UNIVERSITAS SCIENTIARUM SZEGEDIENSIS
UNIVERSITY OF SZEGED
Department of Software Engineering

ARM JIT in nutshell

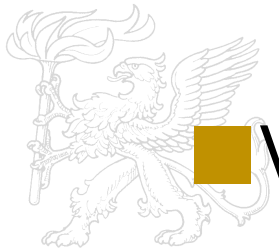
Zoltán Herczeg
University of Szeged

Introduction to ARM-JIT

- Macro assembler based implementation
  - Apple calls it as “Lightweight compiler”
  - Common interface for JIT
    - Most JIT code is written in macro assembler
  - X86 specific interface
    - Src1 and Dst is the same
    - X86 addressing modes

ARM JIT implementation

- Macro Assembler port
  - Mainly ARMv4 instructions
- ARM optimizations
  - Limited to Macro Assembler level
  - Constant pool
    - Experimenting with a constant pool less implementation on ARMv7
  - Vector Floating Point (VFP) support



Enabling JIT

- Enabled by default on WebKit Qt port
- Automatically selects the best options for the target environment using pre-defined preprocessor directives
  - Compiler, CPU, OS
- Disabling features
  - Passing options to qmake
  - JavaScriptCore/wtf/Platform.h

Results

- SunSpider: 1.69 as fast
  - JIT: 9800.4ms
  - Interpreter: 16594.8 ms
- V8: 2.76 as fast
  - JIT: 32655.1 ms
  - Interpreter: 90290.0 ms
- WindScorpion: 1.70 as slow
  - JIT: 159524.8 ms
  - Interpreter: 270898.3 ms

Ongoing works

- Constant pool less implementation for ARMv7
- Enabling Apples's Thumb2 implementation on Linux
- Profiling WebKit using oprofile
  - Adding WebKit JIT support for oprofile
- Thompson based pattern matching

Future

- Maintaining JIT
  - Any feedback is welcome
- Benchmarking JIT memory consumption
- Regulary compare to other ARM-JIT engines
  - V8, TraceMonkey