

Master's Thesis

Splitting of Large Wasm Functions to Enable JIT Compilation

Student: Bernhard Schertler

Advisor: Univ.-Prof. Dr. Stefan Marr

Begin: 01.03.2026

Univ.-Prof. Dr.

Stefan Marr

Head of the Institute for
System Software

P +43 732 2468 4340

F +43 732 2468 4345

stefan.marr@jku.at

Office:

ARⁱⁿ Karin Gusenbauer

Ext: 4342

karin.gusenbauer@jku.at

The goal of this master's thesis is to explore how WebAssembly (Wasm) can be supported more effectively on JIT-compiling systems with strict compilation-unit size limits that are not tuned for typical Wasm functions. The work focuses on JVM-based Wasm runtimes, where large Wasm functions can translate to JVM methods that exceed class-file limits or remain interpreter-executed because of HotSpot's default huge-method policy.

The underlying motivation is based on two distinct JVM constraints. Endive runs on top of standard JVMs and first compiles Wasm to JVM bytecode before execution, mapping each function to a JVM method. A method whose length exceeds 65,535 bytes cannot be represented as a valid JVM class-file method. HotSpot's default DontCompileHugeMethods policy additionally excludes methods larger than 8,000 bytes from JIT compilation. Large Wasm functions may therefore require a Wasm-interpreter fallback or execute as valid JVM bytecode without JIT compilation.

Large Wasm functions can arise from aggressive LLVM-based inlining, intrinsically large function bodies, interpreter dispatch loops, and wide or sparse multi-way control flow. Such functions can contain regions with varying execution frequencies, many of which are not performance critical. Selectively outlining suitable regions may therefore reduce JVM method sizes while preserving the hot execution core. Thus, we believe that splitting large Wasm functions running on the JVM has the potential to improve both peak performance and warm-up speed.

To work towards this goal, the work for this thesis needs to:

- a) Explore instrumentation techniques for collecting function- and region-level execution frequencies and assess whether richer path or trace information provides additional value for profile-guided splitting.
- b) Explore techniques for outlining Wasm regions and splitting large functions while preserving correctness and minimizing overhead.

- c) Develop benchmarks based on synthetic and real-world workloads relevant to execution on the JVM.
- d) Develop the methodology and infrastructure to assess correctness and the performance trade-offs of different splitting strategies.

Agreed Supervision Mode:

The project progress is usually discussed every two weeks. As part of the Master's Seminar, three presentations are given to discuss the project progress with the institute. As opportunities arise, progress is discussed with interested external parties, also to gain feedback on relevant experiments to validate performance on real-world use cases. The final version of the dissertation is expected before the end of February 2027.