Subject: regular expressions
Posted by Michael Werger on Fri, 04 Jun 1999 07:00:00 GMT
View Forum Message <> Reply to Message

Dear IDL'ers

for a complex batch processing in IDL I need to do some regular
expression handling. Of course I can do this like:

```
function regexp_match,argument,pattern
 defsysv,!true, 1 eq 1  ; defined here only for completeness
 defsysv,!false, 1 eq 0  ; see above

 command='perl -e "print ('"+argument+"' =~ m/'+pattern+'/)"'
 spawn,command,result
 if (result[0] eq 1) then result = !true else result = !false
 return,result
end
```

and then in some code:

```
if regexp_match(string,'\s*\d+') then print,'(spaces and) digits found!'
```

but this is rather slow, requires perl to be setup properly and
so on. Did anyone already wrote some routines like
regexp_replace and regexp_match (I think these names are speaking
for themselves? - like the tcl routines regsub and regexp?

Suggestions to improve the above routine are also welcome.

--
Michael Werger   ----------o
                        ESA ESTEC & Praesepe B.V. |
                Astrophysics Division   mwerger@astro.estec.esa.nl|
| Postbus 299   http://astro.estec.esa.nl |
| 2200 AG Noordwijk   +31 71 565 3783 (Voice)
o------------------- The Netherlands   +31 71 565 4690 (FAX)

Subject: Re: Regular expressions
Posted by Brian Jackel on Fri, 10 Nov 2000 08:00:00 GMT
View Forum Message <> Reply to Message

This has been around for a couple versions (since 5.3?)

"The STREGEX function performs regular expression matching against the
strings

contained in StringExpression. STREGEX can perform either a simple
boolean
True/False evaluation of whether a match occurred, or it can return the
position
and offset within the strings for each match. The regular expressions
accepted
by this routine, which correspond to "Posix Extended Regular
Expressions", are
similar to those used by such UNIX tools as egrep, lex, awk, and Perl."

Brian

James Tappin wrote:
>
> Does there exist a routine for regular expression matching in IDL?
>
> I'm thinking of something along the lines of an extended STR_SEP that could
> (say) separate a string into components separated by 0 or 1 commas and an
> arbitrary number of spaces.
>
> I could do it by spawning a perl script but that's not too pretty and I
> can't see an _easy_ way to do it natively.
>
> James
>
> --
> +----------------------+-----------------------------+-- -------+
> | James Tappin | School of Physics & Astronomy | O__ |
> | sjt@star.sr.bham.ac.uk | University of Birmingham | -- \/` |
> | Ph: 0121-414-6462. Fax: 0121-414-3722 | | |
> +---+-- -------+

Subject: Re: Regular expressions
Posted by Martin Schultz on Fri, 10 Nov 2000 08:00:00 GMT
View Forum Message <> Reply to Message

"Liam E. Gumley" wrote:
>
> James Tappin wrote:
>>
>> Does there exist a routine for regular expression matching in IDL?
>>
>> I'm thinking of something along the lines of an extended STR_SEP that could
>> (say) separate a string into components separated by 0 or 1 commas and an
>> arbitrary number of spaces.
>>

>> I could do it by spawning a perl script but that's not too pretty and I
>> can't see an _easy_ way to do it natively.
>
> The STREGEX function was introduced in IDL 5.3:
>
> [...][/color]

One of the VERY good reasons why version 4.xx i snot enough for me at least ;-)

CHeers,
Martin

--
[[[[[[[[[
[[Dr. Martin Schultz   Max-Planck-Institut fuer Meteorologie   [[
[[Bundesstr. 55, 20146 Hamburg   [[
[[phone: +49 40 41173-308   [[
[[fax: +49 40 41173-298   [[
[[martin.schultz@dkrz.de   [[
[[[[[[[[[

Subject: Re: Regular expressions
Posted by Liam E. Gumley on Fri, 10 Nov 2000 08:00:00 GMT
View Forum Message <> Reply to Message

James Tappin wrote:
>
> Does there exist a routine for regular expression matching in IDL?
>
> I'm thinking of something along the lines of an extended STR_SEP that could
> (say) separate a string into components separated by 0 or 1 commas and an
> arbitrary number of spaces.
>
> I could do it by spawning a perl script but that's not too pretty and I
> can't see an _easy_ way to do it natively.

The STREGEX function was introduced in IDL 5.3:

"The STREGEX function performs regular expression matching against the strings contained in StringExpression. STREGEX can perform either a simple boolean True/False evaluation of whether a match occurred, or it can return the position and offset within the strings for each match.
The regular expressions accepted by this routine, which correspond to "Posix Extended Regular Expressions", are similar to those used by such UNIX tools as egrep, lex, awk, and Perl."

Cheers,
Liam.
http://cimss.ssec.wisc.edu/~gumley

Subject: Re: Regular Expressions
Posted by John-David T. Smith on Fri, 16 Mar 2001 00:09:41 GMT
View Forum Message <> Reply to Message

Wayne Landsman wrote:
>
> The following is probably a simple question for anyone familiar with
> regular expressions, but I am still trying to learn the STREGEX
> function.
>
> Suppose I want to find the first occurence in a string of an 'l' ithat
> is not part of a double 'l'. For
> example, in the string
>
> IDL> st = 'The rolling hills and lake'
>
> I want to return the character position of the 'l' in lake (=21).
>
> The following expression almost works -- it will search for any 'l'
> which is both preceded and followed by anything that is not "l"
>
> IDL> print,stregex(st, '[^l]l[^l]')
>
> but it won't work for the string 'The rolling hills and pool' because
> the final 'l' has no characters following it. Any suggestions?

IDL> print, stregex(st,'(^|[^l])l($|[^l])')

which means "a character that is not 'l', or the beginning of the
string, followed by an 'l', followed by a character that is not 'l', or
the end of the string". Aren't you glad Ken Thompson didn't decide
originally to develop regexps in english?

This will also work on

IDL> st = "let's all go the the movies"

JD

Subject: Re: Regular Expressions
Posted by Wayne Landsman on Fri, 16 Mar 2001 03:24:50 GMT

JD Smith wrote:

> IDL> print, stregex(st,'(^|[^l])l($|[^l])')
>
> which means "a character that is not 'l', or the beginning of the
> string, followed by an 'l', followed by a character that is not 'l', or
> the end of the string". Aren't you glad Ken Thompson didn't decide
> originally to develop regexps in english?
>
> This will also work on
>
> IDL> st = "let's all go the the movies"

Thanks. But I now realize that my original formulation was not quite correct, since the above expression (usually!) returns the position of the character *before* the 'l', so to get the position of the first single 'l' one has to add 1

IDL> l_position = stregex(st,'(^|[^l])l($|[^l])') + 1

Unfortunately, if 'l' is the first character, then you *don't* want to add the 1. (The expression stregex(st,'(^|[^l])l($|[^l])') returns a value of 0 for both st ='long days' and st ='slow nights'.)
One solution is to forget about the beginning of string anchor and just concatenate a blank to the beginning to the string

IDL> l_position = stregex(' ' + st,'[^l]l($|[^l])')

--Wayne

P.S. The real-life problem I am working on deals not with 'l' but with apostrophes. I am trying to speed up the processing of FITS header values, where is a string is delineated by non-repeating apostrophes, and a possessive is indicated by a double apostrophe.

VALUE = 'This is Wayne''s FITS value' / Example string field

Subject: Re: Regular Expressions
Posted by John-David T. Smith on Fri, 16 Mar 2001 17:20:50 GMT

Wayne Landsman wrote:

> P.S. The real-life problem I am working on deals not with 'l' but with
> apostrophes. I am trying to speed up the processing of FITS header

> values, where is a string is delineated by non-repeating apostrophes, and a
> possessive is indicated by a double apostrophe.
>
> VALUE = 'This is Wayne''s FITS value' / Example string field

how about:

IDL> value= "VALUE = 'This is Wayne''s FITS value' / A FITS COMMENT"
IDL> print,(stregex(value,/SUBEXPR,/EXTRACT,"= *'(.*)'([^']|$)"))[1]

You will always have something before the initial "'" in the full header record.

You can then change double quotes to single quotes in the usual way with a strpos loop.

JD

Subject: Re: Regular Expressions
Posted by Pavel A. Romashkin on Mon, 19 Mar 2001 21:52:57 GMT
View Forum Message <> Reply to Message

Wouldn't it be easier to analyse a byte array with more human-readible functions, than those beautiful regular expressions you guys brought up?

Cheers,
Pavel

Subject: Re: Regular Expressions
Posted by Martin Schultz on Tue, 20 Mar 2001 12:39:34 GMT
View Forum Message <> Reply to Message

"Pavel A. Romashkin" wrote:
>
> Wouldn't it be easier to analyse a byte array with more human-readible
> functions, than those beautiful regular expressions you guys brought up?
>
> Cheers,
> Pavel

Oh no! Pavel! That would mean to take all the fun out of it! Just imagine IDL got rid of all the quirks we spend so much time musing upon in this group. Wouldn't that be boring (and David would be out of bread and butter, too). With regular expressions, it's a similar thing: they are brain sport! Somewhere I read that people who train

their brain regularily have a better chance of avoiding dementia later
on. So, where's the weekly regular expression contest?

Cheers,

Martin

PS: As far as I know, emacs is nothing else than a smart and well
balanced collection of regular expressions ;-)

--
[[[[[[[[[[
[[Dr. Martin Schultz   Max-Planck-Institut fuer Meteorologie   [[
[[Bundesstr. 55, 20146 Hamburg   [[
[[phone: +49 40 41173-308   [[
[[fax: +49 40 41173-298   [[
[[martin.schultz@dkrz.de   [[
[[[[[[[[[[

Subject: Re: Regular Expressions
Posted by Wayne Landsman on Tue, 20 Mar 2001 20:14:54 GMT
View Forum Message <> Reply to Message

"Pavel A. Romashkin" wrote:

> Wouldn't it be easier to analyse a byte array with more human-readible
> functions, than those beautiful regular expressions you guys brought up?
>

It depends on what you mean by "easier". One nice thing about STREGEX is
that it works on vector strings. One can always convert the string
array to a byte array and analyze, but -- **if you are trying to avoid
loops** -- the indexing can be become extremely opaque, and exercise at
least as many brain cells as using STREGEX. For example, JD's solution
can also apply to a string array where one is trying to extract the
substrings beginning and ending with a singe quote:

```
IDL> st = ["value1 = 'Wayne''s dog'  / First string ", $
         "value2 = 'Sue''s dog and Ralph''s cat' / Second string ", $
         "value3 =  'two pigeons'" ]

IDL> val = (stregex(st, /SUBEXPR,/EXTRACT,"= *'(.*)'([^']|$)"))[1,*]
IDL> print,val
Wayne''s dog
Sue''s dog and Ralph''s cat
two pigeons
```

Of course, one should probably add an English comment to the use of STRGEX

; Find the substring beginning with an "=", followed by any number of
characters,
; followed by a quote, followed by any number of characters (including
double
; quotes) up to the last single quote.   Extract from this substring all
; characters between the first and last single quotes.

Subject: Re: Regular Expressions
Posted by Mark Hadfield on Tue, 20 Mar 2001 22:48:36 GMT
View Forum Message <> Reply to Message

"Wayne Landsman" <landsman@mpb.gsfc.nasa.gov> wrote in message
news:3AB7BA3E.CA411E1B@mpb.gsfc.nasa.gov...
> Of course, one should probably add an English comment to the use of STRGEX
>
> ; Find the substring beginning with an "=", followed by any number
> ; of characters, followed by a quote, followed by any number of
> ; characters (including double quotes) up to the last single quote.
> ; Extract from this substring all characters between the first
> ; and last single quotes.

So, you're saying that STREGEX is a good thing because (like HISTOGRAM) it
allows you to write code in which the executable statements are several
times shorter than the comments required to explain them?

Mark Hadfield
m.hadfield@niwa.cri.nz http://katipo.niwa.cri.nz/~hadfield
National Institute for Water and Atmospheric Research

Subject: Re: Regular Expressions
Posted by Craig Markwardt on Tue, 20 Mar 2001 23:23:52 GMT
View Forum Message <> Reply to Message

"Mark Hadfield" <m.hadfield@niwa.cri.nz> writes:

> "Wayne Landsman" <landsman@mpb.gsfc.nasa.gov> wrote in message
> news:3AB7BA3E.CA411E1B@mpb.gsfc.nasa.gov...
>> Of course, one should probably add an English comment to the use of STRGEX
>>
>> ; Find the substring beginning with an "=", followed by any number
>> ; of characters, followed by a quote, followed by any number of

>> ; characters (including double quotes) up to the last single quote.
>> ; Extract from this substring all characters between the first
>> ; and last single quotes.
>
> So, you're saying that STREGEX is a good thing because (like HISTOGRAM) it
> allows you to write code in which the executable statements are several
> times shorter than the comments required to explain them?

Wouldn't that be APL?

Craig

--
 -- --------------
Craig B. Markwardt, Ph.D. EMAIL: craigmnet@cow.physics.wisc.edu
Astrophysics, IDL, Finance, Derivatives | Remove "net" for better response
 -- --------------

Subject: Re: Regular Expressions
Posted by John-David T. Smith on Tue, 20 Mar 2001 23:28:40 GMT
View Forum Message <> Reply to Message

Mark Hadfield wrote:
>>
>> ; Find the substring beginning with an "=", followed by any number
>> ; of characters, followed by a quote, followed by any number of
>> ; characters (including double quotes) up to the last single quote.
>> ; Extract from this substring all characters between the first
>> ; and last single quotes.
>
> So, you're saying that STREGEX is a good thing because (like HISTOGRAM) it
> allows you to write code in which the executable statements are several
> times shorter than the comments required to explain them?

I take that as a personal jab. Actually, the code I write is much more
comprehensible than the examples I post here -- I *do* have a reputation
to maintain though.

Somehow, I think the equivalent byte array version would be even
uglier. Anyone care to whip up a version for comparison, using the
detailed comments above?

JD
