Subject: Re: Faster way to search/split a string?
Posted by David Fanning on Wed, 23 Nov 2011 14:40:19 GMT
View Forum Message <> Reply to Message

Rob writes:

> I was hoping that someone might have a cleverer way of approaching
> this problem.
>
> The following command is the bootleneck in my code:
>
> row_of_data=strsplit(all_rows[(where(stregex(all_rows, id, /boolean)
> eq 1))[0]],' ', /extract)
>
> I have a large text file with lots of columns of data (which I don't
> know exactly what the columns are until I've read them in). There are
> then say 10000 rows of this data.
>
> This is all read into one large string array (all_rows) which contains
> each row as a single very long string.
> The first 20 characters of the row contain a unique id which I need to
> search the rows for and then extract the entire matching row. This row
> then needs to be split up into it's columns (space delimited).
>
> Hopefully that all makes sense.
>
> The problem is having to do this 10000 times, (once for for each id)
> is very slow and the time to do all of the other stuff done in the
> code, reading, writing, some maths, etc is being dominated by this one
> line.
>
> Any thoughts or suggestions?

I guess I would try just reading the IDs. Then sort the IDs,
and use Value_Locate to find all the unique IDs at once. Then
you could subset your string array and do the string extraction.
Hard to tell if this would be faster.

> p.s. This needs to be GDL compatible as well which I think most
> solutions would be anyway.

This is like telling most of us "be sure it works in the
Martian atmosphere." How the hell would I know!? :-)

Cheers,

David

--
David Fanning, Ph.D.
Fanning Software Consulting, Inc.
Coyote's Guide to IDL Programming: http://www.idlcoyote.com/
Sepore ma de ni thui. ("Perhaps thou speakest truth.")

Subject: Re: Faster way to search/split a string?
Posted by rjp23 on Wed, 23 Nov 2011 15:37:56 GMT
View Forum Message <> Reply to Message

On Nov 23, 2:40 pm, David Fanning <n...@dfanning.com> wrote:
> I guess I would try just reading the IDs. Then sort the IDs,
> and use Value_Locate to find all the unique IDs at once. Then
> you could subset your string array and do the string extraction.
> Hard to tell if this would be faster.
>
>> p.s. This needs to be GDL compatible as well which I think most
>> solutions would be anyway.
>
> This is like telling most of us "be sure it works in the
> Martian atmosphere." How the hell would I know!? :-)
>
> Cheers,
>
> David
>

Thanks, I'll give it a go and see.

Subject: Re: Faster way to search/split a string?
Posted by Jeremy Bailin on Wed, 23 Nov 2011 16:02:23 GMT
View Forum Message <> Reply to Message

On 11/23/11 7:13 AM, Rob wrote:
> I was hoping that someone might have a cleverer way of approaching
> this problem.
>
> The following command is the bootleneck in my code:
>
> row_of_data=strsplit(all_rows[(where(stregex(all_rows, id, /boolean)
> eq 1))[0]],' ', /extract)
>
> I have a large text file with lots of columns of data (which I don't
> know exactly what the columns are until I've read them in). There are

> then say 10000 rows of this data.
>
> This is all read into one large string array (all_rows) which contains
> each row as a single very long string.
> The first 20 characters of the row contain a unique id which I need to
> search the rows for and then extract the entire matching row. This row
> then needs to be split up into it's columns (space delimited).
>
> Hopefully that all makes sense.
>
> The problem is having to do this 10000 times, (once for for each id)
> is very slow and the time to do all of the other stuff done in the
> code, reading, writing, some maths, etc is being dominated by this one
> line.
>
> Any thoughts or suggestions?
>
> Cheers
>
> Rob
>
> p.s. This needs to be GDL compatible as well which I think most
> solutions would be anyway.

This screams like something that should be outsourced to an external perl or python script. Dealing with strings is not IDL's forte, so why not use a better tool for that part if it's the bottleneck?

-Jeremy.

Subject: Re: Faster way to search/split a string?
Posted by Vincent Sarago on Wed, 23 Nov 2011 18:17:54 GMT
View Forum Message <> Reply to Message

I'm not sure to understand, maybe with examples it will be easier to me.

```
tmp = stregex(all_rows, '^[a-zA-Z0-9]{20}',/extract) ; Array of all IDs

test = uniq[sort(tmp)] ; Array determining witch ID is uniq

for ii = 0, n_elements(test) - 1 do begin

  id = tmp[test[ii]]

  test = where(tmp eq id, count)
  if count ne 0 then begin
```

```
    void = all_rows[test] ; all row for uniq ID

    subset = stregex(void, string(id, format='("[^",a,"].+")'), /extract)  ; only value (without ID)

    ;do what you need to do here

     tmp2 = strsplit(subset, ',/extract)



  endif

endfor
```

Subject: Re: Faster way to search/split a string?
Posted by David Fanning on Wed, 23 Nov 2011 18:21:52 GMT
View Forum Message <> Reply to Message

Vincent Sarago writes:

```
>  I'm not sure to understand, maybe with examples it will be easier to me.
>
>
>  tmp = stregex(all_rows, '^[a-zA-Z0-9]{20}',/extract) ; Array of all IDs
>
>  test = uniq[sort(tmp)] ; Array determining witch ID is uniq
>
>  for ii = 0, n_elements(test) - 1 do begin
>
>    id = tmp[test[ii]]
>
>    test = where(tmp eq id, count)
>    if count ne 0 then begin
>      tmp2 = strsplit(all_rows[test], ',/extract) ; Array of ID + Other but split
>
>     void = all_rows[test] ; all row for uniq ID
>  subset = stregex(b[ii], string(id[ii], format='("[^",a,"].+")'), /extract)
>      ;do what you need to do here
>
>
>
>    endif
>
>  endfor
```

Yeah, sorry. Big holiday coming up and company arriving shortly.
Maybe next week. :-)

Cheers,

David

--
David Fanning, Ph.D.
Fanning Software Consulting, Inc.
Coyote's Guide to IDL Programming: http://www.idlcoyote.com/
Sepore ma de ni thui. ("Perhaps thou speakest truth.")

Subject: Re: Faster way to search/split a string?
Posted by wlandsman on Thu, 24 Nov 2011 09:15:52 GMT
View Forum Message <> Reply to Message

Some thoughts (though I am not certain I understand the situation):

1. If you don't need regular expressions then I believe that using STRPOS() is quicker than using STREGEX().
2. If the ID is always in the first 20 characters of ALL_ROWS then I would created a new vector row_id = strmid(all_rows,0,20) and search on that.
3. If the ID is always exactly 20 characters, you could use a program like http://idlastro.gsfc.nasa.gov/ftp/pro/misc/match.pro to then find the matching indices in row_id and id. This is similar to David's suggestion of sorting and using VALUE_LOCATE.
4. You want to make only a single call to STRSPLIT() for all 10,000 rows. Since IDL V8.0, STRSPLIT returns a list data type when supplied with an array (since in principle each string element could have a different number of "columns"). If you have an earlier version of IDL -- or if this capability is not available in GDL, then I would use the vector capability of STRMID (http://www.idlcoyote.com/code_tips/strmidvec.html)

--Wayne

Subject: Re: Faster way to search/split a string?
Posted by rjp23 on Thu, 24 Nov 2011 14:16:03 GMT
View Forum Message <> Reply to Message

On Nov 24, 9:15 am, wlandsman <wlands...@gmail.com> wrote:
> Some thoughts (though I am not certain I understand the situation):
>
> 1. If you don't need regular expressions then I believe that using STRPOS() is quicker than using STREGEX().
> 2. If the ID is always in the first 20 characters of ALL_ROWS then I would created a

new vector row_id = strmid(all_rows,0,20) and search on that.
> 3. If the ID is always exactly 20 characters, you could use a program likehttp://idlastro.gsfc.nasa.gov/ftp/pro/misc/match.proto then find the matching indices in row_id and id. This is similar to David's suggestion of sorting and using VALUE_LOCATE.
> 4. You want to make only a single call to STRSPLIT() for all 10,000 rows. Since IDL V8.0, STRSPLIT returns a list data type when supplied with an array (since in principle each string element could have a different number of "columns"). If you have an earlier version of IDL -- or if this capability is not available in GDL, then I would use the vector capability of STRMID (http://www.idlcoyote.com/code_tips/strmidvec.html)
>
> --Wayne

Thanks for that. Doing #2 made a huge difference along with David's suggestion.

On a related note but a different problem (and I know people aren't GDL experts here so apologies) I'm finding that strsplit in GDL is much slower than in IDL. Has anyone come across this before? I've tried searching the GDL help but it's obviously not as extensive as here.

I also just wanted to check that I'm doing this the optimum way for IDL.

If I have a string array like: array=[abc_001, abc_002, abc_003, bcdef_001, bcdef_002, cdf_001_001, cdf_001_002]

I've assumed the fastest way to get the start section characters is to do:

```
for loop=0, n_elements(array)-1 do begin
start_section[loop]=(strsplit(array[loop], '_', /extract))[0]
endfor
```

or is there a vectorised way to use strsplit in IDL? (limited to IDL 7 and/or GDL)

Subject: Re: Faster way to search/split a string?
Posted by rjp23 on Thu, 24 Nov 2011 14:41:31 GMT
View Forum Message <> Reply to Message

On Nov 24, 2:16 pm, Rob <rj...@le.ac.uk> wrote:
> On Nov 24, 9:15 am, wlandsman <wlands...@gmail.com> wrote:
>
>> Some thoughts (though I am not certain I understand the situation):
>

>> 1. If you don't need regular expressions then I believe that using STRPOS() is quicker than using STREGEX().
>> 2. If the ID is always in the first 20 characters of ALL_ROWS then I would created a new vector row_id = strmid(all_rows,0,20) and search on that.
>> 3. If the ID is always exactly 20 characters, you could use a program likehttp://idlastro.gsfc.nasa.gov/ftp/pro/misc/match.prototh en find the matching indices in row_id and id. This is similar to David's suggestion of sorting and using VALUE_LOCATE.
>> 4. You want to make only a single call to STRSPLIT() for all 10,000 rows.
 Since IDL V8.0, STRSPLIT returns a list data type when supplied with an array (since in principle each string element could have a different number of "columns"). If you have an earlier version of IDL -- or if this capability is not available in GDL, then I would use the vector capability of STRMID (http://www.idlcoyote.com/code_tips/strmidvec.html)
>
>> --Wayne

Ah after re-reading #4 a few times I realised it's what I was asking

strmid(variable_names, 0, transpose(strpos(variable_names, '_')))

Thanks :-)
