How do I speed up this _for_ loop? With data.table + lapply?

2

This code generates a dataset similar to my own:

```
df <- c(seq(as.Date("2012-01-01"), as.Date("2012-01-10"), "days"))
  df <- as.data.frame(df)
  df <- rbind(df, df)

id <- c(rep.int(1, 10), rep.int(2, 10))
  id <- as.data.frame(id)

cnt <- c(1:3, 0, 0, 4, 5:8, 0, 1, 0, 1:7)
  cnt <- as.data.frame(cnt)

df <- cbind(id, df, cnt)
  names(df) <- c("id", "date", "cnt")

df$date[df$date == "2012-01-10"] <- "2012-01-20"
```

I'm trying to find the sum of variable 'cnt' that has occurred within the last 7 days. Sometimes dates are not continuous (see the last date in the preceeding 'df') -- by id.

Here's the loop:

```
system.time(

  for(i in 1:length(df$date)) {
    df$cnt.weekly[i] <-
      sum(df$cnt[which((df$date == df$date[i] - 1) & df$id == df$id[i])],
          df$cnt[which((df$date == df$date[i] - 2) & df$id == df$id[i])],
          df$cnt[which((df$date == df$date[i] - 3) & df$id == df$id[i])],
          df$cnt[which((df$date == df$date[i] - 4) & df$id == df$id[i])],
          df$cnt[which((df$date == df$date[i] - 5) & df$id == df$id[i])],
          df$cnt[which((df$date == df$date[i] - 6) & df$id == df$id[i])])})
```

I'm ultimately running this on an 8 million row data.frame (thousands of ids), so while the toy is fast here it is very slow in practice.

I've had very good luck with the data.table package in other parts of the code, but I can't figure out how to get it to work here. Maybe lapply inside of data.table?

Thanks in advance!

r | data.table | lapply

link | improve this question

asked **May 23 at 13:39**
Topher
45 ●4

tagged

r × 14064

data.table × 210

lapply × 63

asked **1 month ago**
viewed **70 times**
active **1 month ago**

Linked

Speed up the loop operation in R

Mean of 50 most recent entries in R

Related

Try `rollapply` ? Also, store your `df$id==df$id[i]` comparison so it doesn't get recalculated each time. Also, take advantage of the fact that if `i-6` is within a week, then `i-5`, `i-4` etc. are also. See also: stackoverflow.com/questions/2908822/... – gsk3 May 23 at 13:50

Thank you, great suggestions. – Topher May 23 at 14:09

feedback

1 Answer

active oldest **votes**

5

How about :

```
> DT = as.data.table(df)
> DT
      id       date cnt
 [1,]  1 2012-01-01   1
 [2,]  1 2012-01-02   2
 [3,]  1 2012-01-03   3
 [4,]  1 2012-01-04   0
 [5,]  1 2012-01-05   0
 [6,]  1 2012-01-06   4
 [7,]  1 2012-01-07   5
 [8,]  1 2012-01-08   6
 [9,]  1 2012-01-09   7
[10,]  1 2012-01-20   8
[11,]  2 2012-01-01   0
[12,]  2 2012-01-02   1
[13,]  2 2012-01-03   0
[14,]  2 2012-01-04   1
[15,]  2 2012-01-05   2
[16,]  2 2012-01-06   3
[17,]  2 2012-01-07   4
[18,]  2 2012-01-08   5
[19,]  2 2012-01-09   6
[20,]  2 2012-01-20   7
```

Then cumulate within group. This step is currently ugly, but `:=` by group (soon to be in 1.8.1) will tidy this up.

```
> DT[,cumcnt:=DT[,cumsum(cnt),by=id][[2]]]
      id       date cnt cumcnt
 [1,]  1 2012-01-01   1      1
 [2,]  1 2012-01-02   2      3
 [3,]  1 2012-01-03   3      6
 [4,]  1 2012-01-04   0      6
 [5,]  1 2012-01-05   0      6
 [6,]  1 2012-01-06   4     10
 [7,]  1 2012-01-07   5     15
 [8,]  1 2012-01-08   6     21
 [9,]  1 2012-01-09   7     28
[10,]  1 2012-01-20   8     36
[11,]  2 2012-01-01   0      0
[12,]  2 2012-01-02   1      1
[13,]  2 2012-01-03   0      1
[14,]  2 2012-01-04   1      2
[15,]  2 2012-01-05   2      4
[16,]  2 2012-01-06   3      7
[17,]  2 2012-01-07   4     11
[18,]  2 2012-01-08   5     16
[19,]  2 2012-01-09   6     22
[20,]  2 2012-01-20   7     29
```

Now join to 7 days ago, allowing for irregular dates :

- How to create a column containing a string of stars to inidcate levels of a factor in a data frame in R
- setting levels inside lapply loop in r
- How to return a data.frame with a given name from a function?
- Using lapply with changing arguments
- Proper/fastest way to reshape a data.table
- How to use lapply with a formula?
- Convert column classes in data.table
- What are the restrictions for the column classes in data.table?
- efficient row-wise operations on a data.table
- Using get inside lapply, inside a function
- Loop through columns in a data.table and transform those columns
- Aggregate over categories that contain NAs with ddply and lapply?
- Can't access items after an lapply
- How does lapply really work - lapply dcast?
- How do you delete a column in data.table?
- “Loop through” data.table to calculate conditional averages
- Access lapply index names inside FUN
- function := not found from package data.table?
- Call list by name from loop or lapply, in R
- R: how to delete colums in a data.table?
- How would you translate this into data.table package language in R?
- R: using data.table := operations to calculate new columns
- How to delete a row by reference in R data.table?
- How to best join one column of a data.table with another column of the same data.table?
- lapply and do.call running very slowly?

```
> setkey(DT,id,date)
> DT[,before7dayago:=DT[SJ(id,date-7),cumcnt,roll=TRUE,mult="last"]]
      id       date cnt cumcnt before7dayago
 [1,]  1 2012-01-01   1      1            NA
 [2,]  1 2012-01-02   2      3            NA
 [3,]  1 2012-01-03   3      6            NA
 [4,]  1 2012-01-04   0      6            NA
 [5,]  1 2012-01-05   0      6            NA
 [6,]  1 2012-01-06   4     10            NA
 [7,]  1 2012-01-07   5     15            NA
 [8,]  1 2012-01-08   6     21             1
 [9,]  1 2012-01-09   7     28             3
[10,]  1 2012-01-20   8     36            28
[11,]  2 2012-01-01   0      0            NA
[12,]  2 2012-01-02   1      1            NA
[13,]  2 2012-01-03   0      1            NA
[14,]  2 2012-01-04   1      2            NA
[15,]  2 2012-01-05   2      4            NA
[16,]  2 2012-01-06   3      7            NA
[17,]  2 2012-01-07   4     11            NA
[18,]  2 2012-01-08   5     16             0
[19,]  2 2012-01-09   6     22             1
[20,]  2 2012-01-20   7     29            22
```

And finally subtract one from the other.

```
> DT[,`7daysum`:=cumcnt-before7dayago]
      id       date cnt cumcnt before7dayago 7daysum
 [1,]  1 2012-01-01   1      1            NA       NA
 [2,]  1 2012-01-02   2      3            NA       NA
 [3,]  1 2012-01-03   3      6            NA       NA
 [4,]  1 2012-01-04   0      6            NA       NA
 [5,]  1 2012-01-05   0      6            NA       NA
 [6,]  1 2012-01-06   4     10            NA       NA
 [7,]  1 2012-01-07   5     15            NA       NA
 [8,]  1 2012-01-08   6     21             1       20
 [9,]  1 2012-01-09   7     28             3       25
[10,]  1 2012-01-20   8     36            28        8
[11,]  2 2012-01-01   0      0            NA       NA
[12,]  2 2012-01-02   1      1            NA       NA
[13,]  2 2012-01-03   0      1            NA       NA
[14,]  2 2012-01-04   1      2            NA       NA
[15,]  2 2012-01-05   2      4            NA       NA
[16,]  2 2012-01-06   3      7            NA       NA
[17,]  2 2012-01-07   4     11            NA       NA
[18,]  2 2012-01-08   5     16             0       16
[19,]  2 2012-01-09   6     22             1       21
[20,]  2 2012-01-20   7     29            22        7
```

That should be very fast.

link | improve this answer

answered **May 23 at 15:39**

Matthew Dowle

5,009 ●8 ●28

1 Bravo! Thank you, this works amazing. Looks like I need to dig into data.table deeper. I wasn't aware of the 'by' function, though I just started working with data.table. – Topher May 23 at 16:51

feedback

Your Answer

log in or

Name

Email

Home Page

Post Your Answer

By posting your answer, you agree to the privacy policy and terms of service.

Not the answer you're looking for? Browse other questions tagged r data.table lapply or ask your own question.

question feed