

Compiling Linear Sax to WASM

CMU CS 15-817: HOT Compilation
Mikail Khan

Abstract

Sax is an intermediate representation (IR) that helps bridge the gap between high level, richly typed languages and lower level imperative execution. WebAssembly (WASM) is a bytecode instruction set focused on speed, security, and portability. It is an appealing compilation target because it can be run quickly and securely on many platforms, including embedded devices and web browsers, and is used in industry for accelerating web performance or running untrusted code. In this paper, I detail the structure and implementation of a Sax to WASM compiler, supporting purely linear types and 32-bit integers.

1 Introduction

The most interesting aspect of compiling linear Sax to WASM is memory management. WASM is lower-level than Sax and exposes details like stack and local layout, so when starting this project I spent a lot of time and effort trying to optimize these. However, WASM runtimes often include JIT compilers that make precise stack and local layout irrelevant to performance. Additionally, `wasm-opt` is a CLI tool and library that runs many optimization passes, including local allocation. So the main focus of this compiler is on exploiting linear typing's advantages for memory allocation, with an eye on maintaining extensibility for supporting adjoint Sax through WASM GC.

The variant of Sax implemented by this compiler is a modified version of the core linear, positive fragment introduced in Lab 1, with two additions. First, it supports 32-bit signed integers, leveraging WASM's `i32` type. It also supports closures. Both of these types are always unrestricted, supporting all structural rules. Units are also unrestricted.

The performance of a compiler targeting WASM largely depends on the source language. Manually memory-managed languages like C or Rust are often only 20-40% slower through WASM, while more dynamic languages like OCaml or Scala can be 3-8x slower (no sources). Thus, the upper performance bound for linear Sax compiled to WASM should be similar to the performance of C compiled to WASM.

1.1 WASM Structure

The structure and interpretation of WASM programs is largely similar to that of well-known instruction sets like x86 or Java bytecode, with the major difference that all its control flow is structured. It is primarily a stack VM; instructions operate on some combination of bytecode immediates and stack operands. There are no stack manipulation instructions. Instead, there is a per-function set of local values which can be accessed through static indices, emulating registers.

WASM also supports both a manually managed heap and a garbage collector.

1.2 WASM Runtimes

There are many different WASM runtimes. The most common are browser engines like Chrome's V8 and Safari's JavaScript Core, however, there are a variety of WASM runtimes aimed at non-web usage. In particular, I chose a runtime called Wasmtime to evaluate the compiler.

WASM does not specify a system-level interface for I/O on its own. Instead, it supports a robust foreign-function interface, through which it requires WASM Runtimes to inject I/O functions. There is a proposal for a standardized WASM system interface called WASI which is gaining support in major runtimes. However, it is not well documented so I opted to embed custom I/O functions in the runtime. Thus, there is a small amount of code which is not runtime agnostic, for now.

Wasmtime is a popular and well-supported runtime focused on secure embedding. It is built as a JIT compiler on top of the Cranelift code generator, which is similar to LLVM but optimized for compilation speed rather than generating optimal code. It already supports WASI, and can be used through a Rust crate. The WASM files generated by this compiler must be run through a small Rust program which injects I/O functions and runs Wasmtime.

1.3 Integers

To achieve competitive performance on real programs, I supplement Sax with a native 32-bit integer type. The only changes to sax are the built-in `int` type and a few built-in metavariables used through Sax's existing `call` command. The supported builtins are as follows:

- `call _const_{n} dest`: parses the integer out of the metavariable name and writes it to `dest`
- `call _add_ dest l r`: writes `(i32.add l r)` to `dest`
- `call _sub_ dest l r`: writes `(i32.sub l r)` to `dest`
- `call _eqz_ dest n`: writes `(i32.eqz n)` to `dest`
 - In WASM, `(i32.eqz n)` pushes 1 when `n` is zero, and 0 otherwise.
 - I did not add a builtin boolean type to Sax. Instead, this just pushes a sum tag with a unit injection. It makes the most sense when used with a type like `+{ 'false' : 1, 'true' : 1 }`, but is not checked.

Example 1 shows a simple example of using integers in Sax.

```
proc sum_tailrec (d : int) (n : int) (acc : int) =
  cut tst : bool
    call _eqz_ tst n
  read tst {
    | 'true(u) =>
      id d acc
    | 'false(u) => cut n1 : int
                    cut one : int
                    call _const_1 one
                    call _sub_ n1 n one
                    cut nxt : int
                    call _add_ nxt acc n
                    call sum_tailrec d n1 nxt
  }
```

Example 1: $\text{sum_tailrec}(n, \text{acc}) = \sum(0..n) + \text{acc}$

1.4 Compilation Stages and Execution

The compiler and runtime have several components. First, the compiler does a simple pass over the Sax, generating a simple stack-based sequential IR and doing a simple static analysis. Then, it does another pass over the stack-based IR to generate WASM instructions. Finally, the compiler expects the WASM module to be optimized by `wasm-opt`. It should not be strictly necessary, but there seem to be differences in how Wasmtime and `wasm-opt` validate modules, so output directly from the compiler often does not run in Wasmtime without using `wasm-opt` first.

Unlike the Sax compiler from class, this compiler requires a `main` proc, with a destination of any type. The runner will find the `main` proc and print its output.

1.5 Limitations

There are some limitations driven by time constraints, but they should be fixable without modifying the whole approach.

- The main proc's type must be a typename
 - because of how printing works, as described below
- Subtyping probably has bugs
 - the compiler assumes that all instances of a type have the same layout, but does not enforce this on downcasts
- Memory does not grow
 - the allocator does not ever grow the memory, so it is limited to one WASM page of 64 KiB.

2 Implementation

The translation from Sax to WASM is guided by a few principles. First, the destination of the current translated command is just the top of the stack. Thus, translating a metavariable writing to destination d results in a sequence of WASM instructions that result in d on the top of the stack.

The translation from the stack IR to WASM carefully tracks the stack and locals. In particular, it tries to track the relationship between different variables, in order to accurately locate them in cases where a stack value may represent both a variable and e.g. the left pi of a pair. I believe this approach is convoluted and could be simplified by taking advantage of properties of the initial translation from Sax to the stack IR. I designed it when I was trying to optimize exact stack and local layout.

Possible stack values are as follow:

- `Addr s`: an address bound to variable s
- `GcRef s`: a GC'd reference bound to variable s
- `InjTag s`: the tag of plus-type variable s
- `InjData s`: the injection of plus-type variable s
- `PairFst s`: the π_1 of pair-type variable s
- `PairSnd s`: the π_2 of pair-type variable s
- `Unit s`: a unit bound to variable s
 - These never actually materialize in WASM, and are instead used to pad with zero consts when needed
- `Int s`: an integer bound to variable s
 - Integers are passed by value

The basic translation judgement is:

$$S_I \vdash \llbracket c \rrbracket = (W; S_O)$$

meaning that Sax command c yields a list of WASM instructions W and a stack representation S_O . Our rules will be used pretty informally, and leave out some info about state.

2.1 Allocation and Value Layout

The layout of values is the same on the stack and heap. Cells are always represented as two i32s. They may be a pair of primitives, or a primitive followed by a tag, where the primitives could represent an address, unit, or integer.

Since all cells are the same size, our allocator is very simple. We keep an implicit freelist, where the first value of a freed cell is the offset to the next cell. There are two functions:

- `alloc(v1: i32, v2: i32)` writes $v1$ and $v2$ to the cell on top of the freelist, bumps the freelist pointer by the offset previously written to the cell, and returns the address allocated.

- `free(addr: i32)` writes the current freelist head to the front of the cell, and sets the new freelist pointer to `addr`.

I made some strange choices in this design. First, I opted to start the freelist with every allocated cell by writing an offset of 8 to every cell on initialization. This saves storing an additional pointer to space that has yet to be allocated or freed, but is probably slower when memory is resized.

Additionally, I wrote the allocator in Rust, embedding it into the runtime, instead of using WASM's built-in memory instructions. This was just done to save myself from writing WASM by hand. It makes the generated code less portable, but is likely a little bit faster, since the Rust code is not subject to the same safety checks that WASM would have.

2.2 Cuts and Locals

Using our invariants the cut translation is pretty straightforward. Since the destination of any command is the top of the stack, we know that the cut variable will be on top, either as components as a single address. With our stack representation, we can differentiate each case.

The binding created by cut should be randomly accessible by the second command, so we greedily make a local for each cut in case it is not on top of the stack when needed. This creates a lot of churn and local rewriting but `wasm-opt` handles it.

If the top of the stack after the first command is already an address, our translation just creates a local for it. If it is a decomposed cell of an injection tag and value or pair components, we allocate them and then push the address to our locals.

$$\frac{S \vdash \llbracket P \rrbracket = W_P; \text{Addr } x :: S_P \quad S_P \vdash \llbracket Q \rrbracket = W_Q; S_Q}{S \vdash \llbracket \text{cut } (x : T) P(x) Q(x) \rrbracket = W_P :: \text{local.set } n :: W_Q; S_Q} \text{Cut}_{\text{addr}}$$

$$\frac{S \vdash \llbracket P \rrbracket = W_P; \diamond :: \diamond :: S_P \quad S_P \vdash \llbracket Q \rrbracket = W_Q; S_Q}{S \vdash \llbracket \text{cut } (x : T) P(x) Q(x) \rrbracket = W_P :: \text{call alloc} :: \text{local.set } n :: W_Q; S_Q} \text{Cut}_{\text{pair}}$$

Note that Cut_{pair} applies to both pairs and injections, or even pairs of addresses where the top address is not the destination.

2.3 Read

Read commands dereference and free an address, put its components into locals, and then build a switch if there are multiple cases.

2.4 Write

Write commands move something onto the top of the stack.

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Quod idem licet transferre in voluptatem, ut postea variari voluptas distinguere possit, augeri amplificarique non possit. At.

2.5 Calls and Tail Calls

The call translation is straightforward. ...

Sax's only iteration construct is recursion. To avoid stack-overflows, we need to properly tail recurse on tail-recursive Sax functions. This is straightforward using WASM's tail call proposal, which has been merged and is supported by most runtimes. However, I found that it is still a bit slow in Wasmtime.

To detect when a tail is a tail-call, we just check if the destination of a call is the destination of the whole function. Then, it suffices to replace the WASM `call` instruction with `return_call`.

2.6 Id

Id is very simple. We just need to ensure that the required address is on top of the stack; there are two cases. If it is already on top, we continue, otherwise we must fetch it from our locals.

2.7 Printing

Printing is a little complicated. Since tags are indexed by type, we need the full type of an object to print it. It would be possible to generate a print function for each type, but I instead opted to write a single print function in Rust which accepts the type index as a parameter. To do so, we must serialize every type def so that it is accessible to the runtime. The compiler uses `yojson` to serialize the type defs to JSON, and then exports it to the runtime using WASM's data segments. This undoubtedly makes printing pretty slow.

2.8 Optimizations

We implement a few optimizations, mostly aimed at minimizing allocations.

2.8.1 Cut/Read

If the variable instantiated by a cut is read in the same function, we skip allocating it. Instead, we put its components in locals. This modifies the translation rules for Cut and Read.

2.8.2 Cut/Id

The simple Cut/Id optimization is implicit through our translation of Id.

2.8.3 Vertical Reuse

Though we could easily implement vertical reuse in the compilation,

2.9 Runtime

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Quam ipsum accusantium rem et similique in enim est. Ut enim aeque doleamus animo, cum corpore dolemus, fieri tamen permagna accessio potest, si aliquod aeternum et infinitum impendere malum nobis opinemur. Quod idem licet transferre in voluptatem, ut postea variari voluptas distinguere possit, augeri amplificarique non possit. At etiam Athenis, ut e patre audiebam facete et urbane Stoicos irridente, statua est in quo a nobis philosophia defensa et collaudata est, cum id, quod maxime placeat, facere possimus, omnis voluptas assumenda est, omnis dolor repellendus. Temporibus autem quibusdam et aut officiis debitis aut rerum necessitatibus saepe eveniet, ut et voluptates repudiandae sint et molestiae non recusandae. Itaque earum rerum defuturum, quas natura non depravata desiderat. Et quem ad me accedis, saluto: 'chaere,' inquam, 'Tite!' lictores, turma omnis chorusque: 'chaere, Tite!' hinc hostis mi Albucius, hinc inimicus. Sed iure Mucius. Ego autem mirari satis non queo unde hoc sit tam insolens domesticarum rerum fastidium. Non est omnino hic docendi locus; sed ita prorsus existimo, neque eum Torquatum, qui hoc primus cognomen invenerit, aut torquem illum hosti detraxisse, ut aliquam ex eo est consecutus? – Laudem et caritatem, quae sunt vitae sine metu degendae praesidia firmissima. – Filium morte multavit. – Si sine causa, nollem me ab eo delectari, quod ista Platonis, Aristoteli, Theophrasti orationis ornamenta neglexerit. Nam illud quidem physici, credere aliquid esse minimum, quod profecto numquam putavisset, si a Polyaeno, familiari suo, geometrica discere maluisset quam illum etiam ipsum dedocere. Sol Democrito magnus videtur, quippe homini erudito in geometriaque perfecto, huic pedalis fortasse; tantum enim esse omnino in nostris poetis aut inertissimae segnitiae est aut fastidii delicatissimi. Mihi quidem videtur, inermis ac nudus est. Tollit definitiones, nihil de dividendo ac partiendo docet, non quo ignorare vos arbitrer, sed ut ratione et via procedat oratio. Quaerimus

igitur, quid sit extremum et ultimum bonorum, quod omnium philosophorum sententia tale debet esse, ut eius magnitudinem celeritas, diuturnitatem allevatio consoletur. Ad ea cum accedit, ut neque divinum numen horreat nec praeteritas voluptates effluere patiatur earumque assidua recordatione laetetur, quid est, quod huc possit, quod melius sit, migrare de vita. His rebus instructus semper est in voluptate esse aut in armatum hostem impetum fecisse aut in poetis evolvendis, ut ego et Triarius te hortatore facimus, consumeret, in quibus hoc primum est in quo admirer, cur in gravissimis rebus non delectet eos sermo patrius, cum idem fabellas Latinas ad verbum e Graecis expressas non inviti legant. Quis enim tam inimicus paene nomini Romano est, qui Ennii Medeam aut Antiopam Pacuvii spernat aut reiciat, quod se isdem Euripidis fabulis delectari dicat, Latinas litteras oderit? Synephebos ego, inquit, potius Caecilii aut Andriam Terentii quam utramque Menandri legam? A quibus tantum dissentio, ut, cum Sophocles vel optime scripserit Electram, tamen male conversam Atilii mihi legendam putem, de quo Lucilius: 'ferreum scriptorem', verum, opinor, scriptorem tamen, ut legendus sit.

3 Evaluation

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magnam aliquam quaerat voluptatem. Ut enim aequale doleamus animo, cum corpore dolemus, fieri tamen permagna accessio potest, si aliquod aeternum et infinitum impendere malum nobis opinemur. Quod idem licet transferre in voluptatem, ut postea variari voluptas distinguere possit, augeri amplificarique non possit. At etiam Athenis, ut e patre audiebam facete et urbane Stoicos irridente, statua est in quo a nobis philosophia defensa et collaudata est, cum id, quod maxime placeat, facere possimus, omnis voluptas assumenda est, omnis dolor repellendus. Temporibus autem quibusdam et aut officiis debitis aut rerum necessitatibus saepe eveniet, ut et voluptates repudiandae sint et molestiae non recusandae. Itaque earum rerum defuturum, quas natura non depravata desiderat. Et quem ad me accedis, saluto: 'chaere,' inquam, 'Tite!' lictores, turma omnis chorusque: 'chaere, Tite!' hinc hostis mi Albucius, hinc inimicus. Sed iure Mucius. Ego autem mirari satis non queo unde hoc sit tam insolens domesticarum rerum fastidium. Non est omnino hic docendi locus; sed ita prorsus existimo, neque eum Torquatum, qui hoc primus cognomen invenerit, aut torquem illum hosti detraxisse, ut aliquam ex eo est consecutus? – Laudem et caritatem, quae sunt vitae sine metu degendae praesidia firmissima. – Filium morte multavit. – Si sine causa, nollem me ab eo delectari, quod ista Platonis, Aristoteli, Theophrasti orationis ornamenta neglexerit. Nam illud quidem physici, credere aliquid esse minimum, quod profecto numquam putavisset, si a Polyaeno, familiari suo, geometrica discere maluisset quam illum etiam ipsum dedocere. Sol Democrito magnus videtur, quippe homini erudito in geometriaque perfecto, huic pedalis fortasse; tantum enim esse omnino in nostris poetis aut inertissimae segnitiae est aut fastidii delicatissimi. Mihi quidem videtur, inermis ac nudus est. Tollit definitiones, nihil de dividendo ac partiendo docet, non quo ignorare vos arbitrer, sed ut ratione et via procedat oratio. Querimus igitur, quid sit extremum et ultimum bonorum, quod omnium philosophorum sententia tale debet esse, ut eius magnitudinem celeritas, diuturnitatem allevatio consoletur. Ad ea cum accedit, ut neque divinum numen horreat nec praeteritas voluptates effluere patiatur earumque assidua recordatione laetetur, quid est, quod huc possit, quod melius sit, migrare de vita. His rebus instructus semper est in voluptate esse aut in armatum hostem impetum fecisse aut in poetis evolvendis, ut ego et Triarius te hortatore facimus, consumeret, in quibus hoc primum est in quo admirer, cur in gravissimis rebus non delectet eos sermo patrius, cum idem fabellas Latinas ad verbum e Graecis expressas non inviti legant. Quis enim tam inimicus paene nomini Romano est, qui Ennii Medeam aut Antiopam Pacuvii spernat aut reiciat, quod se isdem Euripidis fabulis delectari dicat, Latinas litteras oderit? Synephebos ego, inquit, potius Caecilii aut Andriam Terentii quam utramque Menandri legam? A quibus tantum dissentio, ut, cum Sophocles vel optime scripserit Electram, tamen male conversam

Atilii mihi legendam putem, de quo Lucilius: 'ferreum scriptorem', verum, opinor, scriptorem tamen, ut legendus sit.

4 Related Work

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magnam aliquam quaerat voluptatem. Ut enim aequale doleamus animo, cum corpore dolemus, fieri tamen permagna accessio potest, si aliquod aeternum et infinitum impendere malum nobis opinemur. Quod idem licet transferre in voluptatem, ut postea variari voluptas distinguere possit, augeri amplificarique non possit. At etiam Athenis, ut e patre audiebam facete et urbane Stoicos irridente, statua est in quo a nobis philosophia defensa et collaudata est, cum id, quod maxime placeat, facere possimus, omnis voluptas assumenda est, omnis dolor repellendus. Temporibus autem quibusdam et aut officiis debitis aut rerum necessitatibus saepe eveniet, ut et voluptates repudiandae sint et molestiae non recusandae. Itaque earum rerum defuturum, quas natura non depravata desiderat. Et quem ad me accedis, saluto: 'chaere,' inquam, 'Tite!' lictores, turma omnis chorusque: 'chaere, Tite!' hinc hostis mi Albucius, hinc inimicus. Sed iure Mucius. Ego autem mirari satis non queo unde hoc sit tam insolens domesticarum rerum fastidium. Non est omnino hic docendi locus; sed ita prorsus existimo, neque eum Torquatum, qui hoc primus cognomen invenerit, aut torquem illum hosti detraxisse, ut aliquam ex eo est consecutus? – Laudem et caritatem, quae sunt vitae sine metu degendae praesidia firmissima. – Filium morte multavit. – Si sine causa, nollem me ab eo delectari, quod ista Platonis, Aristoteli, Theophrasti orationis ornamenta neglexerit. Nam illud quidem physici, credere aliquid esse minimum, quod profecto numquam putavisset, si a.

5 Conclusion

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magnam aliquam quaerat voluptatem. Ut enim aequale doleamus animo, cum corpore dolemus, fieri tamen permagna accessio potest, si aliquod aeternum et infinitum impendere malum nobis opinemur. Quod idem licet transferre in voluptatem, ut postea variari voluptas distinguere possit, augeri amplificarique non possit. At etiam Athenis, ut e patre audiebam facete et urbane Stoicos irridente, statua est in quo a nobis philosophia defensa et collaudata est, cum id, quod maxime placeat, facere possimus, omnis voluptas assumenda est, omnis dolor repellendus. Temporibus autem quibusdam et aut officiis debitis aut rerum necessitatibus saepe eveniet, ut et voluptates repudiandae sint et molestiae non recusandae. Itaque earum rerum.