Subject: Re: Splitting An Array Of Strings Without Using Loops
Posted by mchinand on Fri, 25 Jul 2003 05:04:07 GMT
View Forum Message <> Reply to Message

In article <e5624c04.0307240935.7234e53@posting.google.com>,
Darrick White <darrick.white@med.ge.com> wrote:
> This is probably simple, but I'm having a time trying to figure it
> out. I want to be able to split an array of strings without using
> loops.
>
> Example:
> dataPoints is an array of strings with N elements
> The format of each element within dataPoints is "x:y1:y2:y3:yn". More
> than likely, the data will be in the format of x:y".
>
> This array will become data points (the first element is always
> considered the x coordinate): (x,y) = 1,23. In case of multiple
> points (2:21:34:54), the data will look like: (2,21), (2,34), (2,54).
>
> I need a way to take:
> dataPoints[0] = 1:23
> dataPoints[1] = 2:32
> dataPoints[2] = 3:30
> dataPoints[3] = 4:45
>
>
> and create
> points[2,4]
> 1 23
> 2 32
> 3 30
> 4 45
>
> -Darrick

For the simpler case of just 'x:y' pairs the following show work:

IDL> data=['1:23','2:32','3:30','4:45']

put it into one big string

IDL> datajoin=strjoin(data,':')

IDL> print, datajoin
1:23:2:32:3:30:4:45

Then split it up and reform it into a two by four array

```
IDL> dataint=reform(fix(strsplit(datajoin,':',/extract)),2,4)
IDL> print, dataint
      1      23
      2      32
      3      30
      4      45
```

Hope that helps,

--Mike

--
Michael Chinander
m-chinander@uchicago.edu
Department of Radiology
University of Chicago

Subject: Re: Splitting An Array Of Strings Without Using Loops
Posted by darrick.white on Fri, 25 Jul 2003 13:38:06 GMT
View Forum Message <> Reply to Message

mchinand@midway.uchicago.edu (Mike Chinander) wrote in message news:<bT2Ua.33$Z4.10135@news.uchicago.edu>...
> In article <e5624c04.0307240935.7234e53@posting.google.com>,
> Darrick White <darrick.white@med.ge.com> wrote:
>> This is probably simple, but I'm having a time trying to figure it
>> out. I want to be able to split an array of strings without using
>> loops.
>>
>> Example:
>> dataPoints is an array of strings with N elements
>> The format of each element within dataPoints is "x:y1:y2:y3:yn". More
>> than likely, the data will be in the format of x:y".
>>
>> This array will become data points (the first element is always
>> considered the x coordinate): (x,y) = 1,23. In case of multiple
>> points (2:21:34:54), the data will look like: (2,21), (2,34), (2,54).
>>
>> I need a way to take:
>> dataPoints[0] = 1:23
>> dataPoints[1] = 2:32
>> dataPoints[2] = 3:30
>> dataPoints[3] = 4:45
>>
>>
>> and create

>> points[2,4]
>> 1 23
>> 2 32
>> 3 30
>> 4 45
>>
>> -Darrick
>
> For the simpler case of just 'x:y' pairs the following show work:
>
> IDL> data=['1:23','2:32','3:30','4:45']
>
> put it into one big string
>
> IDL> datajoin=strjoin(data,':')
>
> IDL> print, datajoin
> 1:23:2:32:3:30:4:45
>
> Then split it up and reform it into a two by four array
>
> IDL> dataint=reform(fix(strsplit(datajoin,':',/extract)),2,4)
> IDL> print, dataint
> 1 23
> 2 32
> 3 30
> 4 45
>
>
> Hope that helps,
>
> --Mike

Thanks for the reply. Yes, that works, and that's what I am doing now. However, there may be cases when the data may look like this:
['1:23','2:32:43','3:30:54','4:45']

In this case, I would like the data formatted into a 3x4 array

1 23 NaN
2 32 43
3 30 54
4 45 Nan

This is where I'm having a little trouble.

Thanks
-Darrick

Subject: Re: Splitting An Array Of Strings Without Using Loops
Posted by R.Bauer on Wed, 30 Jul 2003 18:25:45 GMT
View Forum Message <> Reply to Message

Darrick White wrote:
>> Dear Darrick,
>>
>> here is a second solution using reads.
>>
>> pro test
>> data=['1:23','2:32','3:30','4:45']
>>
>> s={x:bytarr(1),s:bytarr(1),y:bytarr(2)}
>> s=replicate(s,4)
>>
>> reads,byte(data),s
>>
>> print,string(s.x)
>> print,string(s.y)
>> end
>>
>> IDL> 1 2 3 4
>> IDL> 23 32 30 45
>
>
>
> It looks like I'm not explaining my problem clearly. For instance,
> the following sets of data are valid inputs to my application:
>
> 1) data=['1:23','2:32','3:30','4:45']
> 2) data=['12:23','22:32:34:45','32:30','42:45:90']
> 3) data=['100:23','200:32','300:30','400:45']
> 4) data=['1:23:2','2:32:2','3:30:2','4:45:2']
>

ok
do the followings. reads is very powerful (not only histogram)

data=replace_string(data,':',',')

1) data=replace_string(data,':',',')

```
   a=intarr(2)
   b=intarr(2)
   c=intarr(2)
   d=intarr(2)
reads,data,a,b,c,d
print,a,b,c,d
      1      23
```

```
       2      32
       3      30
       4      45

2) data=replace_string(data,':',',')
   a=intarr(2)
   b=intarr(4)
   c=intarr(2)
   d=intarr(3)
reads,data,a,b,c,d
print,a,b,c,d
      12      23
      22      32      34      45
      32      30
      42      45      90

3) data=replace_string(data,':',',')
   a=intarr(2)
   b=intarr(2)
   c=intarr(2)
   d=intarr(2)
reads,data,a,b,c,d
print,a,b,c,d
     100      23
     200      32
     300      30
     400      45


4) data=replace_string(data,':',',')

   a=intarr(3)
   b=intarr(3)
   c=intarr(3)
   d=intarr(3)
reads,data,a,b,c,d
print,a,b,c,d
       1      23       2
       2      32       2
       3      30       2
       4      45       2
```

You'll find replace_string at our library:
http://www.fz-juelich.de/icg/icg-i/idl_icglib/idl_source/idl _html/dbase/download/replace_string.tgz

or as idl5.6 binary

http://www.fz-juelich.de/icg/icg-i/idl_icglib/idl_source/idl _html/dbase/download/replace_string.sav

Please have a look for licensing and further routines at
http://www.fz-juelich.de/icg/icg-i/idl_icglib/idl_lib_intro. html

If you write yourself a function to count the ":" signs +1 then this
could be easy automated.

regards

Reimar

> The resulting transformation would like this for both:
>
> 1) print, intarr(2,4)
> 1 23
> 2 32
> 3 30
> 4 45
>
> 2) print, intarr(4,4)
> 12 23 NaN NaN
> 22 32 34 45
> 32 30 NaN NaN
> 42 45 90 NaN
>
> 3) print, intarr(2,4)
> 100 23
> 200 32
> 300 30
> 400 45
>
> 4) print, intarr(3,4)
> 1 23 2
> 2 32 2
> 3 30 2
> 4 45 2
>
> Is there a way (not knowing what data set input is used) to transform
> my data into the corresponding result array? Note: For
> transformation #2 above, I need to append each point to my new array.
> If the array dimensions don't match, I need to fill in those missing
> elements with 'NaN'.
>
> Thanks

> -Darrick

--
Reimar Bauer

Institut fuer Stratosphaerische Chemie (ICG-I)
Forschungszentrum Juelich
email: R.Bauer@fz-juelich.de
---------- -------
a IDL library at ForschungsZentrum Juelich
http://www.fz-juelich.de/icg/icg-i/idl_icglib/idl_lib_intro. html
========== =======

Subject: Re: Splitting An Array Of Strings Without Using Loops
Posted by darrick.white on Thu, 31 Jul 2003 15:39:36 GMT
View Forum Message <> Reply to Message

JD Smith <jdsmith@as.arizona.edu> wrote in message
news:<pan.2003.07.28.23.00.27.659018.15959@as.arizona.edu>...
> On Mon, 28 Jul 2003 11:24:50 -0700, Rick Towler wrote:
>
>
>> "Darrick White" wrote...
>>
>>> It looks like I'm not explaining my problem clearly.
>
>>> Is there a way (not knowing what data set input is used) to transform
>>> my data into the corresponding result array?
>>
>> I don't think the issue is one of clarity, but of possibility. Unless
>> JD can save you with some magical incarnation of HISTOGRAM you are going
>> to have to change your design criteria or use a loop. If performance is
>> really that important write this function in C.
>>
>> -Rick
>
>
> Come on people. I don't use HISTOGRAM for everything. I use it very
> rarely, in fact.
>
> How about something like:
>
> nums=strsplit(strjoin(data,':'),':',/EXTRACT)
> cnts=long(total(byte(data) eq 58b,1))+1L
>
> Now you have a list of tuple-counts and the tuples themselves in a long

> list. You could (yes) use HISTOGRAM or perhaps many other methods to
> stick these into an array as you describe without looping, but rather than
> show something you'd forget 5 minutes after dropping it into your code,
> I'll join Rick in saying that if parsing these strings quickly is this
> important to you, you'll get better results by re-designing the input
> format, or pre-parsing them using a language better suited to these
> manipulations. And on the off chance that you're suffering from the
> "must-optimize-everything-in-sight" disease, you'll want to make sure a
> readable and straightforward input loop won't meet your needs before
> venturing too far into IDL esoterica:
>
> b=make_array(/LONG,VALUE=-1,max(cnt),n_elements(data))
> for i=0,n_elements(data)-1 do b[0,i]=strsplit(data[i],':',/EXTRACT)
>
> Note that there's no integer (long or otherwise) definition of NaN, so I
> used -1.
>
> JD

Thank you.

cnts=long(total(byte(data) eq 58b,1))+1L was what I was looking for.

-Darrick
