Ruby master - Bug #5946
Remove too early and unnecessary calls to heaps_increment
01/30/2012 01:12 PM - funny_falcon (Yura Sokolov)

<table>
<thead>
<tr>
<th>Status:</th>
<th>Rejected</th>
<th>Backport:</th>
</tr>
</thead>
<tbody>
<tr>
<td>Priority:</td>
<td>Normal</td>
<td></td>
</tr>
<tr>
<td>Assignee:</td>
<td>authorNari (Narihiro Nakamura)</td>
<td></td>
</tr>
<tr>
<td>Target version:</td>
<td>2.0.0</td>
<td></td>
</tr>
<tr>
<td>ruby -v:</td>
<td>ruby 2.0.0dev (2012-01-29 trunk 34394) [i686-linux]</td>
<td></td>
</tr>
</tbody>
</table>

Description
Too early call in lazy_sweep slows down sweep phase, cause while loop breaks on if (has_free_object) { ... return TRUE;}.

Heap expand in gc_clear_mark_on_sweep_slots unnecessary expands number of heaps, which slows down iterations in rb_objspace_call_finalizer, and gc_marks (cause we often need to binary search heap for pointer).

Testing suit: https://gist.github.com/1702301

Before:

$ sh siege.sh
Transaction rate: 114.71 trans/sec
Transaction rate: 117.84 trans/sec
Transaction rate: 121.62 trans/sec
$ sh siege.sh
Transaction rate: 118.72 trans/sec
Transaction rate: 120.32 trans/sec
Transaction rate: 121.12 trans/sec

After:

$ sh siege.sh
Transaction rate: 121.62 trans/sec
Transaction rate: 122.12 trans/sec
Transaction rate: 123.12 trans/sec
$ sh siege.sh
Transaction rate: 123.25 trans/sec
Transaction rate: 121.94 trans/sec
Transaction rate: 123.52 trans/sec

https://github.com/ruby/ruby/pull/89

History
#1 - 01/30/2012 01:54 PM - authorNari (Narihiro Nakamura)
- Status changed from Open to Rejected

Hi.

I reject it because lazy sweeping would spent much time if heap is full.
https://github.com/funny-falcon/ruby/commit/f07354e714e4ee64b1c6805d4dc111cb434f50a#L0R2314

I reject it. Please see r32894.
https://github.com/funny-falcon/ruby/commit/f07354e714e4ee64b1c6805d4dc111cb434f50a#L0R2629

Thanks :)

#2 - 01/30/2012 02:12 PM - funny_falcon (Yura Sokolov)

Could you make a benchmark, please?

My benchmark shows 1% improvement when patch is applied to ruby-trunk and 6% after other patch, which I want to introduce today's evening/tomorrow.
I remade benchmark. It seems that I was wrong.

Excuse me.