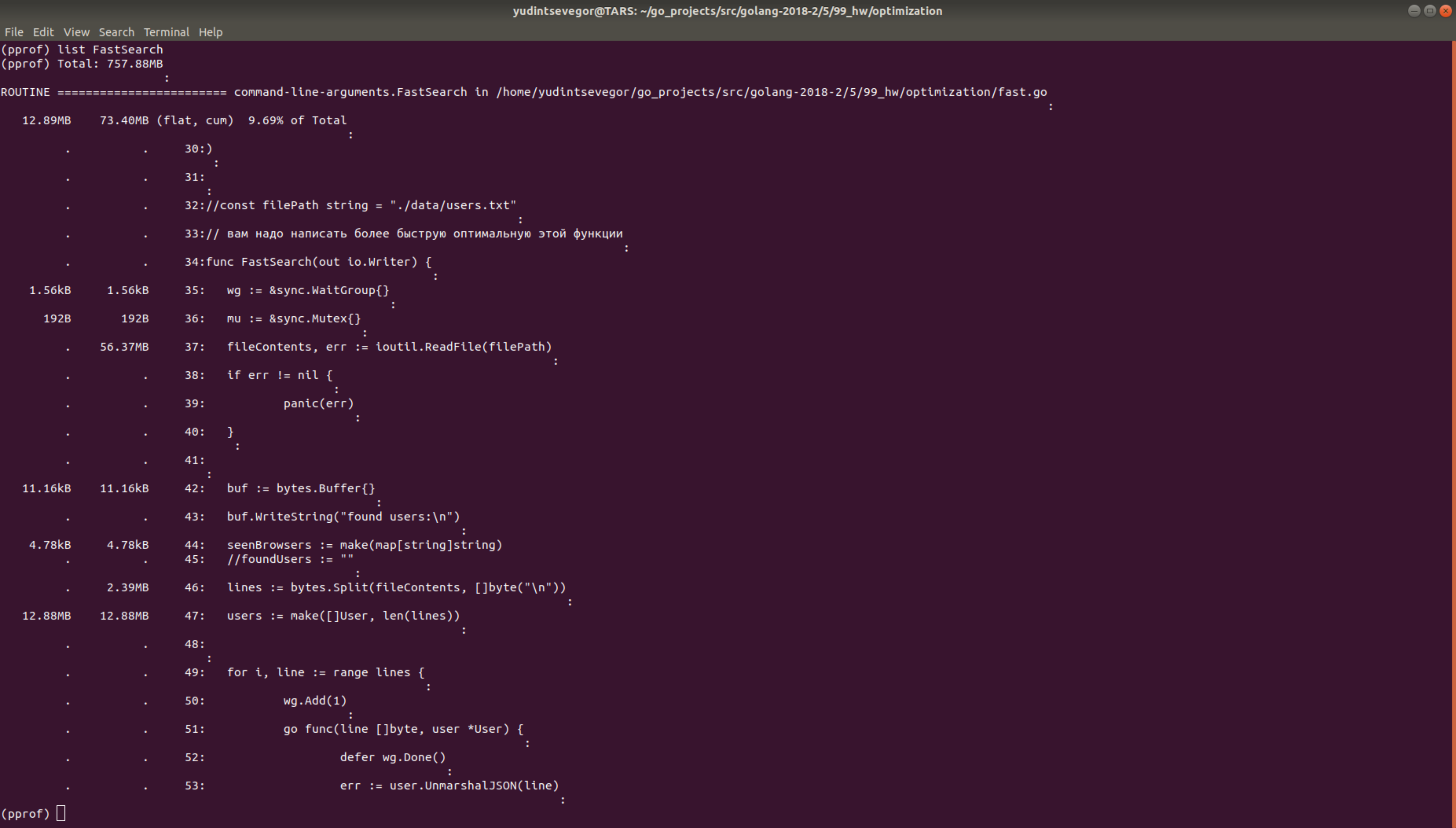
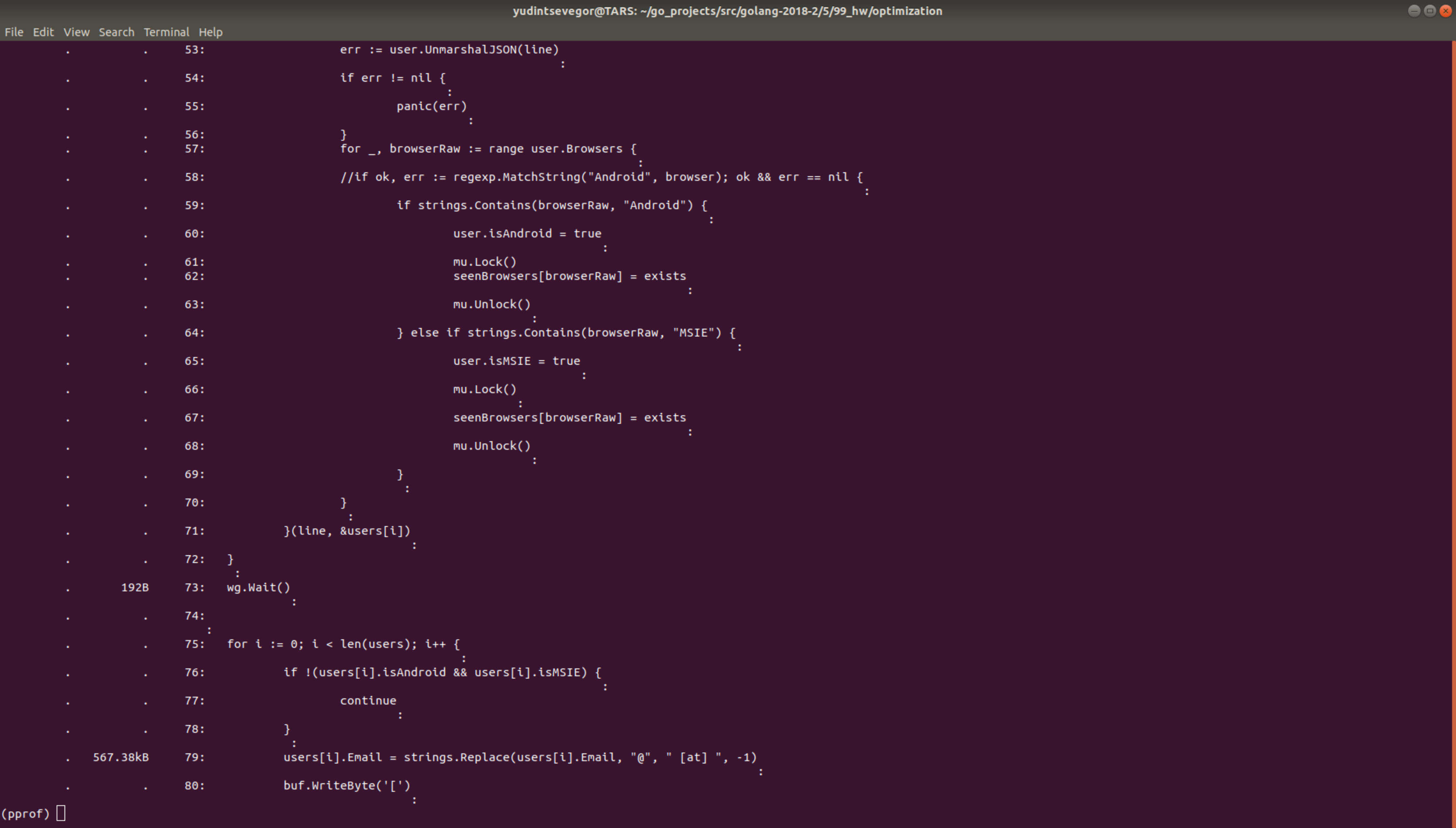
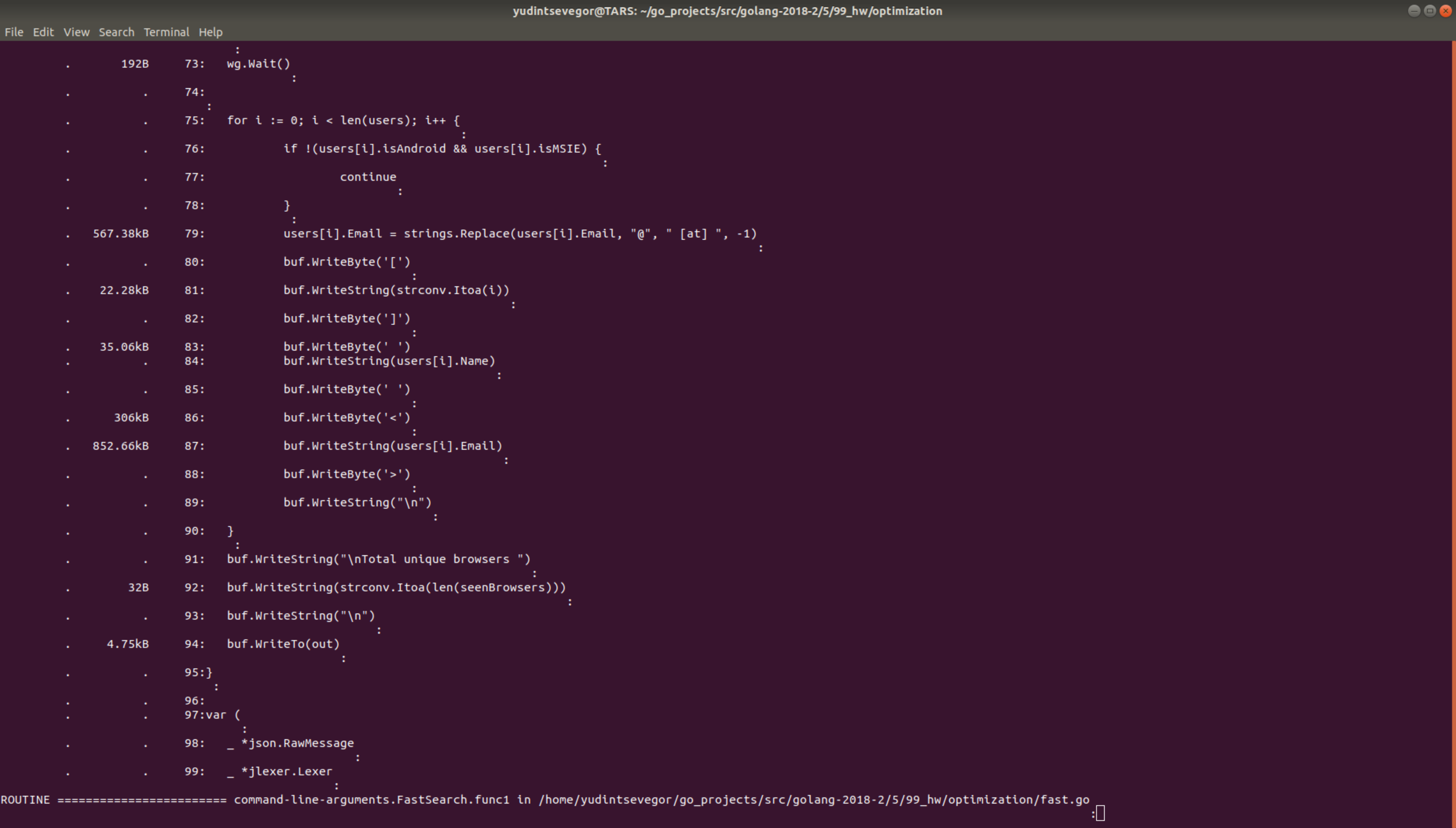

yudintsevegor@TARS: ~/go_projects/src/golang-2018-2/5/99_hw/optimization
File Edit View Search Terminal Help

```
(pprof) list FastSearch
(pprof) Total: 757.88MB
                       :
ROUTINE ======================== command-line-arguments.FastSearch in /home/yudintsevegor/go_projects/src/golang-2018-2/5/99_hw/optimization/fast.go
                                                                                                                                                    :
   12.89MB    73.40MB (flat, cum)  9.69% of Total
                                                 :
         .          .     30:)
                              :
         .          .     31:
                             :
         .          .     32://const filePath string = "./data/users.txt"
                                                                         :
         .          .     33:// вам надо написать более быструю оптимальную этой функции
                                                                                        :
         .          .     34:func FastSearch(out io.Writer) {
                                                            :
    1.56kB     1.56kB     35:   wg := &sync.WaitGroup{}
                                                      :
      192B       192B     36:   mu := &sync.Mutex{}
                                                  :
         .    56.37MB     37:   fileContents, err := ioutil.ReadFile(filePath)
                                                                             :
         .          .     38:   if err != nil {
                                              :
         .          .     39:           panic(err)
                                                 :
         .          .     40:   }
                                 :
         .          .     41:
                             :
   11.16kB    11.16kB     42:   buf := bytes.Buffer{}
                                                    :
         .          .     43:   buf.WriteString("found users:\n")
                                                                :
    4.78kB     4.78kB     44:   seenBrowsers := make(map[string]string)
         .          .     45:   //foundUsers := ""
                                                 :
         .     2.39MB     46:   lines := bytes.Split(fileContents, []byte("\n"))
                                                                               :
   12.88MB    12.88MB     47:   users := make([]User, len(lines))
                                                                :
         .          .     48:
                             :
         .          .     49:   for i, line := range lines {
                                                          :
         .          .     50:           wg.Add(1)
                                                :
         .          .     51:           go func(line []byte, user *User) {
                                                                         :
         .          .     52:                   defer wg.Done()
                                                                :
         .          .     53:                   err := user.UnmarshalJSON(line)
                                                                                :
(pprof) 
```

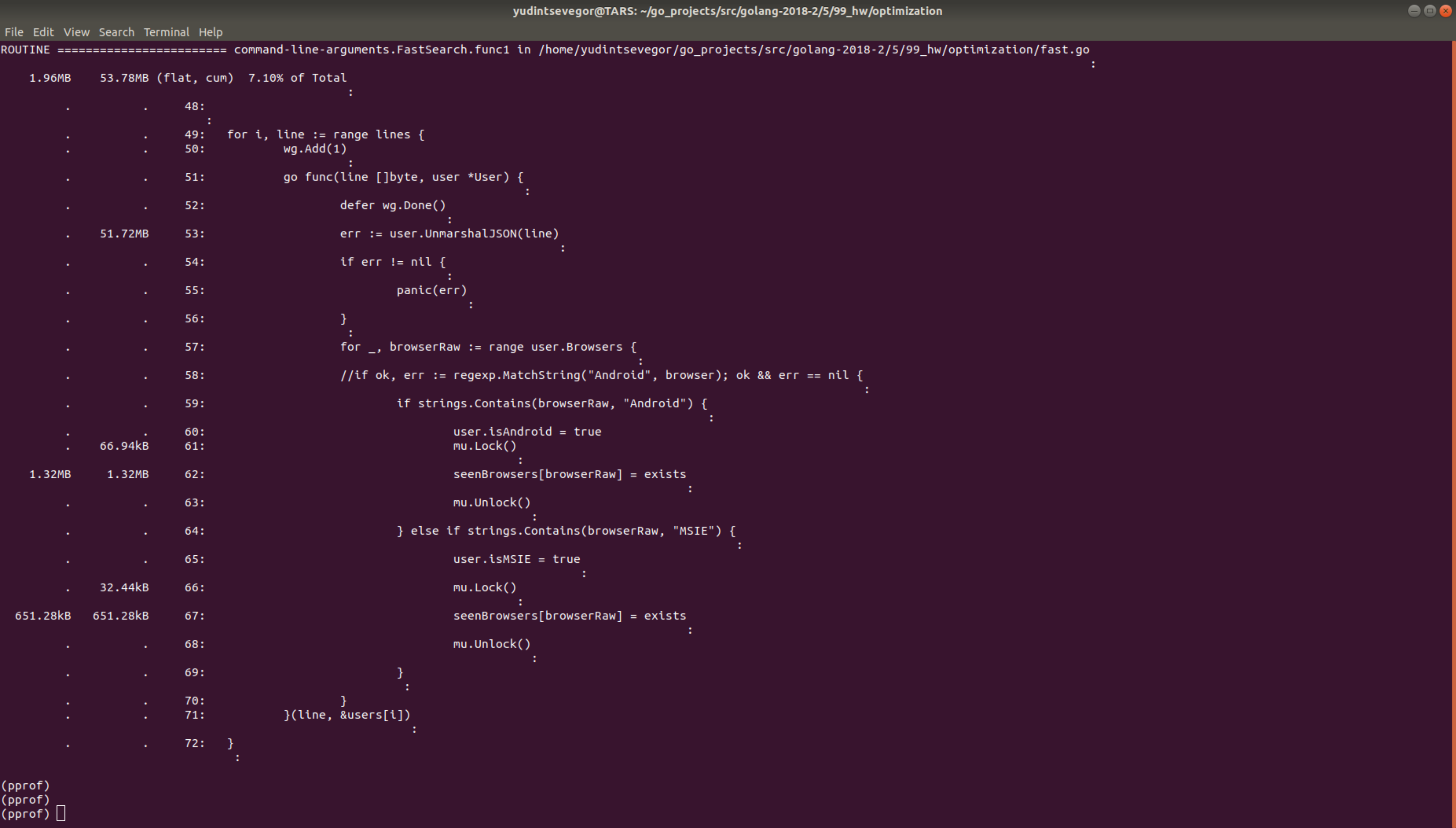


yudintsevegor@TARS: ~/go_projects/src/golang-2018-2/5/99_hw/optimization

```
         .          .     53:                   err := user.UnmarshalJSON(line)
                                                                                :
         .          .     54:                   if err != nil {
                                                                :
         .          .     55:                           panic(err)
                                                                   :
         .          .     56:                   }
         .          .     57:                   for _, browserRaw := range user.Browsers {
                                                                                           :
         .          .     58:                   //if ok, err := regexp.MatchString("Android", browser); ok && err == nil {
                                                                                                                           :
         .          .     59:                           if strings.Contains(browserRaw, "Android") {
                                                                                                     :
         .          .     60:                                   user.isAndroid = true
                                                                                      :
         .          .     61:                                   mu.Lock()
         .          .     62:                                   seenBrowsers[browserRaw] = exists
                                                                                                  :
         .          .     63:                                   mu.Unlock()
                                                                            :
         .          .     64:                           } else if strings.Contains(browserRaw, "MSIE") {
                                                                                                         :
         .          .     65:                                   user.isMSIE = true
                                                                                   :
         .          .     66:                                   mu.Lock()
                                                                          :
         .          .     67:                                   seenBrowsers[browserRaw] = exists
                                                                                                  :
         .          .     68:                                   mu.Unlock()
                                                                            :
         .          .     69:                           }
                                                          :
         .          .     70:                   }
                                                  :
         .          .     71:           }(line, &users[i])
                                                           :
         .          .     72:   }
                                  :
         .       192B     73:   wg.Wait()
                                          :
         .          .     74:
                              :
         .          .     75:   for i := 0; i < len(users); i++ {
                                                                  :
         .          .     76:           if !(users[i].isAndroid && users[i].isMSIE) {
                                                                                      :
         .          .     77:                   continue
                                                         :
         .          .     78:           }
                                          :
         .   567.38kB     79:           users[i].Email = strings.Replace(users[i].Email, "@", " [at] ", -1)
                                                                                                            :
         .          .     80:           buf.WriteByte('[')
                                                           :
(pprof) 
```


yudintsevegor@TARS: ~/go_projects/src/golang-2018-2/5/99_hw/optimization
File
Edit
View
Search
Terminal
Help

```
                                  :
         .       192B     73:   wg.Wait()
                                          :
         .          .     74:
                              :
         .          .     75:   for i := 0; i < len(users); i++ {
                                                                  :
         .          .     76:           if !(users[i].isAndroid && users[i].isMSIE) {
                                                                                      :
         .          .     77:                   continue
                                                        :
         .          .     78:           }
                                          :
         .   567.38kB     79:           users[i].Email = strings.Replace(users[i].Email, "@", " [at] ", -1)
                                                                                                             :
         .          .     80:           buf.WriteByte('[')
                                                          :
         .    22.28kB     81:           buf.WriteString(strconv.Itoa(i))
                                                                        :
         .          .     82:           buf.WriteByte(']')
                                                          :
         .    35.06kB     83:           buf.WriteByte(' ')
         .          .     84:           buf.WriteString(users[i].Name)
                                                                      :
         .          .     85:           buf.WriteByte(' ')
                                                          :
         .      306kB     86:           buf.WriteByte('<')
                                                          :
         .   852.66kB     87:           buf.WriteString(users[i].Email)
                                                                       :
         .          .     88:           buf.WriteByte('>')
                                                          :
         .          .     89:           buf.WriteString("\n")
                                                             :
         .          .     90:   }
                                  :
         .          .     91:   buf.WriteString("\nTotal unique browsers ")
                                                                           :
         .       32B      92:   buf.WriteString(strconv.Itoa(len(seenBrowsers)))
                                                                                :
         .          .     93:   buf.WriteString("\n")
                                                     :
         .     4.75kB     94:   buf.WriteTo(out)
                                                :
         .          .     95:}
                              :
         .          .     96:
         .          .     97:var (
                                   :
         .          .     98:   _ *json.RawMessage
                                                  :
         .          .     99:   _ *jlexer.Lexer
                                               :
ROUTINE ======================== command-line-arguments.FastSearch.func1 in /home/yudintsevegor/go_projects/src/golang-2018-2/5/99_hw/optimization/fast.go
                                                                                                                                                          :
```



yudintsevegor@TARS: ~/go_projects/src/golang-2018-2/5/99_hw/optimization

```
ROUTINE ======================= command-line-arguments.FastSearch.func1 in /home/yudintsevegor/go_projects/src/golang-2018-2/5/99_hw/optimization/fast.go
    1.96MB    53.78MB (flat, cum)  7.10% of Total
         .          .     48:
         .          .     49:   for i, line := range lines {
         .          .     50:           wg.Add(1)
         .          .     51:           go func(line []byte, user *User) {
         .          .     52:                   defer wg.Done()
         .    51.72MB     53:                   err := user.UnmarshalJSON(line)
         .          .     54:                   if err != nil {
         .          .     55:                           panic(err)
         .          .     56:                   }
         .          .     57:                   for _, browserRaw := range user.Browsers {
         .          .     58:                   //if ok, err := regexp.MatchString("Android", browser); ok && err == nil {
         .          .     59:                           if strings.Contains(browserRaw, "Android") {
         .          .     60:                                   user.isAndroid = true
         .    66.94kB     61:                                   mu.Lock()
    1.32MB     1.32MB     62:                                   seenBrowsers[browserRaw] = exists
         .          .     63:                                   mu.Unlock()
         .          .     64:                           } else if strings.Contains(browserRaw, "MSIE") {
         .          .     65:                                   user.isMSIE = true
         .    32.44kB     66:                                   mu.Lock()
  651.28kB   651.28kB     67:                                   seenBrowsers[browserRaw] = exists
         .          .     68:                                   mu.Unlock()
         .          .     69:                           }
         .          .     70:                   }
         .          .     71:           }(line, &users[i])
         .          .     72:   }

(pprof)
(pprof)
(pprof)
```