IMPROVING LIVE DEBUGGING OF CONCURRENT THREADS

2016
Max Leske
Andrei Chiș
recap

idea

memory & performance

future work
general concept: threads

- improve debugging of promises
- improve debugging of threads
IDEA
thread

activation record / stack frame

start routine
thread

history of thread
create thread 2

not accessible in debugger!
history of thread 2

virtual call stack

accessible in debugger!

thread 2

thread 1

history of thread 2
Creating a new thread
create thread
bind
resume thread
promises

events

solvable as special cases:

asynchronous network communication

messages (actor model)
MEMORY
object header: 64 bits

slots: 8

identity hash: 22

format: 3

unused: 5

class index: 2

garbage collection: 22

pinned / immutable: 2
### Context

<table>
<thead>
<tr>
<th>object header</th>
</tr>
</thead>
<tbody>
<tr>
<td>sender</td>
</tr>
<tr>
<td>stackp</td>
</tr>
<tr>
<td>closureOrNil</td>
</tr>
</tbody>
</table>

- 16 / 56 variable slots

- 96 / 256 bytes per instance

**no extended header required**
Estimated upper bounds

- Large contexts: 256 kB
  - * 2 (reification)
- Small contexts: 96 kB

- Large contexts: 512 kB
- Small contexts: 192 kB
memory consumption of contexts:
garbage collector

copy

not collected
memory consumption of object graph:
PERFORMANCE
<table>
<thead>
<tr>
<th>garbage collection time</th>
<th>small context (96 bytes)</th>
<th>large context (256 bytes)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>average</td>
<td>median</td>
</tr>
<tr>
<td>included</td>
<td>130.211</td>
<td>130</td>
</tr>
<tr>
<td>excluded</td>
<td>57.127</td>
<td>57</td>
</tr>
</tbody>
</table>

copying stack of 100 000 frames
performance:
FUTURE WORK
VM support
memory consumption
threads -> processes
user interface
memory consumption of contexts:✅
performance:✅
memory consumption of object graph:?