中国科学院大学
University of Chinese Academy of Sciences

计算机科学导论

实验报告

姓名： 唐嘉良

班级： 八班

组号： 第六组

学号： 2020K8009907032

实验名称： 班级排序实验

2021 年 6 月 5 号

一．首先给出信息隐藏实验最终的个人源代码

hide-2.go 代码如下：

```
package main
import (
   "flag"
   "fmt"
   "io/ioutil"
   "os"
 )
const (
   headerSize = 14 + 40 // standard size of header
   lengthFieldSize = 32 // size of occupancy in bmp for the length of hidden data
   infoUnitSize = 4 // size of occupancy in bmp for a byte of hidden data
)
// modify hides a integer to a byte array
func modify(txt int, pix []byte, size int) {
   for i := 0; i < size; i++ {
     pix[i]=byte(((txt)&(0B00000011<<(2*i)))>>(2*i)) + pix[i] & 0xFC
     }
}
var (
   srcImage string // input image path
   txtFile string // input text file path
   destImage string // output doctored image path
)
// init sets terminal option paramters
func init() {
   // DON'T modify this function!!!
   flag.StringVar(&srcImage, "i", "", "input image path.")
```

```
    flag.StringVar(&txtFile, "t", "", "input text file path.")
    flag.StringVar(&destImage, "o", "", "output doctored image path.")
  }
func main() {
    // parse terminal option parameters
    flag.Parse()
    if srcImage == "" || txtFile == "" || destImage == "" {
        flag.PrintDefaults()
        os.Exit(1)
    }
    // read input image to a byte array p
    p, err := ioutil.ReadFile(srcImage)
    if err != nil {
        fmt.Printf("Open image file failed, err = %v\n", err)
        os.Exit(1)
    }
    // read input text file to a byte array t
    t, err := ioutil.ReadFile(txtFile)
    if err != nil {
        fmt.Printf("Open text file failed, err = %v\n", err)
        os.Exit(1)
    }
    // check if the image can contain the content of the text
    offset := headerSize
    if lengthFieldSize+len(t)infoUnitSize > len(p[offset:]) {
        fmt.Println("The content of text is too long")
        os.Exit(1)
    }
      // save the text length to p
    modify(len(t), p[offset:], lengthFieldSize)
    // save the content of text to p
```

```
    for i := 0; i < len(t); i++ {
        offset = headerSize + lengthFieldSize + infoUnitSize*i
        modify(int(t[i]), p[offset:], infoUnitSize)
    }
    // save the modified p to destImage
    err = ioutil.WriteFile(destImage, p, 0666)
    if err != nil {
         fmt.Printf("Save doctored image failed, err = %v\n", err)
        os.Exit(1)
    }
}
```

show-2.go 代码如下：

```
package main
import (
    "flag"
    "io/ioutil"
    "os"
)
const (
    headerSize = 14 + 40 // standard size of header
    lengthFieldSize = 32 // size of occupancy in bmp for the length of hidden data
    infoUnitSize = 4 // size of occupancy in bmp for a byte of hidden data
)
var (
    image string // input doctor image path
    txtFile string // output text file path
)
// init sets terminal option paramters
func init() {
    // DON'T modify this function!!!
    flag.StringVar(&image, "i", "", "input image path.")
    flag.StringVar(&txtFile, "o", "", "output text file path.")
}
func show1(pix[]byte,size int)(int){
    var a int
    for i:=0;i<size;i++{
        a += (int(pix[i])&(0B11))<<(2*i)
    }
    return a
}
func show2(pix[]byte,size int)(byte){
     var b int=0
    for i:=0;i<size;i++{
```

```
        b += int((pix[i]&(0B00000011))<<(2*i))
    }
    return byte(b)
}
func main() {
    // parse terminal option parameters
    flag.Parse()
    if image == "" || txtFile == "" {
        flag.PrintDefaults()
        os.Exit(1)
        }
        p,_:=ioutil.ReadFile(image)
        offset:=headerSize
        var len int
        len=show1(p[offset:],lengthFieldSize)
        var t []byte=make([]byte,len)
        for i:=0;i<len;i++{
          offset=headerSize+lengthFieldSize+infoUnitSize*i
          t[i]=show2(p[offset:],infoUnitSize)
        }
        ioutil.WriteFile(txtFile,t,0666)
        // TODO: write your code here
    }
```

二．实验回顾

Develop hide and show programs to hide and restore the content of a text file in an image file. This project emphasizes **good programming practice and independent work**. Both hide and show programs should follow such practices and should be developed on your own. For detailed information, you can read text book 7.2 and Text Hider slide.

The basic principle of hiding is to replace the least significant two bits of one byte of the Pixel Array by two bits of a character. Your program must satisfy the following requirements:

– The restored text must have same content with the origin text.
– The header of the doctored image file must be same with the original image file.
– Only the least significant two bits of each byte of Pixel Array can be modified.
– The command line options of the hide and show program must be kept unchanged.

三．设计思路

对于以上代码，我的设计思路如下：

1.hide 和 show 实验思路

Algorithm for hide-2.go
Input: A BMP image and a text file.
Output: A doctored BMP image.
Steps:

1. Read doctored image into variable p.
2. Read the text file to hide into variable t.
3. Determine whether the text file is too big.
4. Hide the length of the text into variable p.
5. Hide every character into variable p.
6. Save modified p to doctored BMP image.

List 1 Algorithm for hide-2.go

Algorithm for show-2.go
Input: A doctored BMP image.
Output: An output text file.
Steps:

1. Read doctored image into variable p.
2. Read total length of text content from variable p.
3. Read every character from variable p and save the character into variable t.
4. Change variable t into text content.
5. Save text content to the output file.

List 2 Algorithm for show-2.go

2. 相关函数（hide 实验）

The modify function saves an integer value txt into a byte slice variable pix, 2 bits at a time, for a total of size iterations. My implementation of modify function is：

- First, take out the first six bits of pix[i] by doing “pix[i]&0xFC”
- Second, take out the target bit in txt by doing “ ((txt)&(0b00000011<<(2*i)))>>(2 * i)”
- Last, add bits together, and save the number into pix[i]

源代码给出如下：

```
func modify(txt int, pix []byte, size int) {

  for i := 0; i < size; i++ {

    pix[i]=byte(((txt)&(0B00000011<<(2*i)))>>(2*i)) + pix[i] & 0xFC

    }
```

List 3 Source code of hide-2.go modify function

The main function parses terminal command line parameters to get input image and text file path and output doctored image file path, and it also reads the image and text file, My implementation of main function is：

- First, judge whether the length of text file is too long.
- Second, hide the length of txt file into the picture by using “modify(len(t), p[offset:],lengthFieldSize)”
- Third, do a “for” loop, hide every character into the txt file by using “modify(int(t[i]), p[offset:], infoUnitSize)”
- Last, output the doctored image.

源代码给出如下：

```
func main() {
    // parse terminal option parameters
    flag.Parse()
    if srcImage == "" || txtFile == "" || destImage == "" {
        flag.PrintDefaults()
        os.Exit(1)
    }
    // read input image to a byte array p
    p, err := ioutil.ReadFile(srcImage)
    if err != nil {
        fmt.Printf("Open image file failed, err = %v\n", err)
        os.Exit(1)
    }
    // read input text file to a byte array t
    t, err := ioutil.ReadFile(txtFile)
    if err != nil {
        fmt.Printf("Open text file failed, err = %v\n", err)
        os.Exit(1)
    }
    // check if the image can contain the content of the text
```

```
    offset := headerSize
    if lengthFieldSize+len(t)infoUnitSize > len(p[offset:]) {
        fmt.Println("The content of text is too long")
        os.Exit(1)
    }
     // save the text length to p
    modify(len(t), p[offset:], lengthFieldSize)
    // save the content of text to p
    for i := 0; i < len(t); i++ {
        offset = headerSize + lengthFieldSize + infoUnitSize*i
        modify(int(t[i]), p[offset:], infoUnitSize)
    }
    // save the modified p to destImage
    err = ioutil.WriteFile(destImage, p, 0666)
    if err != nil {
         fmt.Printf("Save doctored image failed, err = %v\n", err)
        os.Exit(1)
    }
}
```

List 4 Source code of hide-2.go main function

3. 相关函数（show 实验）

The modify function saves an integer value txt into a byte slice variable pix, 2 bits at a time, for a total of size iterations. My implementation of modify function is：

- First, take out the hidden bit by using “pix[i]&(0B11)”
- Second, correct the number by doing bit operation “<<(2*i)”
- Last, do “for” loop and add the number together.

源代码给出如下：

```
func show1(pix[]byte,size int)(int){
    var a int
    for i:=0;i<size;i++{
         a += (int(pix[i])&(0B11))<<(2*i)
    }
    return a
}
func show2(pix[]byte,size int)(byte){
     var b int=0
    for i:=0;i<size;i++{
        b += int((pix[i]&(0B00000011))<<(2*i))
    }
    return byte(b)
}
```

List 5 Source code of show-2.go show function

The main function parses terminal command line parameters to get input image and output doctored text file path. My implementation of main function is：

- First, read the input image.
- Second, use “len=show1(p[offset:],lengthFieldSize)”read the length of targeted text first.
- Third, do a “for” loop, and read every character from image by using “t[j] = show2(p[offset:], infoUnitSize)” and record the result into t []byte.
- Last, output the text file

源代码给出如下：

```
func main() {
   // parse terminal option parameters
   flag.Parse()
   if image == "" || txtFile == "" {
       flag.PrintDefaults()
      os.Exit(1)
      }
      p,_:=ioutil.ReadFile(image)
      offset:=headerSize
      var len int
      len=show1(p[offset:],lengthFieldSize)
      var t []byte=make([]byte,len)
      for i:=0;i<len;i++{
        offset=headerSize+lengthFieldSize+infoUnitSize*i
        t[i]=show2(p[offset:],infoUnitSize)
      }
      ioutil.WriteFile(txtFile,t,0666)
      // TODO: write your code here
   }
```

List 6 Source code of show-2.go main function

4. 每个像素隐藏 1 个或 4 个比特

a) For hiding and showing text in the least significant one bits:

Chang “lengthFieldSize” in hide-2.go from 32 to 64

Change “infoUnitSize” in hide-2.go from 4 to 8

Change the modify function in hide-2.go into

```
func modify(txt int, pix []byte, size int) {

   for i := 0; i < size; i++ {

     pix[i]=byte(((txt)&(0B1<<(i)))>>(i)) + pix[i] & 0xFE

     }
```

Change “infoUnitSize” in show-2.go from 4 to 8

Change the show function in show-2.go into

```
func show1(pix[]byte,size int)(int){
    var a int
    for i:=0;i<size;i++{
        a += (int(pix[i])&(0B1))<<(i)
    }
    return a
}
func show2(pix[]byte,size int)(byte){
     var b int=0
    for i:=0;i<size;i++{
       b += int((pix[i]&(0B1))<<(i))
    }
    return byte(b)
}
```

b) For hiding and showing text in the least significant four bits:

Chang “lengthFieldSize” in hide-2.go from 32 to 16

Change “infoUnitSize” in hide-2.go from 4 to 2

Change the modify function in hide-2.go into

```
func modify(txt int, pix []byte, size int) {

   for i := 0; i < size; i++ {

    pix[i]=byte(((txt)&(0B1111<<(4*i)))>>(4*i)) + pix[i] & 0xF0

    }
```

Change “infoUnitSize” in show-2.go from 4 to 2

Change the show function in show-2.go into

```
func show1(pix[]byte,size int)(int){
    var a int
    for i:=0;i<size;i++{
        a += (int(pix[i])&(0B1111))<<(4i)
    }
    return a
}
func show2(pix[]byte,size int)(byte){
     var b int=0
    for i:=0;i<size;i++{
       b += int((pix[i]&(0B1111))<<(4i))
    }
    return byte(b)
}
```

四．良好的编程习惯

1.Use descriptive names for variables and constants: hide-2.go and show-2.go uses descriptive names. For example, the constant headerSize means the standard size of header, the constant lengthFieldSize means size of occupancy in bmp for the length of hidden data, the constant infoUnitSize means size of occupancy in bmp for a byte of hidden data, the variable len means the length of hidden data. What's more, the names of slice serve as description. The name of pix []byte and p []byte mean pixel. The name of t []byte means text file. The variable offset means the offset location to do the function.

2.Write descriptive sentences/phrases to explain the usage of variables or next few command lines: hide-2.go and show-2.go uses descriptive sentences. For example, after defining the constants: headerSize, lengthFieldSize and infoUniSize, I use explanatory phrases such as "standard size of header" "size of occupancy in bmp for the length of hidden data" and "size of occupancy in bmp for a byte of hidden data". What's more, after read the input image in hide-2.go, I write the sentence "// read input image to a byte array p" to remind myself.

3.Use function to simplify the program: In the hide-2.go, I use the function "modify" to hide the character independently.

In the show-2.go, I use the function "show1"and "show2" to show the character independently.

五．实验评价

Picture 1: original image

Picture 2: doctored image

```
/cs101/code > diff HungLouMeng.txt restoredHungLouMeng.txt
/cs101/code >
```

Picture 3: Compare original text file and restored text file with diff command (HungLouMeng.txt is the original text file, restoredHungLouMeng.txt is the restored text file)

我们将文本及图片换成其它同格式的文本及图片，再次验证，仍然成功。因此我们可以断言我们的代码是鲁棒的。