This reduces memory overhead of ivars for common types such as T_DATA the same way T_OBJECT does it.

For 9992 accepted clients on an OpenSSL server, this reduces RSS memory from 77160K to 69248K with the attached ossl.rb script. Connecting client process was reduced from 246312K to 230724K RSS.

OpenSSL 1.0.1e-2+deb7u16 on Debian 7

Associated revisions
Revision 9d9aea7f - 05/29/2015 11:42 PM - normal
variable.c: use indices for generic ivars
This reduces memory overhead of ivars for common types such as T_DATA the same way T_OBJECT does it.

For 9992 accepted clients on an OpenSSL server, this reduces memory from 77160K to 69248K with the script in https://bugs.ruby-lang.org/issues/11170

git-svn-id: svn+ssh://ci.ruby-lang.org/ruby/trunk@50678 b2dd03c8-39d4-4d8f-98ff-823fe69b080e

Revision 50678 - 05/29/2015 11:42 PM - normal
variable.c: use indices for generic ivars
This reduces memory overhead of ivars for common types such as T_DATA the same way T_OBJECT does it.

For 9992 accepted clients on an OpenSSL server, this reduces memory from 77160K to 69248K with the script in https://bugs.ruby-lang.org/issues/11170


Revision 50678 - 05/29/2015 11:42 PM - normal
variable.c: use indices for generic ivars
This reduces memory overhead of ivars for common types such as T_DATA the same way T_OBJECT does it.

For 9992 accepted clients on an OpenSSL server, this reduces memory from 77160K to 69248K with the script in https://bugs.ruby-lang.org/issues/11170

```
```

Revision 50678 - 05/29/2015 11:42 PM - normal

variable.c: use indices for generic ivars

This reduces memory overhead of ivars for common types such as T_DATA the same way T_OBJECT does it.

For 9992 accepted clients on an OpenSSL server, this reduces memory from 77160K to 69248K with the script in https://bugs.ruby-lang.org/issues/11170

```
```

Revision 50678 - 05/29/2015 11:42 PM - normal

variable.c: use indices for generic ivars

This reduces memory overhead of ivars for common types such as T_DATA the same way T_OBJECT does it.

For 9992 accepted clients on an OpenSSL server, this reduces memory from 77160K to 69248K with the script in https://bugs.ruby-lang.org/issues/11170

```
```

Revision 50678 - 05/29/2015 11:42 PM - normal

variable.c: use indices for generic ivars

This reduces memory overhead of ivars for common types such as T_DATA the same way T_OBJECT does it.

For 9992 accepted clients on an OpenSSL server, this reduces memory from 77160K to 69248K with the script in https://bugs.ruby-lang.org/issues/11170

```
```

Revision f6cd5825 - 05/30/2015 12:20 AM - normal

variable.c: avoid compatibility table with generic ivars

This recovers and improves performance of Marshal.dump/load on Time objects compared to when we implemented generic ivars entirely using st_table.
This also recovers some performance on other generic ivar objects, but does not bring Marshal.dump/load performance up to previous speeds.

benchmark results:
minimum results in each 10 measurements.
Execution time (sec)
name trunk geniv after
marshal_dump_flo 0.343 0.334 0.335
marshal_dump_load_geniv 0.487 0.527 0.495
marshal_dump_load_time 1.262 1.401 1.257

Speedup ratio: compare with the result of 'trunk' (greater is better)
name geniv after
marshal_dump_flo 1.026 1.023
marshal_dump_load_geniv 0.925 0.985
marshal_dump_load_time 0.901 1.004

- include/ruby/intern.h (rb_generic_ivar_table): deprecate
- internal.h (rb_attr_delete): declare
- marshal.c (has_ivars): use rb_ivar_foreach (w_ivar): ditto (w_object): update for new interface
- time.c (time_mload): use rb_attr_delete
- variable.c (generic_ivar_delete): implement (rb_ivar_delete): ditto (rb_attr_delete): ditto [ruby-core:69323] [Feature #11170]

Revision 50680 - 05/30/2015 12:20 AM - normalperson (Eric Wong)

variable.c: avoid compatibility table with generic ivars

This recovers and improves performance of Marshal.dump/load on Time objects compared to when we implemented generic ivars entirely using st_table.

This also recovers some performance on other generic ivar objects, but does not bring Marshal.dump/load performance up to previous speeds.

benchmark results:
minimum results in each 10 measurements.
Execution time (sec)
name trunk geniv after
marshal_dump_flo 0.343 0.334 0.335
marshal_dump_load_geniv 0.487 0.527 0.495
marshal_dump_load_time 1.262 1.401 1.257

Speedup ratio: compare with the result of 'trunk' (greater is better)
name geniv after
marshal_dump_flo 1.026 1.023
marshal_dump_load_geniv 0.925 0.985
marshal_dump_load_time 0.901 1.004

- include/ruby/intern.h (rb_generic_ivar_table): deprecate
- internal.h (rb_attr_delete): declare
- marshal.c (has_ivars): use rb_ivar_foreach (w_ivar): ditto (w_object): update for new interface
- time.c (time_mload): use rb_attr_delete
- variable.c (generic_ivar_delete): implement (rb_ivar_delete): ditto (rb_attr_delete): ditto [ruby-core:69323] [Feature #11170]

Revision 50680 - 05/30/2015 12:20 AM - normal

variable.c: avoid compatibility table with generic ivars

This recovers and improves performance of Marshal.dump/load on Time objects compared to when we implemented generic ivars entirely using st_table.

This also recovers some performance on other generic ivar objects, but does not bring Marshal.dump/load performance up to previous speeds.

benchmark results:
minimum results in each 10 measurements.
Execution time (sec)
name trunk geniv after
marshal_dump_flo 0.343 0.334 0.335
Revision 50680 - 05/30/2015 12:20 AM - normal

variable.c: avoid compatibility table with generic ivars

This recovers and improves performance of Marshal.dump/load on
Time objects compared to when we implemented generic ivars
entirely using st_table.

This also recovers some performance on other generic ivar objects,
but does not bring Marshal.dump/load performance up to
previous speeds.

benchmark results:
minimum results in each 10 measurements.
Execution time (sec)
name    trunk   geniv   after
marshal_dump_flo 0.343   0.334   0.335
marshal_dump_load_geniv 0.487   0.527   0.495
marshal_dump_load_time 1.262   1.401   1.257

Speedup ratio: compare with the result of 'trunk' (greater is better)
name    geniv after
marshal_dump_flo 1.026   1.023
marshal_dump_load_geniv 0.925   0.985
marshal_dump_load_time 0.901   1.004

include/ruby/intern.h (rb_generic_ivar_table): deprecate
internal.h (rb_attr_delete): declare
marshal.c (has_ivars): use rb_ivar_foreach (w_ivar): ditto (w_object): update for new interface
time.c (time_mload): use rb_attr_delete
variable.c (generic_ivar_delete): implement (rb_ivar_delete): ditto (rb_attr_delete): ditto [ruby-core:69323] [Feature #11170]

Revision 50680 - 05/30/2015 12:20 AM - normal

variable.c: avoid compatibility table with generic ivars

This recovers and improves performance of Marshal.dump/load on
Time objects compared to when we implemented generic ivars
entirely using st_table.

This also recovers some performance on other generic ivar objects,
but does not bring Marshal.dump/load performance up to
previous speeds.

benchmark results:
minimum results in each 10 measurements.
Execution time (sec)
name    trunk   geniv   after
marshal_dump_flo 0.343   0.334   0.335
marshal_dump_load_geniv 0.487   0.527   0.495
marshal_dump_load_time 1.262   1.401   1.257

Speedup ratio: compare with the result of 'trunk' (greater is better)
name    geniv after
marshal_dump_flo 1.026   1.023
marshal_dump_load_geniv 0.925   0.985
marshal_dump_load_time 0.901   1.004

include/ruby/intern.h (rb_generic_ivar_table): deprecate
internal.h (rb_attr_delete): declare

03/26/2021
Revision 50680 - 05/30/2015 12:20 AM - normal

variable.c: avoid compatibility table with generic ivars

This recovers and improves performance of Marshal.dump/load on Time objects compared to when we implemented generic ivars entirely using st_table.

This also recovers some performance on other generic ivar objects, but does not bring Marshal.dump/load performance up to previous speeds.

benchmark results:
minimum results in each 10 measurements.
Execution time (sec)
name trunk geniv after
marshal_dump_flo 0.343 0.334 0.335
marshal_dump_load_geniv 0.487 0.527 0.495
marshal_dump_load_time 1.262 1.401 1.257

Speedup ratio: compare with the result of 'trunk' (greater is better)
name geniv after
marshal_dump_flo 1.026 1.023
marshal_dump_load_geniv 0.925 0.985
marshal_dump_load_time 0.901 1.004

History
#1 - 05/23/2015 01:35 AM - normalperson (Eric Wong)
- File ossl_11170.rb added

Attached standalone test script, increase "ulimit -n" as necessary.

#2 - 05/23/2015 02:19 AM - ko1 (Koichi Sasada)
+1.

T_CLASS/T_MODULE can use same technique, but it seems not so many use-cases.

#3 - 05/29/2015 12:58 AM - normalperson (Eric Wong)

After the original patch, rb_generic_ivar_table() is much more expensive but kept for compatibility reasons. I propose deprecating it, I'm not sure if any 3rd party C-exts use it.

http://80x24.org/spew/m/1432859944-14374-1-git-send-email-e@80x24.org.txt

[PATCH 3/2] avoid compatibility table with generic ivars

This recovers and improves performance of Marshal.dump/load on Time objects compared to when we implemented generic ivars entirely using st_table.

This also recovers some performance on other generic ivar objects, but does not bring Marshal.dump/load performance up to previous speeds.

benchmark results:
minimum results in each 10 measurements.
Execution time (sec)
name trunk geniv after
marshal_dump_flo 0.343 0.334 0.335
marshal_dump_load_geniv 0.487 0.527 0.495
marshal_dump_load_time 1.262 1.401 1.257
<table>
<thead>
<tr>
<th>Name</th>
<th>Geniv</th>
<th>After</th>
</tr>
</thead>
<tbody>
<tr>
<td>marshal_dump_flo</td>
<td>1.026</td>
<td>1.023</td>
</tr>
<tr>
<td>marshal_dump_load_geniv</td>
<td>0.925</td>
<td>0.985</td>
</tr>
<tr>
<td>marshal_dump_load_time</td>
<td>0.901</td>
<td>1.004</td>
</tr>
</tbody>
</table>

#4 - 05/29/2015 11:43 PM - Anonymous

- Status changed from Open to Closed

Applied in changeset r50678.

variable.c: use indices for generic ivars

This reduces memory overhead of ivars for common types such as T_DATA the same way T_OBJECT does it.

For 9992 accepted clients on an OpenSSL server, this reduces memory from 77160K to 69248K with the script in https://bugs.ruby-lang.org/issues/11170


<table>
<thead>
<tr>
<th>Files</th>
<th>Size</th>
<th>Date</th>
<th>Author</th>
</tr>
</thead>
<tbody>
<tr>
<td>ivar-reduce-combined.patch</td>
<td>17.2 KB</td>
<td>05/23/2015</td>
<td>normalperson (Eric Wong)</td>
</tr>
<tr>
<td>ossl_11170.rb</td>
<td>1.74 KB</td>
<td>05/23/2015</td>
<td>normalperson (Eric Wong)</td>
</tr>
</tbody>
</table>