Subject: Locating sequence of bytes within binary file
Posted by medd on Tue, 15 Jun 2010 11:30:43 GMT
View Forum Message <> Reply to Message

Hi,

I need to locate a given sequence of bytes within a binary file. I do not manage to do it efficiently, and I wanted to ask if somebody here has a clue.

I saw that there are no functions in IDL to look for a given sequence within a byte array, but there are very powerful functions to look for a sequence within a string using regular expressions. This is what I tried:

```
fcontent = BYTARR((FILE_INFO(fn)).size, /NOZERO) ;Variable where to
read in the file
OPENU, unit, fn, /GET_LUN;, /SWAP_ENDIAN
READU, unit, fcontent
IF(STREGEX(STRING(fcontent), STRING(sequence_searched)) LT 0) THEN
print, 'sequence not found'
```

This works!! ... But only as long as the file does not contain a byte with the value 0 (which, too bad!, it does...)

After looking a while, I found in this forum (message "Null terminated strings") and in the IDL help that a string is truncated as soon as this value is found. This explains why this method fails. But it does not propose solutions... :(

Do you know some smart workaround? Or do you know other efficient ways in IDL to locate a sequence of bytes within a binary file?

Thanks!

PS. I thought about replacing all 0's by 1's, but it is a really dirty solution, which might find the sequence at the wrong place in case there is a similar sequence which really contains a 1 instead...

Subject: Re: Locating sequence of bytes within binary file
Posted by penteado on Wed, 16 Jun 2010 17:46:46 GMT
View Forum Message <> Reply to Message

It is not the prettiest solution, but you could turn your byte array into an array of strings of the byte values. That is, a print-like conversion, instead of byte to string conversion. Something like

scontent=strjoin(string(int(fcontent),format='(I3.3)')))

Then search for a similarly stringified version of your byte pattern

spattern=strjoin(string(int(sequence_searched),format='(I3.3)')))

if stregex(scontent,spattern,/boolean) then ...

I did not test it, but I think it should work, at the cost of using 3 bytes for each byte in the original content (3 ASCII digits for each value). This also avoids the other problem with your initial solution, which would happen if the string from your byte pattern happened to have any special characters, which would be interpreted non-literally in a regex.

Subject: Re: Locating sequence of bytes within binary file
Posted by medd on Thu, 17 Jun 2010 08:03:44 GMT
View Forum Message <> Reply to Message

On Jun 16, 7:46 pm, pp <pp.pente...@gmail.com> wrote:
> It is not the prettiest solution, but you could turn your byte array
> into an array of strings of the byte values. That is, a print-like
> conversion, instead of byte to string conversion. Something like
>
> scontent=strjoin(string(int(fcontent),format='(I3.3)')))
>
> Then search for a similarly stringified version of your byte pattern
>
> spattern=strjoin(string(int(sequence_searched),format='(I3.3)')))
>
> if stregex(scontent,spattern,/boolean) then ...
>
> I did not test it, but I think it should work, at the cost of using 3
> bytes for each byte in the original content (3 ASCII digits for each
> value). This also avoids the other problem with your initial solution,
> which would happen if the string from your byte pattern happened to
> have any special characters, which would be interpreted non-literally
> in a regex.

Thanks for all replies.

I find pp's solution to be really funny, you got it :) And still it works, without using external libraries and being reliable. I go for that!

But I have to admit that I am a bit sad to learn that such a problem is so difficult to tackle (in this case regarding memory usage).

I am not an expert, but when I explain IDL to newbies I always say
that it is a "matrix-oriented language", with all possible operations
you can imagine on arrays. But looking for more than one consecutive
value within an array seems to be too hard...

Could it be a suggestion for an addition in later versions? Hopefully
IDL8 is out really soon ! (even without this feature)

Subject: Re: Locating sequence of bytes within binary file
Posted by JDS on Thu, 17 Jun 2010 17:48:57 GMT
View Forum Message <> Reply to Message

On Jun 17, 4:03 am, medd <med...@googlemail.com> wrote:

> I am not an expert, but when I explain IDL to newbies I always say
> that it is a "matrix-oriented language", with all possible operations
> you can imagine on arrays. But looking for more than one consecutive
> value within an array seems to be too hard...

As usual, we can (over-)use IDL's array strengths to brute-force this
using REBIN and dimensional TOTAL:

```
;===============
nh=1000000L
nn=20b
haystack=byte(randomu(sd,nh)*256)
start=long(randomu(sd)*(nh-nn))
needle=haystack[start:start+nn-1]

targ=[nn,ceil(float(nh)/nn)]
search=make_array(/BYTE,targ,/NOZERO)
quill=rebin(needle,targ,/SAMPLE)

for off=0,nn-1 do begin
  search[0]=shift(haystack,-off)
  matches=where(total(search eq quill,1,/PRESERVE_TYPE) eq nn,cnt)
  if cnt gt 0 then break
endfor

match=cnt gt 0?matches[0]*nn+off:-1
;===============
```

If you are interested in more than just the first match, simply omit
the break statement, and accumulate a list of match locations (or
increment a match count). It's limited to 256 byte needles, but that

could be fixed by substituting PRODUCT for TOTAL (at a very slight speed penalty). It's reasonably fast, though of course cannot touch the speed of a true Boyer-Moore string search. I tried it on the same data set using INDEX in perl and found it roughly 40x faster.

Now for the really disappointing news: as is often found, brute-forcing, while emphasizing IDL's strengths, often comes with a penalty compared to more efficient algorithms. I find that STRPOS in IDL is at least 100x faster, likely because it uses an efficient string search algorithm internally. But, as you notice, IDL won't allow null characters (0b) in a string (probably as a questionable concession to 0-delimited C strings).

That motivates another deeply unsatisfying, but resoundingly faster (20-50x) option: simply replace all 0b's with 1b's in both input byte array and search array, and just double-check for spurious matches as you go:

```
;===============
function binpos,haystack,needle
  nn=n_elements(needle)

  sn=string(needle>1b)
  sh=string(haystack>1b)

  pos=-1
  repeat begin
     pos=strpos(sh,sn,pos+1)
     if pos ne -1 && array_equal(haystack[pos:pos+nn-1],needle) then
break
  endrep until pos eq -1
  return,pos
end
;===============
```

In random arrays I find false positives are quite rare for search array lengths greater than a few. Of course, your data probably isn't random.

We might also lobby ITT to let STRPOS and its sort accept byte arrays (since frankly there is very little difference between them internally).

JD

Subject: Re: Locating sequence of bytes within binary file

Posted by on Fri, 18 Jun 2010 16:47:59 GMT
View Forum Message <> Reply to Message

Wow, I had read in Coyote's website that JD Smith always finds amazing solutions ;), but this is over any expectation!! I am simply amazed. Not only I see all possible solutions for my problem and their drawbacks, but I also learn a lot!

I spent some time to understand the methods, managed now...

> As usual, we can (over-)use IDL's array strengths to brute-force this
> using REBIN and dimensional TOTAL:

The problem of this solution might be the size of the variable quill for a large "needle" (not my case). And also the speed, as you acknowledge...

> That motivates another deeply unsatisfying, but resoundingly faster
> (20-50x) option: simply replace all 0b's with 1b's in both input byte
> array and search array, and just double-check for spurious matches as
> you go:

This is good, but it just uses double the amount of memory to preserve the original and the modified array. Works for me, but could be bad for others. And, honestly, it is really not so elegant.

> We might also lobby ITT to let STRPOS and its sort accept byte arrays
> (since frankly there is very little difference between them
> internally).

This would be by far the best solution. I believe a VIP like you have more options to lobby ITT than I do :) Maybe creating a "BYTPOS" or actually "ANYDATAFORMAT_POS" function would be good. A similar problem might arise looking for a sequence of integers instead of bytes...

My problem is small, and solved by any of the solutions which were proposed. But it would be good to have a fast and memory-efficient option available for future IDL-users :) Next time then I do not need to be ashamed of saying that IDL does virtually everything with arrays.

Thanks so much!!!

Subject: Re: Locating sequence of bytes within binary file
Posted by JDS on Fri, 18 Jun 2010 22:12:52 GMT
View Forum Message <> Reply to Message

On Jun 18, 12:47 pm, Axel Martínez <axe...@gmail.com> wrote:
>> As usual, we can (over-)use IDL's array strengths to brute-force this
>> using REBIN and dimensional TOTAL:
>
> The problem of this solution might be the size of the variable quill
> for a large "needle" (not my case).

Memory usage is actually fairly stable with needle size, and
comparable to the other solution. The larger the needle, the fewer
the rows. Put another way, "quill" is always of order the size of the
search array, and no larger.

> And also the speed, as you acknowledge...

This is where a large needle hurts you; the need to wrap the search
array around in n_needle steps.

>> That motivates another deeply unsatisfying, but resoundingly faster
>> (20-50x) option: simply replace all 0b's with 1b's in both input byte
>> array and search array, and just double-check for spurious matches as
>> you go:
>
> This is good, but it just uses double the amount of memory to preserve
> the original and the modified array. Works for me, but could be bad
> for others. And, honestly, it is really not so elegant.

Agreed. That said, if elegance is on your short list of programming
language features, I fear you shouldn't go anywhere near IDL. I think
the other message here is that finding where a given value occurs in
an array is, although seemingly just a short extrapolation away,
actually a completely different problem from finding where a sequence
of values appears. The latter has considerable room for optimization
over the former. For example, you might notice that STRPOS runs in a
time that is almost independent of needle length. Some algorithms run
faster the longer the needle: http://en.wikipedia.org/wiki/BoyerMoore_string_search_algori thm.
Brains always triumph over brute-force.

>> We might also lobby ITT to let STRPOS and its sort accept byte arrays
>> (since frankly there is very little difference between them
>> internally).
>
> This would be by far the best solution. I believe a VIP like you have
> more options to lobby ITT than I do :)

I don't know about that. Most new features start with a simple
request to ITT support; give it a try.

JD

Subject: Re: Locating sequence of bytes within binary file
Posted by David Fanning on Fri, 18 Jun 2010 22:21:57 GMT
View Forum Message <> Reply to Message

JD Smith writes:

>> This would be by far the best solution. I believe a VIP like you have
>> more options to lobby ITT than I do :)
>
> I don't know about that. Most new features start with a simple
> request to ITT support; give it a try.

Yeah, I would have thought an inverse correlation was
more likely the case. At least to judge from my own
experience. ;-)

Cheers,

David

P.S. I do see a few of my ideas (expounded upon at least
since IDL 5.0) showing up, at last, in IDL 8.0. Perhaps
I have more influence than I think. :-)

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

Subject: Re: Locating sequence of bytes within binary file
Posted by kamal on Thu, 24 Jun 2010 08:42:55 GMT
View Forum Message <> Reply to Message

Hi all,

I have a .shp file (vector file). I would like to convert it into ESRI
GRID format. How could it be achieved? Please let me know.

Thanks
Kamal
