

# Haskell in Green Land: Analyzing the Energy Behavior of a Purely Functional Language

Luís Gabriel Lima<sup>1</sup>, Gilberto Melfe<sup>2</sup>, Francisco Soares-Neto<sup>1</sup>,  
Paulo Lieuthier<sup>1</sup>, João Paulo Fernandes<sup>2</sup>, and Fernando Castor<sup>1</sup>



<sup>1</sup> {lgnfl, fmssn, pvjl, castor}@cin.ufpe.br

<sup>2</sup> gilbertomelfe@gmail.com, jpf@di.ubi.pt



# Global energy system is unsustainable



Bloomberg the Company & Its Products | Bloomberg Anywhere Remote Login | Bloomberg Terminal Demo Request

**Bloomberg Business** News Markets Insights Video

Technology

## Inside the Arctic Circle, Where Your Facebook Data Lives

By Ashlee Vance | October 04, 2013

f t in + SEND TO Kindle

A photograph of a reindeer standing in a snowy, forested landscape, symbolizing the Arctic region.

Photograph by Jasper Doest/Foto Natura/Minden Pictures/Corbis

Every year, computing giants including Hewlett-Packard (HPQ), Dell (DELL), and Cisco Systems (CSCO) sell north of \$100 billion in hardware. That's the total for the basic iron—





# Haskell has feelings too!

neurocyte / ghc-android

Watch 32 Star 164

Code Issues 16 Pull requests 1 Wiki Pulse Graphs

Build scripts for building ghc cross compilers targeting Android

76 commits 1 branch 0 releases 6 contributors

Branch: master New pull request New file Upload files Find file SSH git@github.com:neurocyte/ Do

neurocyte Merge pull request #38 from tommyschnabel/master Latest commit 3b6adc

patches	Merge branch 'master' of github.com:joeyhg/ghc-android
.gitignore	Clean-up .gitignore
README.md	update
build	Build both arm and x86 by default
build-arch	Bumped version for automake
mirror	Add build and mirror scripts

Code Android iOS Web Backend Hardware

June 26, 2015 SECURITY · BACKEND

## Fighting spam with Haskell

Simon Marlow

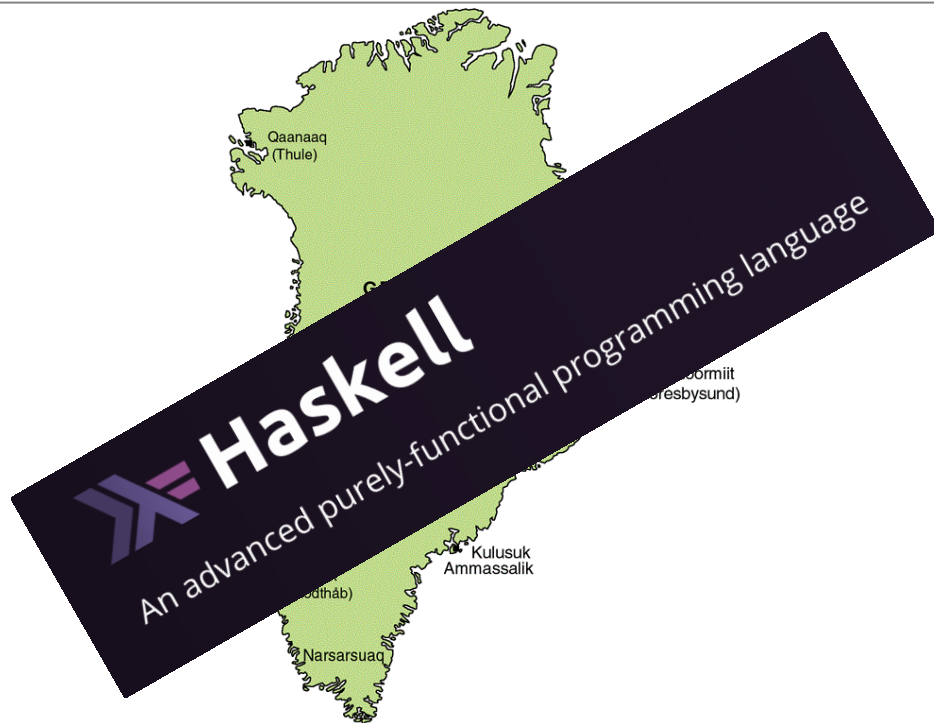
One of our weapons in the fight against spam, malware, and other abuse on Facebook is a system called Sigma. Its job is to proactively identify malicious actions on Facebook, such as spam, phishing attacks, posting links to malware, etc. Bad content detected by Sigma is removed automatically so that it doesn't show up in your News Feed.

We recently completed a two-year-long major redesign of Sigma, which involved replacing the in-house FXL language previously used to program Sigma with Haskell. The Haskell-powered Sigma now runs in production, serving more than one million requests per second.

Haskell isn't a common choice for large production systems like Sigma, and in this post, we'll explain some of the thinking that led to that decision. We also wanted to share the experiences and lessons we learned along the way. We made several improvements to GHC (the Haskell compiler) and fed them back upstream, and we were able to achieve better performance from Haskell compared with the previous implementation.

# The adventures of Haskell in Greenland

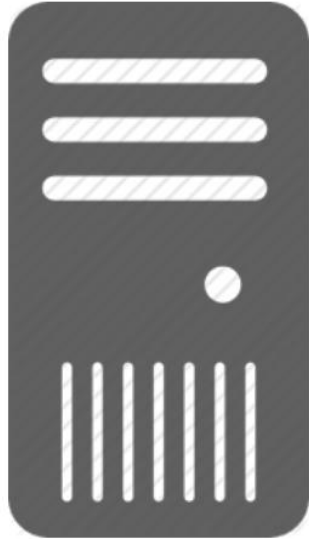
---



To what extent can we save energy by refactoring existing Haskell **programs** to use different **data structure implementations** or **concurrent programming constructs**?

# Experimental Setup

---



2x10-core Intel Xeon E5-2660 v2  
processors (Ivy Bridge) 256GB of DDR3 1600MHz RAM

Criterion

RAPL



# Study 1: Purely functional data structures

**RQ1.** How do **different implementations** of the same **abstractions** compare in terms of **run time** and **energy efficiency**?

**RQ2.** For concrete operations, what is the **relationship** between their **performance** and their **energy consumption**?

# Study 1: Edison Library

---

<b>Collections</b>	<b>Associative Collections</b>	<b>Sequences</b>
EnumSet StandardSet UnbalancedSet LazyPairingHeap LeftistHeap MinHeap SkewHeap SplayHeap	AssocList PatriciaLoMap StandardMap TernaryTrie	BankersQueue SimpleQueue BinaryRandList JoinList RandList BraunSeq FingerSeq ListSeq RevSeq SizedSeq MyersStack

# Study 1: Benchmark

---

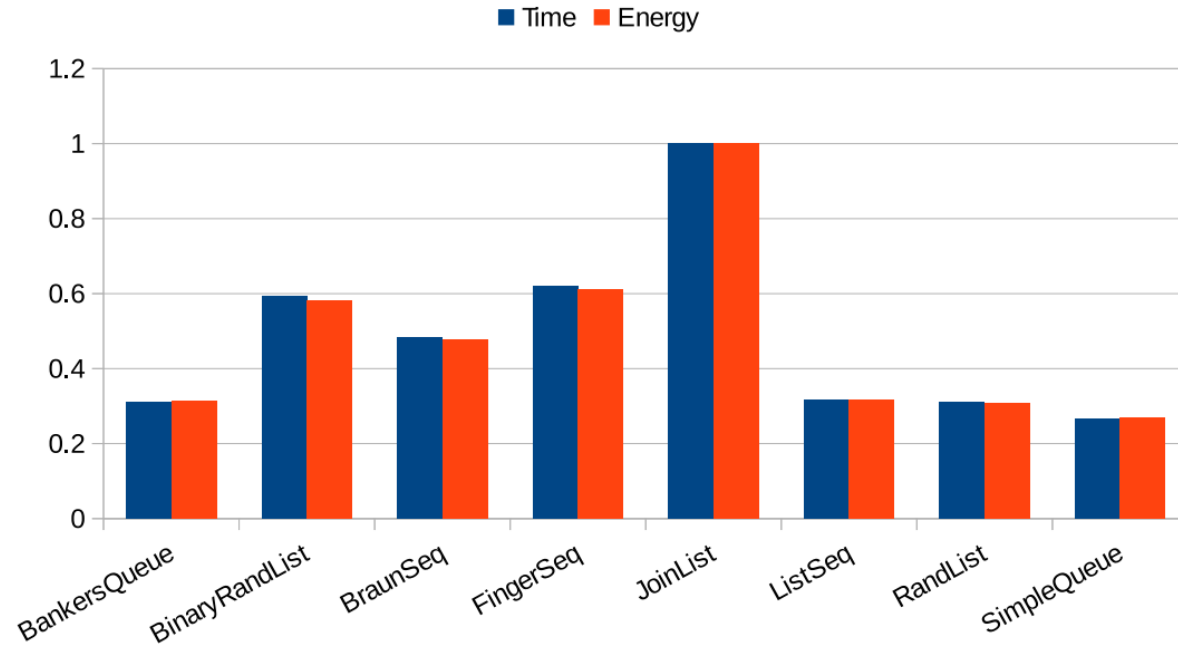
<b>iters</b>	<b>operation</b>	<b>base</b>	<b>aux</b>
1	add	100000	100000
1000	addAll	100000	1000
1	clear	100000	n.a.
1000	contains	100000	1
5000	containsAll	100000	1000
1	iterator	100000	n.a.
10000	remove	100000	1
10	removeAll	100000	1000
5000	toArray	100000	n.a.
10	retainAll	100000	1000

```
    iters = 0;  
while iters < 10  
retainAll base aux;  
iters++;
```

**320 configurations**  
**3000+ executions**

# Study 1: Results

---





**RQ1.** How do **different implementations** of the same **abstractions** compare in terms of **runtime** and **energy efficiency**?

Full details on [green-haskell.github.io](https://github.com/green-haskell/green-haskell).

**RQ2.** For concrete operations, what is the **relationship** between their **performance** and their **energy consumption**?

**Energy** is **proportional** to execution **time**.

# Study 2: Concurrent programming constructs

**RQ1.** Do alternative **thread management** constructs have different impacts on **energy consumption?**

**RQ2. Do alternative data-sharing primitives  
have different impacts on energy  
consumption?**

**Thread management:** `forkIO`, `forkOn`,  
`forkOS`

**Data sharing:** `MVar`, `TVar`, `TMVar`

# The Computer Language Benchmarks Game

9 benchmarks: IO, memory, synchronization bound

Up to 9 variants per benchmark  
9 configurations for # of capabilities

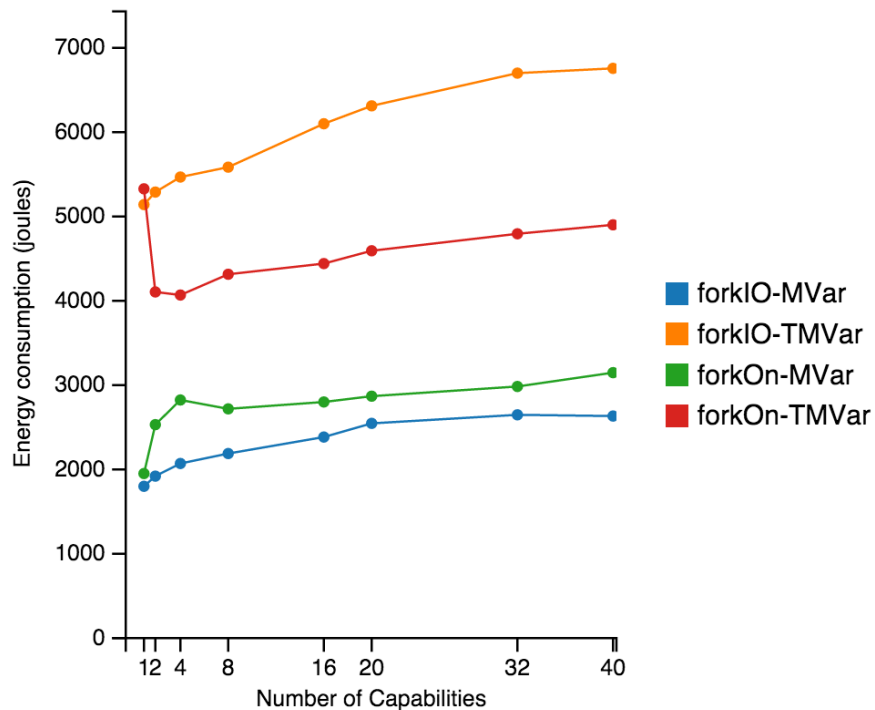
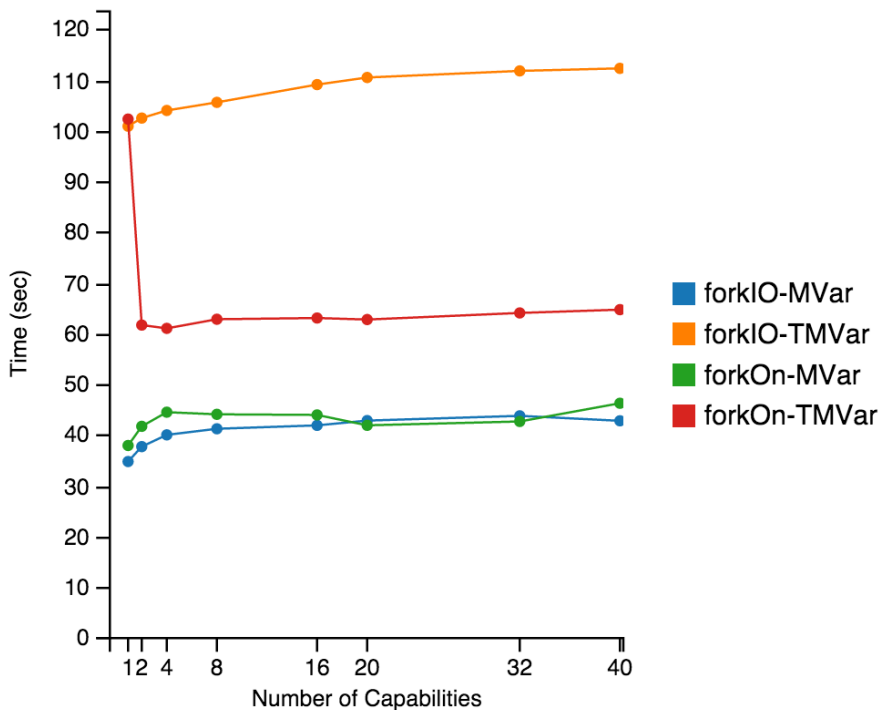


ROSETTACODE.ORG

<http://green-haskell.github.io>

# Small changes can produce big savings

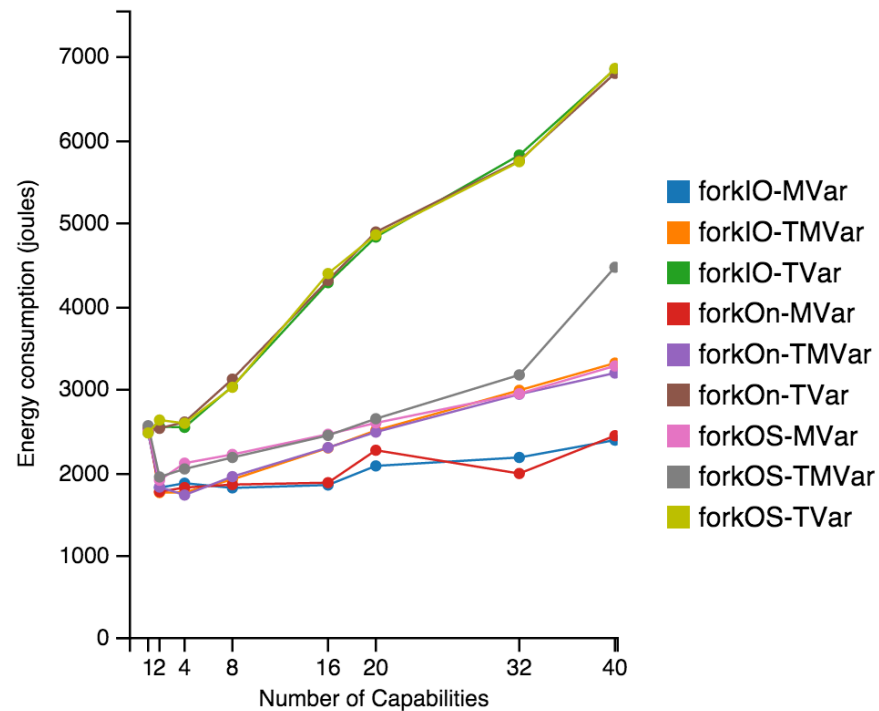
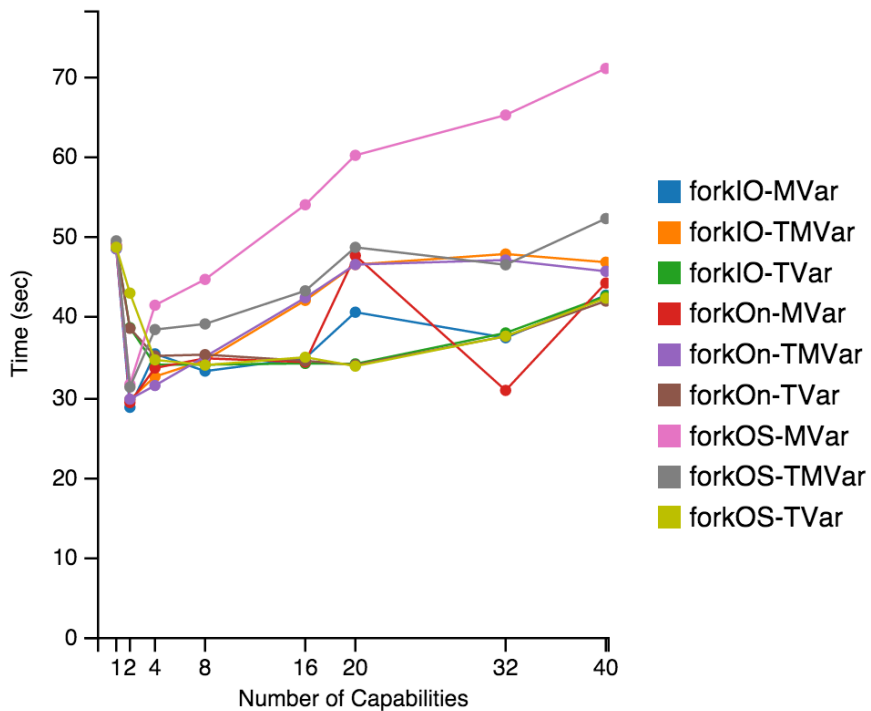
## chameneos-redux





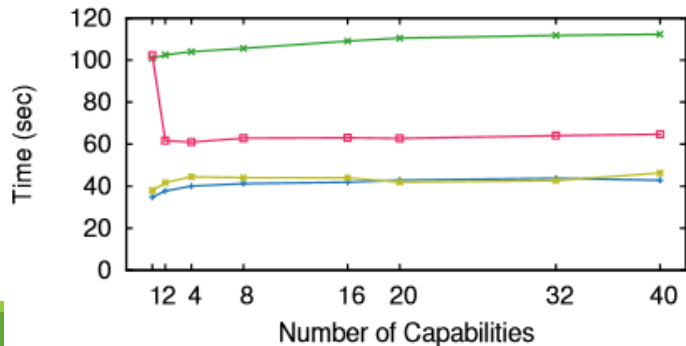
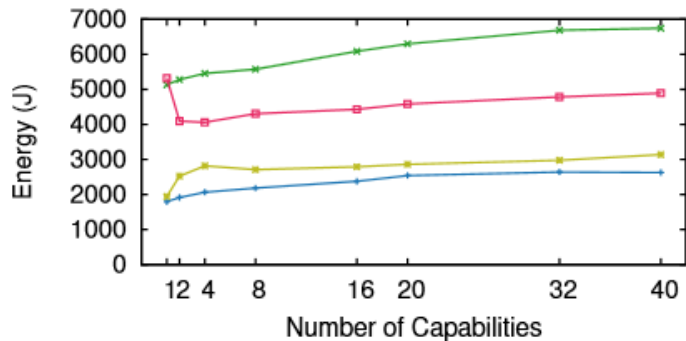
# Faster is not always greener

fasta

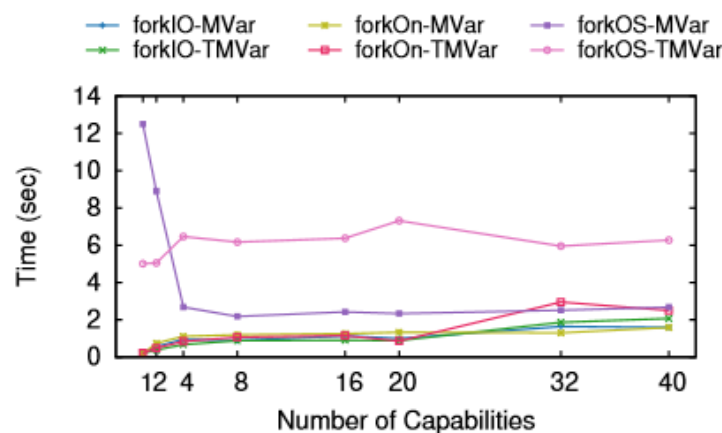
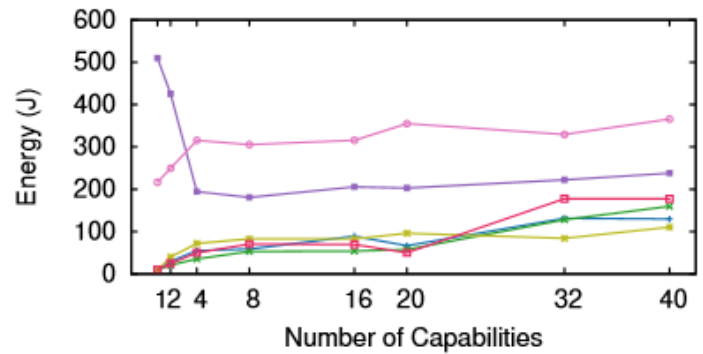


# There is no overall winner

## chameneos-redux



## dining-philosophers





Made two tools **energy-aware**:

GHC profiler

Criterion



Population: 56483 (2013)

Haskell programmers: 0 (est.)



# Haskell has feelings too!

neurocyte / ghc-android

Code Android iOS Web Backend Hardware

Watch 32 Star 144

Build scripts for building ghc cross compilers targeting Android

76 commits 1 branch 6 releases 6 contributors

Branch: master New pull request

neurocyte Merge pull request #38 from tommytschraebelmaster Latest commit 3b6adc

patches Merge branch 'master' of github.com:peyghic/ghc-android

gitignore Clean-up gitignore

README.md update

build Build both arm and x86 by default

build-arch Bumped version for automake

error Add build and error scripts

## Fighting spam with Haskell

One of our weapons in the fight against spam, malware, and other abuse on Facebook is a system called Sigma. Its job is to proactively identify malicious actions on Facebook, such as spam, phishing attacks, posting links to malware, etc. Bad content detected by Sigma is removed automatically so that it doesn't show up in your News Feed.

We recently completed a two-year-long major redesign of Sigma, which involved replacing the in-house FXL language previously used to program Sigma with Haskell. The Haskell-powered Sigma now runs in production, serving more than one million requests per second.

Haskell isn't a common choice for large production systems like Sigma, and in this post, we'll explain some of the thinking that led to that decision. We also wanted to share the experiences and lessons we learned along the way. We made several improvements to GHC (the Haskell compiler) and fed them back upstream, and we were able to achieve better performance from Haskell compared with the previous implementation.

# Haskell has feelings too!

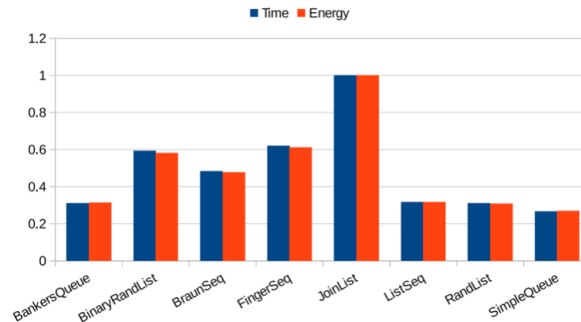
The screenshot shows a GitHub pull request for the repository 'neurocyte / ghc-android'. The pull request is titled 'Fighting spam with Haskell' and is authored by 'Simon Marlow'. The pull request description includes the following text:

One of our weapons in the fight against spam, malware, and other abuse on Facebook is a system called Sigma. Its job is to proactively identify malicious actions on Facebook, such as spam, phishing attacks, posting links to malware, etc. Bad content detected by Sigma is removed automatically so that it doesn't show up in your News Feed.

We recently completed a two-year-long major redesign of Sigma, which involved replacing the in-house FXL language previously used to program Sigma with Haskell. The Haskell-powered Sigma now runs in production, serving more than one million requests per second.

Haskell isn't a common choice for large production systems like Sigma, and in this post, we'll explain some of the thinking that led to that decision. We also wanted to share the experiences and lessons we learned along the way. We made several improvements to GHC (the Haskell compiler) and fed them back upstream, and we were able to achieve better performance from Haskell compared with the previous implementation.

## Study 1: Results

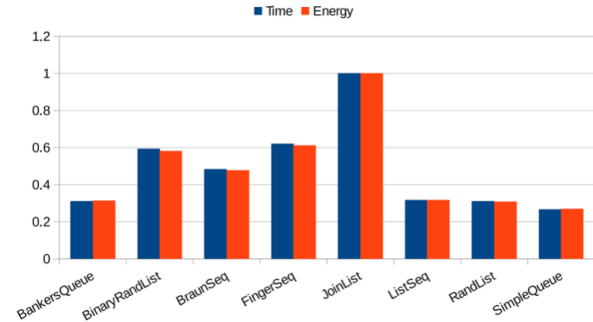




# Haskell has feelings too!

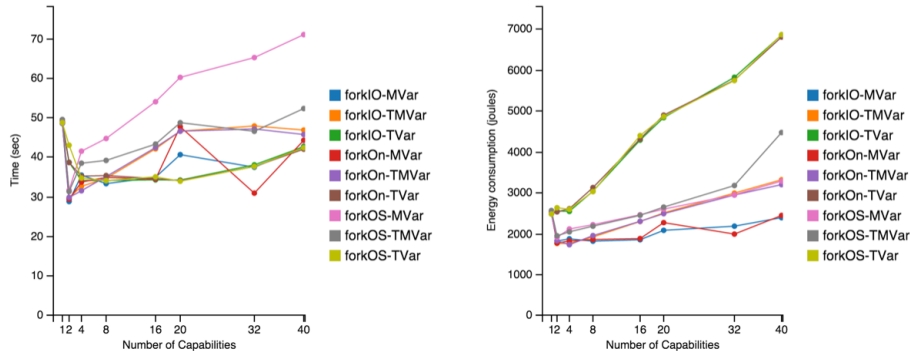
The screenshot shows a GitHub pull request for the repository `neurocyte/ghc-android`. The pull request is titled "Fighting spam with Haskell" and is authored by `neurocyte`. It includes a list of patches such as `patches`, `gpgignore`, `README.md`, `build`, `build-arch`, and `error`. The pull request description discusses the implementation of Sigma, a system for identifying malicious actions on Facebook, and mentions that Haskell was used for the implementation. It also notes that Haskell is not a common choice for large production systems like Sigma.

# Study 1: Results



# Faster is not always greener

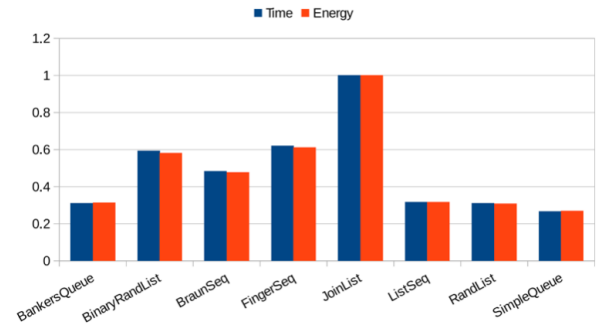
## fasta



# Haskell has feelings too!

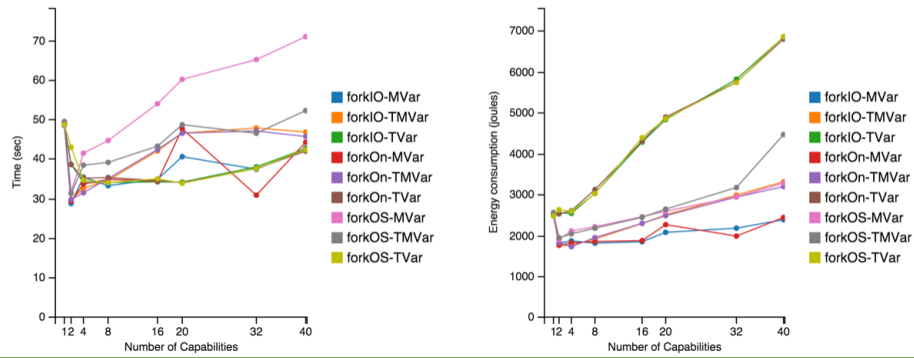
# Study 1: Results

The screenshot shows a GitHub pull request for the repository 'neurocyte / ghc-android'. The pull request is titled 'Fighting spam with Haskell' and is authored by 'neurocyte'. It includes a commit message: 'Merge pull request #38 from tommytschabee/master'. The pull request description states: 'One of our weapons in the fight against spam, malware, and other abuse on Facebook is a system called Sigma. Its job is to proactively identify malicious actions on Facebook, such as spam, phishing attacks, posting links to malware, etc. Bad content detected by Sigma is removed automatically so that it doesn't show up in your News Feed. We recently completed a two-year-long major redesign of Sigma, which involved replacing the in-house FXL language previously used to program Sigma with Haskell. The Haskell-powered Sigma now runs in production, serving more than one million requests per second. Haskell isn't a common choice for large production systems like Sigma, and in this post, we'll explain some of the thinking that led to that decision. We also wanted to share the experiences and lessons we learned along the way. We made several improvements to GHC (the Haskell compiler) and fed them back upstream, and we were able to achieve better performance from Haskell compared with the previous implementation.'



## Faster is not always greener

### fasta



## Small changes can produce big savings

### chameneos-redux

