Subject: Re: String should implement the Boyer Moore algo
Posted by mirek on Sat, 01 Mar 2014 19:19:53 GMT
View Forum Message <> Reply to Message

Well, I could not sleep about this, so I have decided to put it to test

I have created "proper" BM and BMH algorithm, then started benchamrking. Loaded 120MB XML file into String, appended "Hello world" (not present in the file otherwise) and started benchmarking.

At first, U++ brute force was 10 times slower than others. Analyzing it I have found that it is not particulary well optimized (as brute force), calling memcmp on each input byte. So I have added some microoptimizations and this is what I got then:

```
------------
Needle: Hello world
U++ Brute force: 127158988
  Time elapsed: 0.037
Folly: 127158988
  Time elapsed: 0.059
Boyer-Moore: 127158988
  Time elapsed: 0.082

  Time elapsed: 0.049
------------
Needle: Hel
U++ Brute force: 127158988
  Time elapsed: 0.036
Folly: 127158988
  Time elapsed: 0.063
Boyer-Moore: 127158988
  Time elapsed: 0.270

  Time elapsed: 0.154
```

First numbers are for the whole "Hello world" search, then i have searched only for "Hel", which accidentally is not in the file too.

I believe that the only real speed advantage the folly algorithm has lies in that simple loop

```
    while (i[nsize_1] != lastNeedle) {
      if (++i == iEnd) {
        // not found
        return -1;
      }
```

```
}
```

implementing de-facto brutal force approach. Any "smart" skips are not frequent enough to change anything.

As for BM and BMH, it looks like the management of skips in reality and in modern CPU is too costly as compared to streamlined comparison loops.... and becomes really a burden when needle is short. Also interesting is that simpler, less clever variant BMH (BMH is in fact simplified BM) is faster... I guess complexity in the loop shows.

Well, at any case, the real result of this endeavour is optimized, albeit still brute-force, String::Find - IMO not bad at all

Mirek
