HHVM on AArch64

Max Wang
Software Engineer
Agenda

Four questions:

1. What is HHVM?
2. How did we get running on AArch64?
3. Will the demo work?
4. Where do we go from here?
Agenda

Four questions:

1. What is HHVM?

2. How did we get running on AArch64?

3. Will the demo work?

4. Where do we go from here?
Agenda

Four questions:

1. What is HHVM?
2. How did we get running on AArch64?
3. Will the demo work?
4. Where do we go from here?
Agenda

Four questions:

1. What is HHVM?
2. How did we get running on AArch64?
3. Will the demo work?
4. Where do we go from here?
Agenda

Four questions:

1. What is HHVM?
2. How did we get running on AArch64?
3. Will the demo work?
4. Where do we go from here?
Agenda

i.e., Why should you care?

1. What is HHVM?
2. How did we get running on AArch64?
3. Will the demo work?
4. Where do we go from here?
What is HHVM?
In a nutshell

- Just-in-time compiler for PHP and Hack

```
"1" + "2"   == 3
"1" + "2wo" == 3
"15" + "0xf" == 15
("15" == "0xf") == true
```

```
$str = "foo";
list($a, $b) = $str;
var_dump($a);   // "f"
```

```
list($a, $b) = "foo";
var_dump($a);   // NULL
```
What is HHVM?
In a nutshell

- Just-in-time compiler for PHP and Hack

```php
<?hh

class Foo<T> { 
    public async function getBar(
        dict<string,T> $ts
    ): Awaitable<Bar> { 
        return await fetch_bar($this->priv, $ts); 
    }
}
```
What is HHVM?

In a nutshell

- Serves production web traffic for Facebook
- HHVM is fast!
  - Orders of magnitude improvement
- Not just for FB!
What is HHVM?

In a nutshell

• Open source ([https://github.com/facebook/hhvm](https://github.com/facebook/hhvm))
• Used by 3 of the Alexa Top 5
  • Facebook, Baidu, Wikipedia
• Also: Box, Slack, Etsy, Wordpress, ...
What is HHVM?

In a nutshell

- Fast!
- Open source!
- Just-in-time compiler for PHP and Hack
Major performance improvement over PHP5
But:
  - slow ahead-of-time compilation
  - massive binary size
  - static type inference on a dynamic language
Compilation pipeline
High-level PHP bytecode
Compilation pipeline
High-level PHP bytecode

$elem = ...;
if ($elem > 0) {
    ...
}

63 SetL L:4
65 PopC
66 Int 0
75 CGetL2 L:4
77 Gt
78 JmpZ 13 (91)
Compilation pipeline
HHVM intermediate representation

PHP → HHBC → HHIR → x64

63: SetL L:4
(12) t3:Int = LdStk<Int,IRSPOff 0> t1:StkPtr
(14) StLoc<4> t0:FramePtr, t3:Int
66: Int 0
(21) StStk<IRSPOff 0> t1:StkPtr, 0
75: CGetL2 L:4
(24) StStk<IRSPOff 0> t1:StkPtr, t3:Int
77: Gt
78: JmpZ 13 (91)
Compilation pipeline
Bytecode-to-bytecode transformation

PHP → HHBC → HHIR → x64

optionally ahead of time
just in time

63  SetL L:4
65  PopC
66  Int 0
75  CGetL2 L:4
77  Gt
78  JmpZ 13 (91)

63: SetL L:4
(12) t3:Int = LdStk<Int, IRSPOff 0> t1:StkPtr
(14) StLoc<4> t0:FramePtr, t3:Int
66: Int 0
(21) StStk<IRSPOff 0> t1:StkPtr, 0
75: CGetL2 L:4
(24) StStk<IRSPOff 0> t1:StkPtr, t3:Int
(25) StStk<IRSPOff -1> t1:StkPtr, 0
Compilation pipeline

- This is where the magic happens
function addPositive($arr, $n) {
    $sum = 0;
    for ($i = 0; $i < $n; $i++) {
        $elem = $arr[$i];
        if ($elem > 0) {
            $sum = $sum + $elem;
        }
    }
    return $sum;
}
function addPositive($arr, $n) {
    $sum = 0;
    for ($i = 0; $i < $n; $i++) {
        $elem = $arr[$i];
        if ($elem > 0) {
            $sum = $sum + $elem;
        }
    }
    return $sum;
}
function addPositive($arr, $n) {
    $sum = 0;
    for ($i = 0; $i < $n; $i++) {
        $elem = $arr[$i];
        if ($elem > 0) {
            $sum = $sum + $elem;
        }
    }
    return $sum;
}
$c = \texttt{a} \cdot \texttt{b} ;$

$\texttt{len} = \texttt{strlen} (\texttt{c}) ;$

69: CGetL L:1
12) \texttt{t3:Str} = \texttt{LdLoc<Str,1>} \texttt{t0:FramePtr}
13) \texttt{IncRef t3:Str}
71: CGetL2 L:0
16) \texttt{t4:Str} = \texttt{LdLoc<Str,0>} \texttt{t0:FramePtr}
17) \texttt{IncRef t4:Str}
73: Concat
22) \texttt{t5:Str} = \texttt{ConcatStrStr t4:Str, t3:Str}
24) \texttt{DecRef<-> t3:Str}
74: SetL L:2
27) \texttt{StLoc<2> t0:FramePtr, t5:Str}
28) \texttt{IncRef t5:Str}
76: PopC
31) \texttt{DecRef<-> t5:Str}
79: FCallBuiltin 1 1 "\texttt{strlen}"
35) \texttt{t7:Int} = \texttt{LdStrLen t5:Str}
36) \texttt{DecRef<-> t5:Str}
JIT optimizations
HHIR optimization passes

$c = a \cdot b;
\text{len} = \text{strlen}(c);$
$c = \$a \cdot \$b$

$\text{len} = \text{strlen}($c$)$
$c = \cdot \ $b;
$lend = $strlen($c);
JIT optimizations
HHIR optimization passes

/* ... */
Agenda

i.e., Who did all the hard work?

1. What is HHVM?

2. How did we get running on AArch64?

3. Will the demo work?

4. Where do we go from here?
Compilation pipeline

- PHP
- HHBC
- HHIR
- x64

optionally ahead of time

just in time
Compilation pipeline

ARM simulator

PHP → HHBC → HHIR → x64
optionally ahead of time
just in time

HHIR → vixl
Compilation pipeline

**ARM simulator**

- **PHP** → **HHBC** → **HHIR**
- **x64**
- **vixl**

- **Maintenance nightmare**
  - > 600 HHIR ops:
  - We aren’t ARM experts
Compilation pipeline

**PHP** → **HHBC** → **HHIR** → **x64**

 optionally ahead of time  just in time

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>63</td>
<td>SetL L:4</td>
<td>63: SetL L:4</td>
</tr>
<tr>
<td>65</td>
<td>PopC</td>
<td>(12) t3:Int = LdStk&lt;Int,IRSPOff 0&gt; t1:StkPtr</td>
</tr>
<tr>
<td>66</td>
<td>Int 0</td>
<td>(14) StLoc&lt;4&gt; t0:FramePtr, t3:Int</td>
</tr>
<tr>
<td>75</td>
<td>CGetL2 L:4</td>
<td>66: Int 0</td>
</tr>
<tr>
<td>77</td>
<td>Gt</td>
<td>(21) StStk&lt;IRSPOff 0&gt; t1:StkPtr, 0</td>
</tr>
<tr>
<td>78</td>
<td>JmpZ 13 (91)</td>
<td>75: CGetL2 L:4</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(24) StStk&lt;IRSPOff 0&gt; t1:StkPtr, t3:Int</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(25) StStk&lt;IRSPOff -1&gt; t1:StkPtr, 0</td>
</tr>
</tbody>
</table>
just in time

Compilation pipeline

Virtual assembly

- **PHP** → **HHBC** → **HHIR** → **vasm** → **x64**

optionally ahead of time

```
63: SetL L:4
   (12) t3:Int = LdStk<Int, IRSPOff 0> t1:StkPtr
(14) StLoc<4> t0:FramePtr, t3:Int
66: Int 0
   (21) StStk<IRSPOff 0> t1:StkPtr, 0
75: CGetL2 L:4
   (24) StStk<IRSPOff 0> t1:StkPtr, t3:Int
(25) StStk<IRSPOff -1> t1:StkPtr, 0
```

load [%128] => %129
storeb %136(17b), [%rbp - 0x48]
store %129, [%rbp - 0x50]
storeb %136(17b), [%128 + 0x8]
store %129, [%128]
Compilation pipeline

Virtual assembly

- Uncanny resemblance to x64
- Spiritual sibling of WebKit’s Bare Bones Backend
Why don’t you just use LLVM?” 🤔
We tried it:
  - No noticeable performance gains
  - LLVM’s MCJIT is too heavyweight
Compilation pipeline

LLVM? Have you heard of it?

- Experimental LLVM backend stress-tested vasm
  - Calling conventions
  - Register widths
  - ...

PHP → HHBC → HHIR → vasm → LLIR?

optionally ahead of time

just in time
## Compilation pipeline

**ARM backend**

<table>
<thead>
<tr>
<th>PHP</th>
<th>HHBC</th>
<th>HHIR</th>
<th>vasm</th>
<th>x64</th>
<th>arm</th>
</tr>
</thead>
</table>

*optionally ahead of time*

*just in time*
Compilation pipeline
Backends for everyone!

- PHP → HHBC → HHIR → vasm
  - optionally ahead of time
  - just in time

- vasm → x64 → ppc64 → arm
Contributors

The most important slide in this talk

- Lakshmi Pathy — @lpathy
- Dave Estes — @dave-estes
- Jim Saxman — @jim-saxman
- Christoph Müllner — @cmuellner
- Steve Walk — @swalk-cavium
- Andrew Pinski — @apinski-cavium
- ...

ARM backend
Baseline functionality

- Final vasm-to-AArch64 lowering pass
- Code smashing
- Boundary-crossing b/w C++ and jitted code
- Bonus: Continuous integration testing!
ARM backend
Optimizations

- Strength reduction on flag-setting instructions
- 64-bit immediate lifting
- Branch offset optimizations
- ...

Agenda

Hold onto your backends

1. What is HHVM?
2. How did we get running on AArch64?
3. Will the demo work?
4. Where do we go from here?
Agenda

i.e., How can you get involved?

1. What is HHVM?
2. How did we get running on AArch64?
3. Will the demo work?
4. Where do we go from here?
Future work

Code size and layout

• Cache is king!
• Instruction sequences generally larger on AArch64
• HHVM is sensitive to code layout
  • Huge pages
  • Indirect branch rewriting
  • Locality tuning
  • ...

...
Future work

More ARM-specific optimizations

- Profile OSS workloads on ARM using perf
- Make some measurements
Resources

Feel free to contribute!

• Website: http://hhvm.com/
• GitHub: https://github.com/facebook/hhvm
• IRC: #hphp-dev on Freenode
• Mailing list: https://groups.google.com/d/forum/hhvm-arm
• My email: mwang@fb.com
Quick recap

HHVM on AArch64
Quick recap
HHVM on AArch64

1

HHVM
Quick recap
HHVM on AArch64

1. HHVM
2. It runs on AArch64 (thanks to the community)
Quick recap

HHVM on AArch64

1. HHVM

2. It runs on AArch64 (thanks to the community)

3. Seriously, the demo worked and everything
Any questions?

1. HHVM

2. It runs on AArch64 (thanks to the community)

3. Seriously, the demo worked and everything

4. Any questions?
facebook