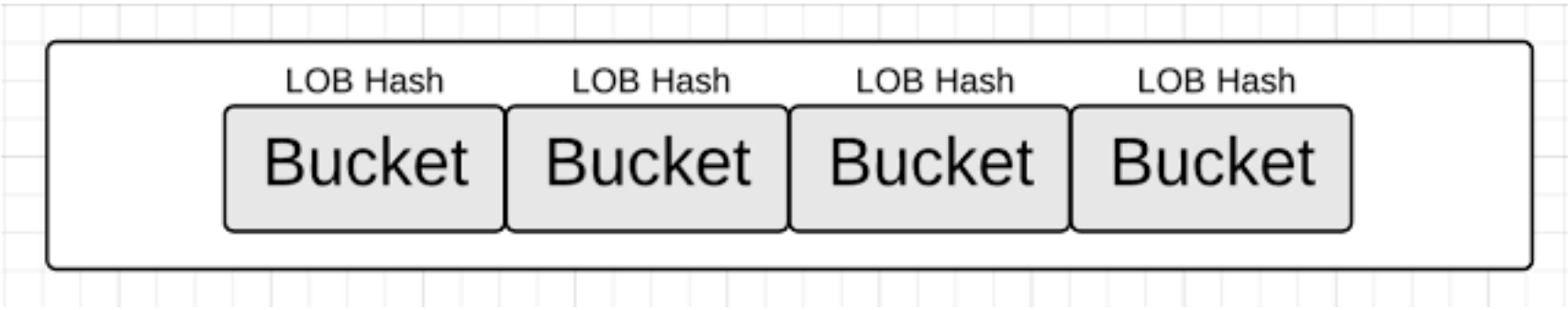
其他

- map的value是不可取地址的，意味着 &m["a"]这样的语法是非法的
- len和cap分别可以获取当前map的kv个数和总容量

内部结构

hashmap结构

golang的map是hash结构的，意味着平均访问时间是O(1)的。同传统的hashmap一样，由一个个bucket组成:



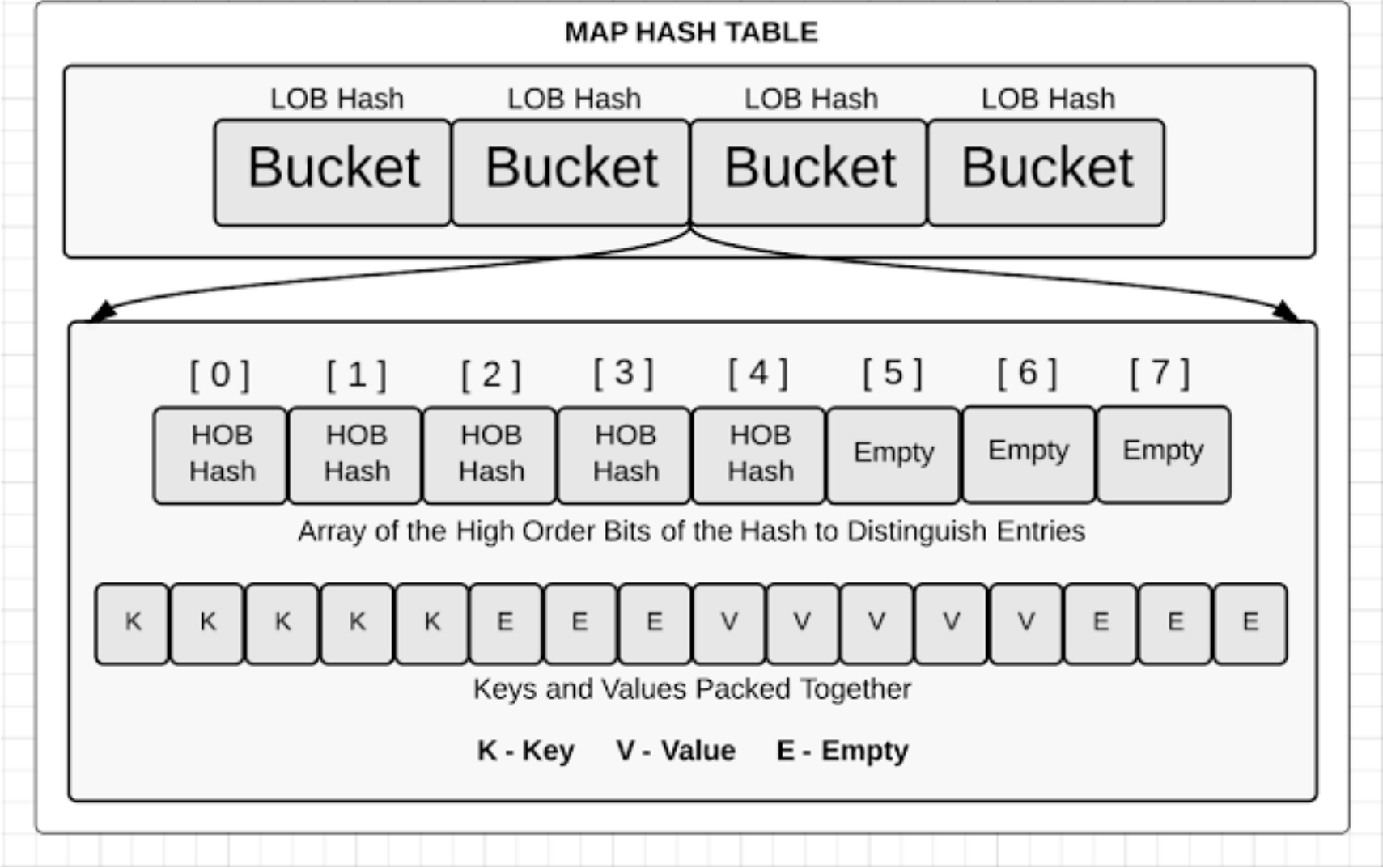
hashmap内部结构

```
// A header for a Go map.
type hmap struct {
    // Note: the format of the Hmap is encoded in ../../cmd/internal/gc/reflect.go a
    nd
    // ../../reflect/type.go. Don't change this structure without also changing that c
    ode!
    count int // # live cells == size of map. Must be first (used by len() builtin)
    flags uint8
    B      uint8 // log_2 of # of buckets (can hold up to loadFactor * 2^B items)
    hash0  uint32 // hash seed

    buckets    unsafe.Pointer // array of 2^B Buckets. may be nil if count==0.
    oldbuckets unsafe.Pointer // previous bucket array of half the size, non-nil onl
    y when growing
    nevacuate  uintptr      // progress counter for evacuation (buckets less than
    this have been evacuated)

    // If both key and value do not contain pointers and are inline, then we mark bu
    cket
    // type as containing no pointers. This avoids scanning such maps.
    // However, bmap.overflow is a pointer. In order to keep overflow buckets
    // alive, we store pointers to all overflow buckets in hmap.overflow.
    // Overflow is used only if key and value do not contain pointers.
    // overflow[0] contains overflow buckets for hmap.buckets.
    // overflow[1] contains overflow buckets for hmap.oldbuckets.
    // The first indirection allows us to reduce static size of hmap.
    // The second indirection allows to store a pointer to the slice in hiter.
    overflow *[2]*[]*bmap
}
```

bucket内部



bucket内部

```
// A bucket for a Go map.
type bmap struct {
    tophash [bucketCnt]uint8
    // Followed by bucketCnt keys and then bucketCnt values.
    // NOTE: packing all the keys together and then all the values together makes the
    // code a bit more complicated than alternating key/value/key/value/... but it allows
    // us to eliminate padding which would be needed for, e.g., map[int64]int8.
    // Followed by an overflow pointer.
}
```

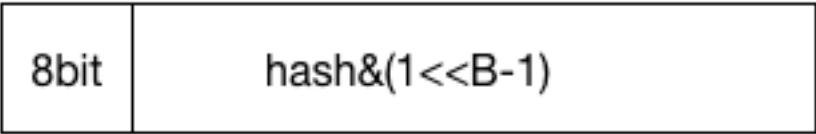
根据一个key得到value

```
func mapaccess1(t *maptype, h *hmap, key unsafe.Pointer) unsafe.Pointer
```

- *maptype为map的类型信息，是编译器在编译期静态生成的，里面包含了map的一些元信息，比如key和value的类型信息等
- *hmap为map的header，即map的引用
- key是一个通用的指针，代表了key的引用
- 返回值为一个指针，指向对应的value引用

hash计算找到bucket

那我们怎么访问到对应的bucket呢，我们需要得到对应key的hash值



bucket的hash

```
alg := t.key.alg
hash := alg.hash(key, uintptr(h.hash0))
m := uintptr(1)<<h.B - 1
b := (*bmap)(add(h.buckets, (hash&m)*uintptr(t.bucketsize)))
```

根据tophash和key定位到具体的bucket

- tophash可以快速试错，如果tophash不相等直接跳过
- tophash相等的话，根据key的比较来判断是否相等，如果相等则找到

- 如果当前bucket都试玩还没有找到，则调到下一个bucket

扩容

loadFactor	%overflow	bytes/entry	hitprobe	missprobe
4.00	2.13	20.77	3.00	4.00
4.50	4.05	17.30	3.25	4.50
5.00	6.85	14.77	3.50	5.00
5.50	10.55	12.94	3.75	5.50
6.00	15.27	11.67	4.00	6.00
6.50	20.90	10.79	4.25	6.50
7.00	27.14	10.15	4.50	7.00
7.50	34.03	9.73	4.75	7.50
8.00	41.10	9.40	5.00	8.00

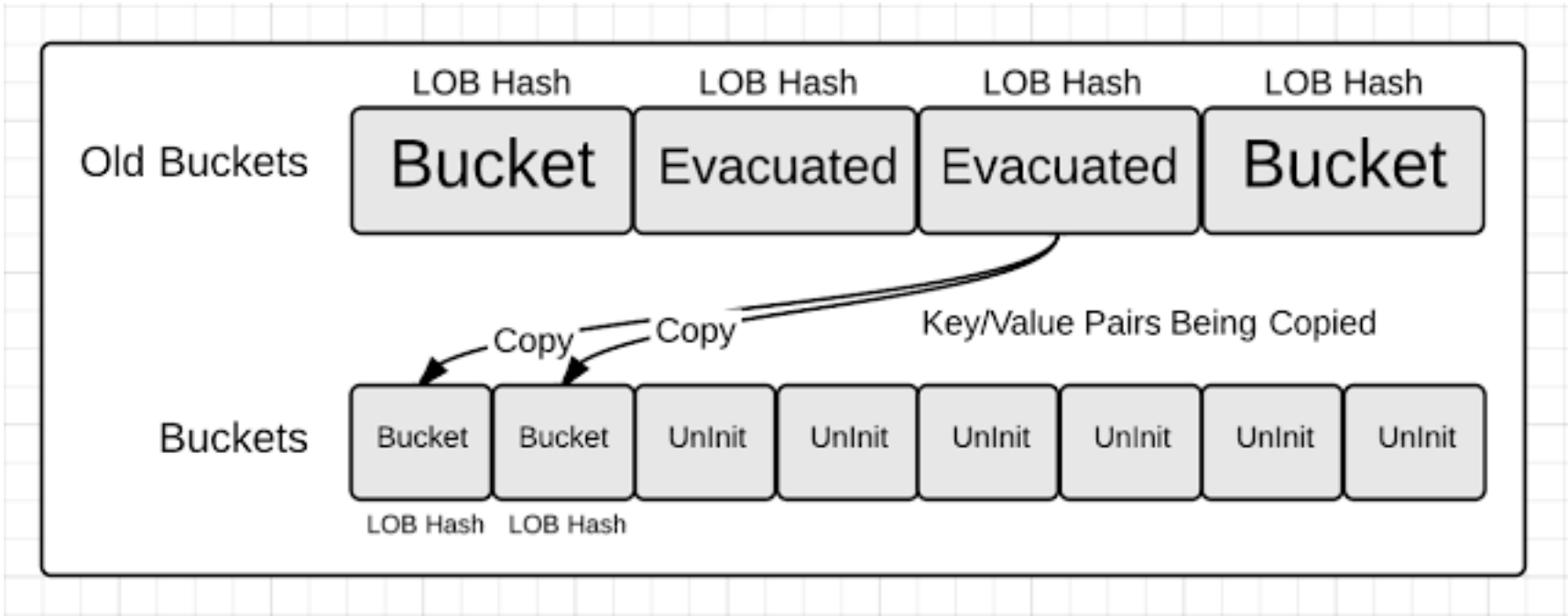
各个参数的意思：

- **%overflow** 溢出率，平均一个bucket有多少个kv的时候会溢出
- **bytes/entry** 平均存一个kv需要额外存储多少字节的数据
- **hitprobe** 找到一个存在的key平均需要找几下
- **missprobe** 找到一个不存在的key平均需要找几下

目前采用的是这一行：

| 6.50 | 20.90 | 10.79 | 4.25 | 6.50 |

迁移



迁移



golang (/nb/4253518)

[举报文章](#)

© 著作权归作者所有



icexin (/u/737a4b893f51)

写了 3103 字，被 24 人关注，获得了 15 个喜欢

(/u/737a4b893f51)3 字，被 24 人关注，获得了 15 个喜欢

+ 关注

如果觉得我的文章对您有用，请随意赞赏。您的支持将鼓励我继续创作！

赞赏支持



(http://cwb.assets.jianshu.io/notes/images/3930122/)






(/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~

被以下专题收入，发现更多相似内容

- 程序员 (/c/NEt52a?utm_source=desktop&utm_medium=notes-included-collection)
- Golang (/c/3e489dead7a7?utm_source=desktop&utm_medium=notes-included-collection)
- Go语言 (/c/c7634b78294d?utm_source=desktop&utm_medium=notes-included-collection)

推荐阅读

更多精彩内容 > (/)

一个多端口的ssh隧道 (/p/3ae69fba1820?utm_campaign=maleskine&utm_content=note&utm_medium=pc_all_hots&utm_source=recommendation)

最近女朋友遇到了一个问题，回家后想连接开发机，奈何vpn只能连接一个开发机，其他开发机没有访问权限。ssh本身有一个ssh-Lport1:ip2:port2的功能来监听本地的port1端口，把接收到的连接转发到ip2:port2

icexin (/u/737a4b893f51?utm_campaign=maleskine&utm_content=user&utm_medium=pc_all_hots&utm_source=recommendation)

基于SOCK5之上的HTTP代理 (/p/40aa4a77cb2e?utm_campaign=maleskine&utm_content=note&utm_medium=pc_all_hots&utm_source=recommendation)

在linux和mac下，好多程序都可以通过设置HTTP_PROXY和HTTPS_PROXY的方式来来进行HTTP代理，比如使用HTTP_PROXY=xxxxcurlexample.com就可以通过代理来访问example.com，这对于从网上下载东西加

icexin (/u/737a4b893f51?utm_campaign=maleskine&utm_content=user&utm_medium=pc_all_hots&utm_source=recommendation)

成长中的孤独，是一种不动声色的力量 (/p/54a2af5da93e?utm_campaign=maleskine&utm_content=note&utm_medium=pc_all_hots&utm_source=recommendation)

文 | 二九七月 01 “小佳，国庆节回家吗？” “不回了，我还要执行我的学习任务，等寒假再说吧。” 不用想了，这个八天小长假小佳肯定又是独自在学校里过了，

二九七月 (/u/496fc1fd5dc5?utm_campaign=maleskine&utm_content=user&utm_medium=pc_all_hots&utm_source=recommendation)

后来，我再也不找你聊天了 (/p/f5d2e7f11fc3?utm_ca... (/p/f5d2e7f11fc3?
时间真的是个神奇的东西，它可以让原本两个陌生的人变得熟悉，然后，又让两 utm_campaign=maleskine&utm_content=note&utm_
个已经熟悉的人变得陌生。01 我与阿泽相识于一场通话。那时候我在一家公司
丁小谢 (/u/5c446bff04a7?
utm_campaign=maleskine&utm_content=user&utm_medium=pc_all_hots&utm_source=recommendation)

有哪些发人深省、引人思考的好书 (/p/3a1d5ddb5ed6... (/p/3a1d5ddb5ed6?
个人很喜欢能给自己带来价值观震荡的书， 所以所谓“发人深省，引人深思”的书 utm_campaign=maleskine&utm_content=note&utm_
略知一二。 不过有个悖论，随着你看得这种类型书的增多， 你的价值观越来越
诸葛闹闹 (/u/7f03af68ad5e?
utm_campaign=maleskine&utm_content=user&utm_medium=pc_all_hots&utm_source=recommendation)