Subject: Re: String::Cat optimization
Posted by mirek on Wed, 30 Nov 2011 20:15:05 GMT
View Forum Message <> Reply to Message

dolik.rce wrote on Wed, 30 November 2011 12:17Huh... and I always thought memcpy is the fastest way to copy things

Any ideas how does the compiler optimization magic works in this case? I'd like to understand, it might be useful in other situations as well.

Honza

Seriously, I am really ambiguos about this optimization, it is really border case. Plus I have only tested with MSC.

Anyway, looking at assembly code, memcpy is really optimized pretty well, but spends a lot of time detecting heavy-lifting scenario (like target and source both aligned etc...), whereas String is all about adding small pieces of data.

The switch leads to simple jump to 'multiplied' position and then 'linear' code up to end. It is a very little bit faster..

All in all, perhaps more data are needed. It should be easy to #ifdef svo_memcpy to regular memcpy...

My benchmarking code was something like this:

```
String str;
for(int i = 0; i < 10000000; i++) {
 str.Clear();
 RTIMING("Cat 18");
 str.Cat("Hello", 5);
 str.Cat("Hello", 5);
 str.Cat("Hello", 5);
}
for(int i = 0; i < 10000000; i++) {
 str.Clear();
 RTIMING("Cat 40");
 str.Cat("Hello", 5);
 str.Cat("Hello", 5);
 str.Cat("Hello", 5);
 str.Cat("Hello", 5);
 str.Cat("Hello", 5);
 str.Cat("Hello", 5);
 str.Cat("Hello", 5);
 str.Cat("Hello", 5);
}
```

before optimization

```
TIMING Cat 40        :  1.98 s  - 198.46 ns ( 2.17 s  / 10000000 ), min:  0.00 ns, max:  1.00 ms, nesting: 1 - 10000000
TIMING Cat 18        : 591.60 ms - 59.16 ns (772.00 ms / 10000000 ), min:  0.00 ns, max:  1.00 ms, nesting: 1 - 10000000
```

after

```
TIMING Cat 40        :  1.48 s  - 148.37 ns ( 1.68 s  / 10000000 ), min:  0.00 ns, max:  1.00 ms, nesting: 1 - 10000000
TIMING Cat 18        : 482.71 ms - 48.27 ns (676.00 ms / 10000000 ), min:  0.00 ns, max:  1.00 ms, nesting: 1 - 10000000
```