Subject: RegExp this'n that
Posted by luoganda on Sun, 20 Nov 2016 15:30:28 GMT
View Forum Message <> Reply to Message

-updated to pcre v8.39, dated 2016...
-more reg submatches, before it was only around cca 10

Issues in v8.10 is that is has problems matching some stuff, issues with many |, or more than 10 submatches.

Original pcre source is taken from pcre website.

This was tested with upp9251 - since this version works on windows xp - latest does not.
I think it should be 'revisited' by someone and integrated in upp.
There is maybe one 'stuff' to revisit/check/fix, because it's defined in two places:
 -max_pcre_offsets - look source and note-simmx.txt

~~~~~~~~~~~

This did not match in original upp pcre-8.10, in 8.39 it does(modified to match more than 10) - as it should:

```
RegExp re(
 "(stuff)\\s+(\\d+)?\\s*(stuffx)?\\s*([a-zA-Z_][a-zA-Z0-9_-]*=(?:\".*\"|\'.*\'|[[:graph:]]*) )*\\s*(\".*?\");"
 "|(stuff2)\\s+(\".*?\");"
 "|(stuff3)\\s+(\".*?\".*?);"
 "|(.*?);"
 );
if(re.Match("tid nanu;"))PromptOK("matches");
```

File Attachments

1) ultimate++-pcre-8.39-properlyWorkingOrMoreSubMatches.7z, downloaded 431 times

Subject: Re: RegExp this'n that
Posted by luoganda on Wed, 23 Nov 2016 09:37:52 GMT
View Forum Message <> Reply to Message

Multiline mode - that is RegExp::MULTILINE - seems to
properly work now in multiple situations when using ^ and $ operands for lines of text.
Without RegExp::MULTILINE, it matches start of a string as it should.

Before you had to use something like ` to correctly match start of a string

Subject: Re: RegExp this'n that
Posted by mirek on Sun, 27 Nov 2016 19:09:55 GMT
~~~~~~~~~~~

Thank you, good work. Merged with trunk. (Hope it is ok...)

Mirek

Subject: Re: RegExp this'n that
Posted by luoganda on Sun, 04 Dec 2016 13:45:59 GMT

Yes.

Newer version doesn't have problems around matching more that cca 10 captures, even if def max_pcre_offsets=30(default),
that's because some bugs were fixed - using default value 30 is ok for general usage(cca18stack_based), and more than
that, lib will use malloc(and copy some values there).

So for upp pcre optimal usage:
-config.h <=remove any max_pcre_offsets definitions(using 30 as defPcreDoesIsEnoughForMost,
that is (30*2)/3-2=18maxStackBasedCaptures
-pcre_exec.c <=modify lines near REC_STACK_SAVE_MAX into:

```
  #ifdef pcre_max_stack_offsets
  #define REC_STACK_SAVE_MAX pcre_max_stack_offsets
  #else
  #define REC_STACK_SAVE_MAX 30
  #endif
```

-RegExp.h <=modify lines near

```
  #ifdef pcre_max_stack_offsets
  int pos[pcre_max_stack_offsets]; //must be multiple of 3
  #else
  int pos[30];    //original 30(okForMostOfGeneralStuff)=(30*2)/3=max 20-2(forErr)=18
                          capturedBackRefs stack based, else malloc is used(and copied!)
  #endif
```

Now,if you want to fine tune RegExp stack based usage, define pcre_max_stack_offsets in TheIDE, or command line - multipleOf 3.

This matches in updated pcre version:

```
RegExp re(
 "(00name)|(02name)|(03name)|(04name)|(05name)|(06name)|(07name)|(08name)|(09name)|(10name)|"
 "(01name)|(12name)|(13name)|(14name)|(15name)|(16name)|(17name)|(18name)|(19name)|(20name)|"
 "(21name)|(22name)|(23name)|(24name)|(25name)|(26name)|(27name)|(28name)|(29name)|(30
```

```
name)|"
 "(31name)|(32name)|(33name)|(34name)|(35name)|(36name)|(37name)|(38name)|(39name)|(40
name)|"
 "(41name)|(42name)|(43name)|(44name)|(45name)|(46name)|(47name)|(48name)|(49name)|(50
name)|"
 "(51name)|(52name)|(53name)|(54name)|(55name)|(56name)|(57name)|(58name)|(59name)|(60
name)|"
 "(61name)|(62name)|(63name)|(64name)|(65name)|(66name)|(67name)|(68name)|(69name)|(70
name)|"
 "(71name)|(72name)|(73name)|(74name)|(75name)|(76name)|(77name)|(78name)|(79name)|(80
name)|"
 "(81name)|(82name)|(83name)|(84name)|(85name)|(86name)|(87name)|(88name)|(89name)|(90
name)|"
 "(91name)|(92name)|(93name)|(94name)|(95name)|(96name)|(97name)|(98name)|(99name)|(10
0name)"  //100

 "(100name)|(102name)|(103name)|(104name)|(105name)|(106name)|(107name)|(108name)|(109
name)|(110name)|"
 "(101name)|(112name)|(113name)|(114name)|(115name)|(116name)|(117name)|(118name)|(119
name)|(120name)|"
 "(121name)|(122name)|(123name)|(124name)|(125name)|(126name)|(127name)|(128name)|(129
name)|(130name)|"
 "(131name)|(132name)|(133name)|(134name)|(135name)|(136name)|(137name)|(138name)|(139
name)|(140name)|"
 "(141name)|(142name)|(143name)|(144name)|(145name)|(146name)|(147name)|(148name)|(149
name)|(150name)|"
 "(151name)|(152name)|(153name)|(154name)|(155name)|(156name)|(157name)|(158name)|(159
name)|(160name)|"
 "(161name)|(162name)|(163name)|(164name)|(165name)|(166name)|(167name)|(168name)|(169
name)|(170name)|"
 "(171name)|(172name)|(173name)|(174name)|(175name)|(176name)|(177name)|(178name)|(179
name)|(180name)|"
 "(181name)|(182name)|(183name)|(184name)|(185name)|(186name)|(187name)|(188name)|(189
name)|(190name)|"
 "(191name)|(192name)|(193name)|(194name)|(195name)|(196name)|(197name)|(198name)|(199
name)|(200name)" //200

 "(200name)|(202name)|(203name)|(204name)|(205name)|(206name)|(207name)|(208name)|(209
name)|(210name)|"
 "(201name)|(212name)|(213name)|(214name)|(215name)|(216name)|(217name)|(218name)|(219
name)|(220name)|"
 "(221name)|(222name)|(223name)|(224name)|(225name)|(226name)|(227name)|(228name)|(229
name)|(230name)|"
 "(231name)|(232name)|(233name)|(234name)|(235name)|(236name)|(237name)|(238name)|(239
name)|(240name)|"
 "(241name)|(242name)|(243name)|(244name)|(245name)|(246name)|(247name)|(248name)|(249
name)|(250name)|"
 "(251name)|(252name)|(253name)|(254name)|(255name)|(256name)|(257name)|(258name)|(259
```

```
name)|(260name)|"
 "(261name)|(262name)|(263name)|(264name)|(265name)|(266name)|(267name)|(268name)|(269
name)|(270name)|"
 "(271name)|(272name)|(273name)|(274name)|(275name)|(276name)|(277name)|(278name)|(279
name)|(280name)|"
 "(281name)|(282name)|(283name)|(284name)|(285name)|(286name)|(287name)|(288name)|(289
name)|(290name)|"
 "(291name)|(292name)|(293name)|(294name)|(295name)|(296name)|(297name)|(298name)|(299
name)|(300name)" //300
);
if(re.Match("300name"))PromptOK("Matches");
```

Subject: Re: RegExp this'n that
Posted by luoganda on Sun, 25 Dec 2016 17:02:52 GMT
View Forum Message <> Reply to Message

Although previous post describes optimal solution,
note that 'pcre_max_stack_offsets'(ifUsed) must be defined in two places to work,
it won't work if you just define it in pcre package.

Default 30 value still doesn't work correctly,
setting this to 33 does - i am not sure why, maybe it has something to do with two 1st values used in lib.

So updated optimal solution for now is:
-setting default pos[33] in RegExp.h and REC_STACK_SAVE_MAX=33
-allow user to modify this with pcre_max_stack_offsets: should be >=33 and mutiple of 3

Subject: Re: RegExp this'n that
Posted by mirek on Wed, 28 Dec 2016 16:05:49 GMT
View Forum Message <> Reply to Message

luoganda wrote on Sun, 25 December 2016 18:02Although previous post describes optimal solution,
note that 'pcre_max_stack_offsets'(ifUsed) must be defined in two places to work,
it won't work if you just define it in pcre package.

Default 30 value still doesn't work correctly,
setting this to 33 does - i am not sure why, maybe it has something to do with two 1st values used in lib.

So updated optimal solution for now is:
-setting default pos[33] in RegExp.h and REC_STACK_SAVE_MAX=33
-allow user to modify this with pcre_max_stack_offsets: should be >=33 and mutiple of 3

Uhm, anything that I should apply to plugin/pcre?

Mirek

Subject: Re: RegExp this'n that
Posted by luoganda on Fri, 06 Jan 2017 21:25:03 GMT
View Forum Message <> Reply to Message

Maybe only what has been proposed so far.

Setting stack values to 120(as had been proposed in 1st few msgs) in RegExp.h and for REC_STACK_SAVE_MAX works ok, but it's a little bit too much for generic usage.
Default value for this is 30 - but it doesn't work properly.

So, using 33 for this seems ok - but it's more or less in 'experimental' stage, so 2things:
-maybe more tests with 33 value
-maybe find a way to specify/declare 'pcre_max_stack_offsets' only once - so it can be tweaked

---

Subject: Re: RegExp this'n that: unneded creation of lib
Posted by luoganda on Thu, 27 Apr 2017 09:32:43 GMT
View Forum Message <> Reply to Message

when pcre package is used with non gcc compilers,
library is unnecesarily produced - it's not needed for upp:)

pcre lib internally defines PCRE_STATIC for gcc(which in upp prevents lib creation),
but for upp it can be defined for all compilers.
So, adding new compiler option to pcre pack with -DPCRE_STATIC
wont create unnecesary lib/exp/work(including msvc).

For pcre 'stack_based' case; for many tests it seems to work ok with ... pos[33] - in RegExp.h, stuff in lib/config.h can be removed, REC_STACK_SAVE_MAX(in pcre_exec.c) can be set to 33

Subject: Re: RegExp this'n that: patch for 9251(cbInter),11040
Posted by luoganda on Thu, 04 May 2017 07:42:12 GMT
View Forum Message <> Reply to Message

Pcre 9251 is in next/prev post.

Pcre patch for 11030(andSomePreviousVers) and up - Event interface,
read note in zip for more...

nonbloated, working version,
update: rewrite plugin/pcre dir with this one, note can be found in 9251 next/prev post

File Attachments

1) pcre-patch-11040.7z, downloaded 472 times

Subject: Re: RegExp this'n that: patch for 9251(cbInter),11040
Posted by luoganda on Thu, 04 May 2017 07:47:56 GMT
View Forum Message <> Reply to Message

Pcre 11040(andSomePrevVers) is in prev/next post.

Pcre patch for 9251 - Callback interface,

full version,
update: delete contents of plugin/pcre and copy this one to it

read note in zip for more...

File Attachments

1) pcre-patch-9251-withCbInterface-full.7z, downloaded 474 times

---

Subject: Re: RegExp this'n that
Posted by luoganda on Sun, 15 Jul 2018 21:09:17 GMT
View Forum Message <> Reply to Message

This does not match, but it's taken directly from pcre 8.xx manual.
It matches correctly on many pcreCompatibleOnlinePages,eg this one regexr, if testing - don't forget to check pcre there in right-upper corner and to use single '\' if copying down pattern.
Also, subfunc of Match func in this case produces an error(pcre_exec returns -5 which is PCRE_ERROR_UNKNOWN_OPCODE), but it's not cought by upp, that is - error funcs doesn't know about it, a silent error.
This should match a balanced '(...abc(...)abc...)' pattern.

```
String s="(abc)";
RegExp re("\\(([^()]++|(?R))*\\)");
if(re.Match(s))PromptOK("\1Matches");
if(re.IsError())PromptOK(String("\1RegExpErr: ")<<re.GetError());
```

Anyone has some idea why this is so?
