Higher Order Company

Welcome to the massively parallel future of computing!

Victor Taelin & friends

Problem

Software isn't ready for parallel hardware

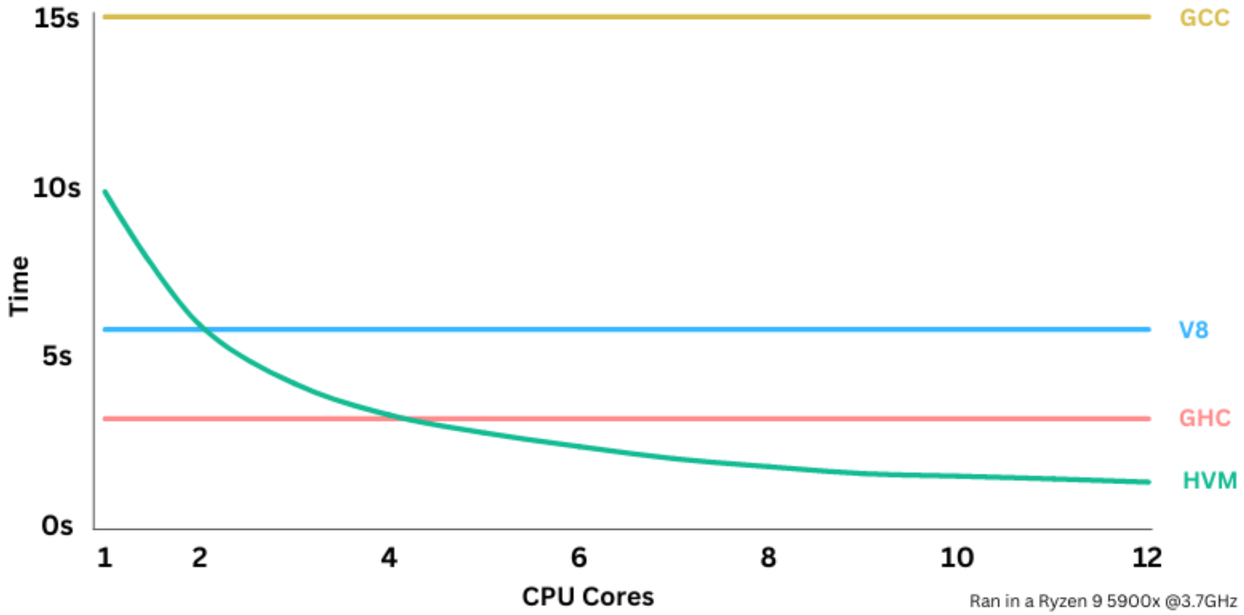
- CPUs with **increasingly more cores** build pressure to parallelize software
- Most modern programming languages are **single threaded** by default
- Parallel programming is **very expensive**, because:

  1. concurrency errors are complex (race conditions, deadlocks, etc.)
  2. non-deterministic behavior is very hard to debug
  3. parallelism overhead can actually reduce performance

Solution

HVM: a massively parallel runtime

- A runtime capable of automatic parallelism with near-ideal speedups
- Allows existing software to scale horizontally with available cores



To illustrate, we implemented a tree radix sort and compared running it on stablished runtimes vs HVM. Only on HVM, the more cores you have, the faster it runs! This is not a cherry picked example, but a general rule that is seen in most tests.

Benchmark: https://github.com/VictorTaelin/HOC/tree/master/bench

Product

ThreadBender: make your code massively parallel

- Transpiles popular languages (Python, JavaScript, etc.) to HVM on the fly
- Low entry barrier: just **npm install** and **bend** which functions to parallelize!

```
// slow code...
bigdata = function(size) {
  if (size <= 1) {
    return 1;
  } else {
    return {
      x: bigdata(size / 2),
      y: bigdata(size / 2),
    };
  }
}
console.log(sum(bigdata(2 ** 26)));
```

Time to run: 2.8 seconds

On V8, the default runtime

Product

ThreadBender: make your code massively parallel

- Transpiles popular languages (Python, JavaScript, etc.) to HVM on the fly
- Low entry barrier: just **npm install** and **bend** which functions to parallelize!

```
//          \/ just bend it!
bigdata = bend function(size) {
  if (size <= 1) {
    return 1;
  } else {
    return {
      x: bigdata(size / 2),
      y: bigdata(size / 2),
    };
  }
}
console.log(sum(bigdata(2 ** 26)));
```

Time to run: 0.4 seconds

With ThreadBender + HVM

That's a 700% speedup with 8 cores

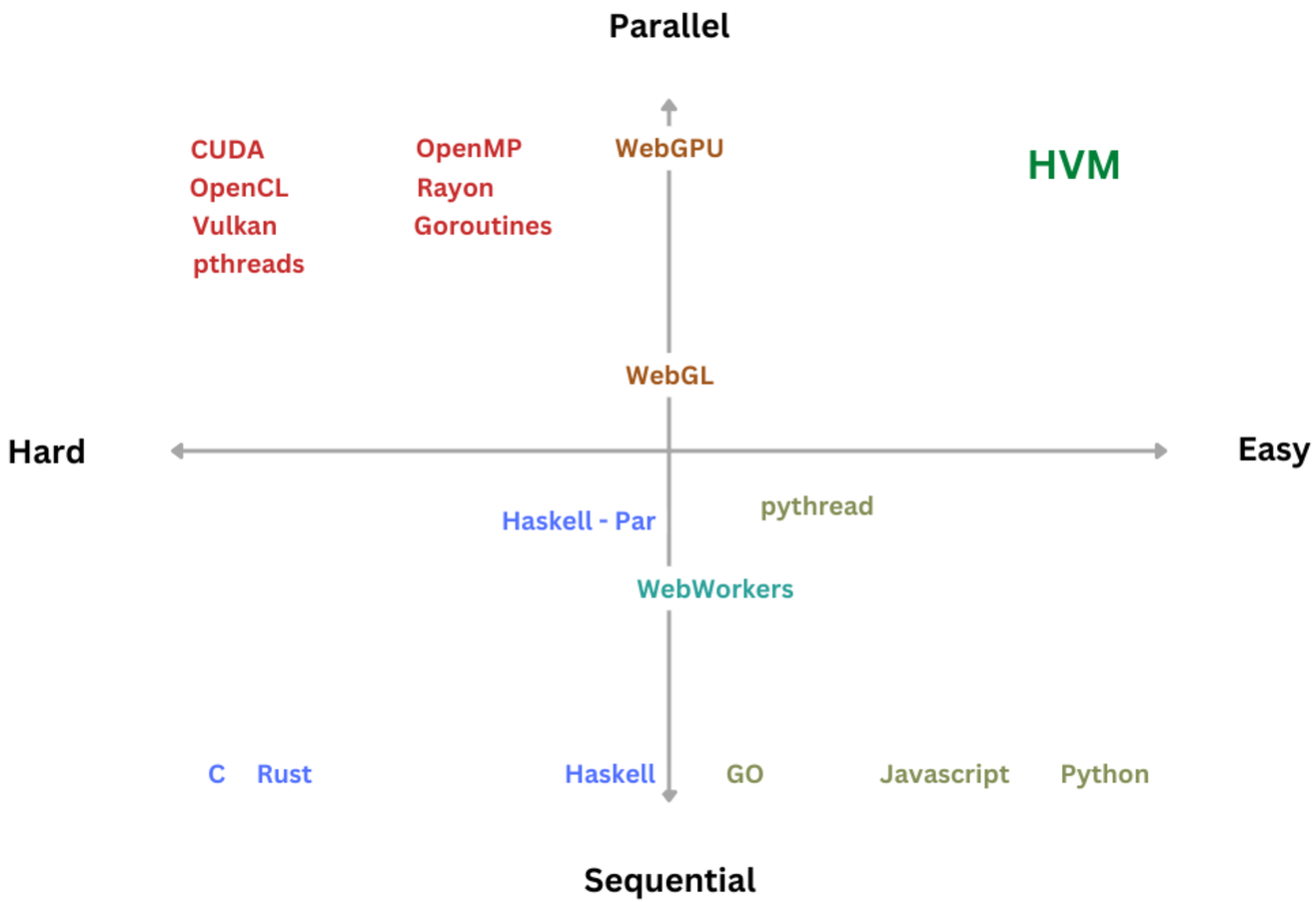
Benchmark: https://github.com/VictorTaelin/HOC/tree/master/bench

Business Model

Monetize on ThreadBender licenses, support and services

	Individual	Company	Enterprise
HVM (always free, open-source)	✅	✅	✅
ThreadBender (freemium, paid licenses)	✅	✅	✅
Consulting Services		✅	✅
Email Support		✅	✅
24/7 Support			✅
	starting at **$0**	starting at **$??**	starting at **$??**

Competition



There are several tools and languages used for parallelism, but they are either limited in scope, or require expensive development, due to very strict limitations and hard-to-debug issues.

As for automatic parallelism, this isn't a new idea. There is a wide body of research trying to achieve it, but, until now, success was limited.

HVM can auto-parallelize common features like data allocation, recursion, lambdas. That's why **ThreadBender** is so easy to use: no need to learn locks, mutexes, threading. ***Just bend it.***

Technology

How we solve automatic parallelism

We use a new model of computation, the **Interaction Calculus**, which *completes* the **Lambda Calculus** with **Interaction Net** semantics. Looks complex, but the key insights are simple:

- 1. Make **everything pure** (like Haskell) - *no side effects*
- 2. Make **everything linear** (like Rust) - *no shared references*
- 3. Add a **first-class lazy cloner** ("*fan nodes*") - makes it *Turing complete*
- 4. Keep a **thread pool** with a **work stealing queue** of **interaction rules**

This **new theoretical foundation** built on the shoulder of giants (Yves Lafont, Girard, Lamping...) let us create **HVM**, the first general-purpose, parallel runtime with near-linear speedup!

In-depth explanation: https://github.com/Kindelia/HVM/blob/master/HOW.md
Interaction Combinators: https://www.semanticscholar.org/paper/Interaction-Combinators-Lafont/6cfe09aa6e5da6ce98077b7a048cb1badd78cc76

Adoption

Our prototype already conquered developer's hearts!

▲ High-order Virtual Machine (HVM): Massively parallel, optimal functional runtime (github.com/kindelia)

493 points by Kinrany 10 months ago | hide | past | favorite | 151 comments

HVM Public

A massively parallel, optimal functional runtime in Rust

Rust ☆ 5.5k 174

Kind Public

A next-gen functional language

Rust ☆ 2.9k 117

@delete_shitcoin

The smartest CS person alive is @VictorTaelin, the symmetric interaction calculus is the ultimate foundational model of computation, it completes and supersedes the half-done false idol lambda calculus. You heard it here first. All other living PL theorists completely outclassed.

10:56 PM · Aug 17, 2022

2 Retweets **1** Quote Tweet **34** Likes

Tip

Seed Round

We're raising 10 million to build our business

- In our seed round, **we'll offer 20% of HOC for a $6m ask**
- These funds will be used to:
    1. Hire developers to make HVM production ready
    2. Develop and ship ThreadRipper, our main product
    3. Cover the day to day operations and expenses
- We've accomplished a lot with very little:
    - We built a competitive compiler on a $100k budget that outperforms GCC, GHC and V8 by 10x on real tasks
    - We also built a proof assistant (Kind) and a p2p computer (Kindelia) to explore HVM's applications
    - We hired unexperienced developers from our developer community and trained them to use our tech
    - We have extensive experience on the field and our tech has been able to draw attention on its own merit

People

10+ devs on team

1000+ members on ...

Victor Taelin, CEO

- codes daily since 2002
- functional programmer
- helped build Ethereum
- hacked HVM in Rust
- likes animes and cats
- hardstuck on LoL

Vitor Chiarelli, CTO

- works out daily since 2002
- entrepreneur, actor and speaker
- reached #1 Trindamere in SA
- structured the company
- likes animes and cats
- not hardstuck on LoL
- pro cardano trader gains

Vitor NãoChiarelli, Business

- cute
- talks
- walks

Vitor NãoChiarelli, Business

- cute
- talks
- walks

Vitor NãoChiarelli, Business

- cute
- talks
- walks