A Prototype for Fast Type-Aware Memory Profiling

Nico Weber
thakis@chromium.org
A Prototype for Fast Type-Aware Memory Profiling

Nico Weber
thakis@chromium.org

Dai Mikurube
dmikurube@chromium.org
<table>
<thead>
<tr>
<th>PID</th>
<th>Process Name</th>
<th>User</th>
<th>% CPU</th>
<th>Threads</th>
<th>Real Mem</th>
<th>Kind</th>
</tr>
</thead>
<tbody>
<tr>
<td>44303</td>
<td>Xcode</td>
<td>thakis</td>
<td>107.8</td>
<td>11</td>
<td>1.06 GB</td>
<td>Intel (64 bit)</td>
</tr>
<tr>
<td>136</td>
<td>Keynote</td>
<td>thakis</td>
<td>0.0</td>
<td>4</td>
<td>81.6 MB</td>
<td>Intel</td>
</tr>
<tr>
<td>132</td>
<td>Google Chrome</td>
<td>thakis</td>
<td>0.0</td>
<td>33</td>
<td>77.6 MB</td>
<td>Intel</td>
</tr>
<tr>
<td>137</td>
<td>Preview</td>
<td>thakis</td>
<td>0.0</td>
<td>2</td>
<td>56.3 MB</td>
<td>Intel (64 bit)</td>
</tr>
</tbody>
</table>
State of the art

• Most profilers* hook malloc()
  • and collect stacks
• But type information would be nice
What’s “Type-Aware” mean?

- LookupType(address) -> type info
- available at runtime
- can aggregate memory use by type
State of the art 2

• Some languages (Java…) have rich enough type metadata to get this for free

• C++: memtrack uses `#define new MAGIC`
  
  • but that uses a define
    
    • for `new`
    
    • ugh
Our stuff

• compiler-based instrumentation
• operator new(...) ⇒
  ___op_new_intercept__(
    operator new(...), size_t, type_info)
• User code implements
  ___op_new_intercept___!
Example: logger

```cpp
void* ::__op_new_intercept__(
    void* address, size_t size,
    const std::type_info& type) {
    fprintf("Allocated %lu bytes "
            "for %s at %016p.\n", 
            size, type.name(), address);
    return address;
}
```
Results

• Early days

• Looks like 30-40% of browser memory are from string-related types

• Hope to use this to provide data on Clang/LLVM’s memory usage soon
Next steps

• Prove usefulness in chromium, clang
• Ideally, make it possible to do type-aware profiling with regular clang
• Dai will reach out to the list
Thanks!
Links

• http://dev.chromium.org/developers/deep-memory-profiler
• http://src.chromium.org/viewvc/chrome/trunk/deps/third_party/llvm-allocated-type
• http://crrev.com/158752
Links 2

• http://www.almostinfinite.com/memtrack.html

• http://ieeexplore.ieee.org/xpl/freeabs_all.jsp?reload=true&&arnumber=6080813
Other approaches

- Don’t do this in client code but in compiler-rt (locking etc)
- Intercept at LLVM level instead of clang level
- Change signature of operator new()
- Have type info available in magic variable in operator new()