

Re: Regular Expression – Matching Multiples of 3 Characters exactly.

Re: Regular Expression – Matching Multiples of 3 Characters exactly.

Source: <http://coding.derkeiler.com/Archive/Python/comp.lang.python/2008-04/msg03897.html>

- *From:* blaine <frikker@xxxxxxxx>
 - *Date:* Mon, 28 Apr 2008 10:23:02 -0700 (PDT)
-

On Apr 28, 6:30 am, Nick Craig-Wood <n...@xxxxxxxxxxxxxxxx> wrote:

blaine <frik...@xxxxxxxx> wrote:

I'm trying to write a string matching algorithm for genomic sequences. I'm pulling out Genes from a large genomic pattern, with certain start and stop codons on either side. This is simple enough... for example:

```
start = AUG stop=AGG
BBBBBBAUGWWWWWAGGBBBBBB
```

So I obviously want to pull out AUGWWWWWAGG (and all other matches).

This works great with my current regular expression.

The problem, however, is that codons come in sets of 3 bases. So there are actually three different 'frames' I could be using. For example:

```
ABCDEFGHIJ
```

I could have ABC DEF GHI or BCD EFG HIJ or CDE FGH IJx... etc.

So finally, my question. How can I represent this in a regular expression? :) This is what I'd like to do:

(Find all groups of any three characters) (Find a start codon) (find any other codons) (Find an end codon)

Re: Regular Expression – Matching Multiples of 3 Characters exactly.

Is this possible? It seems that I'd want to do something like this: `(\w \w \w)+(AUG)(\s)(AGG)(\s)*` – where `\w \w \w` matches EXACTLY all sets of three non-whitespace characters, followed by `AUG \s AGG`, and then anything else.

I'm not sure what the `\s` are doing in there – there doesn't appear to be any whitespace in your examples.

I hope I am making sense. Obviously, however, this will make sure that ANY set of three characters exist before a start codon. Is there a way to match exactly, to say something like 'Find all sets of three, then AUG and AGG, etc.'.

I think you want

```
^(\\w\\w\\w)*(AUG)((\\w\\w\\w)*?)(AGG)
```

which will match up 0 or more triples, match AUG match 0 or more triples then AGG. The `?` makes it a minimum match otherwise you'll match more than you expect if there are two AUG...AGG sequences in a given genome.

```
>>> import re
>>> m=re.compile(r"^(\\w\\w\\w)*(AUG)((\\w\\w\\w)*?)(AGG)")
>>> m.search("BBBBBBAUGWWWWWAGGBBBBB").groups()
('BBB', 'AUG', 'WWWWW', 'WWW', 'AGG')
>>> m.search("BBBQBBBAUGWWWWWAGGBBBBB")
>>> m.search("BBBQQBBBAUGWWWWWAGGBBBBB")
>>> m.search("BBBQQBBQBAUGWWWWWAGGBBBBB")
<_sre.SRE_Match object at 0xb7de33e0>
>>> m.search("BBBQQBBQBAUGWWWWWAGGBBBBB").groups()
('BQB', 'AUG', 'WWWWW', 'WWW', 'AGG')
>>> m.search("BBBQQBBQBAUGWQWWWWWAGGBBBBB")
>>> m.search("BBBQQBBQBAUGWWWWQWWAGGBBBBB")
>>> m.search("BBBQQBBQBAUGWWQWWQWWAGGBBBBB")
>>> m.search("BBBQQBBQBAUGWWQWAWQWWAGGBBBBB")
<_sre.SRE_Match object at 0xb7de33e0>
>>> m.search("BBBQQBBQBAUGWWQWAWQWWAGGBBBBB").groups()
('BQB', 'AUG', 'WWQWAWQWW', 'QWW', 'AGG')
>>>
```

This way, I could scan for genes, remove the first letter, scan for more genes, remove the first letter again, and scan for more genes. This would hypothetically yield different genes, since the frame would be shifted.

Of you could just unconstrain the first match and it will do them all

Re: Regular Expression – Matching Multiples of 3 Characters exactly.

at once :-

```
(AUG)((\w\w\w)*?)(AGG)
```

You could run this with `re.findall`, but beware that this will only return non-overlapping matches which may not be what you want.

I'm not sure re's are the best tool for the job, but they should give you a quick idea of what the answers might be.

—

Nick Craig-Wood <n...@xxxxxxxxxxxxxxxx> --<http://www.craig-wood.com/nick>

Thank you! Your suggestion was overly helpful.

Also thank you for the package suggestions. BioPython is on my plate to check out, but I needed a kind of quick fix for this one. The documentation for biopython seems pretty thick – I'm not a biologist so I'm not even sure what kind of packages I'm even looking for.

thanks!
Blaine

.