Programming Languages

Harley Eades

“We need to do away with the myth that computer science is about computers. Computer science is no more about computers than astronomy is about telescopes, biology is about microscopes or chemistry is about beakers and test tubes. Science is not about tools, it is about how we use them and what we find out when we do.” –Michael R. Fellows

“Controlled side effects is the many ingredient of a safe and useful PL.” - Simon Peyton Jones

What is this course about?

haskell data scientist

apply now

job id: bus000143
date posted: 04/24/2017
location: sunnyvale, california – united states

Description:
At Target, we are surrounded by a wide range of interesting problems: Supply-Chain Optimization, Inventory Control, Merchandise Selection, Space Design, Demand Prediction, Price Optimization, to name just a few. Historically, the retail industry has thrown Operations Research, Econometric Modeling and Database Design techniques at these problems, with moderate success. However, at Target, we see an opportunity to boost revenues and reduce costs of the order of billions of dollars by merging these traditional techniques with modern and sometimes unusual methods - Machine Learning, Distributed Computing, Functional Programming, Compilers for Domain-Specific Languages, Graph Theory, Algorithms on Algebraic Datatypes, Stochastic Modeling, Control Theory, Differential Equations. Our problems are challenging and we have to dig deep into each of the areas above to penetrate our problems. We feel fortunate to be able to explore so much breadth as well as depth in the structure of our technical work.

To top it all, our leadership consists of Ph.D.s in Operations Research, Machine Learning, Algebra, Algorithms and Astrophysics. Our leaders are obsessed with hard Math problems and dream in Haskell. Our team is not only technically strong but is also passionate about solving interesting practical problems in our retail business.

If you are an experienced Haskell Programmer with a good understanding of Category Theory (can do magic with Haskell's Typeclasses) and have a flair for developing Algorithms, this will be your dream job.

YouTube

Search

Haskell at Target

Tikhon Jelvis

tikhon.jelvis@target.com

0:01 / 41:52

Tikhon Jelvis: Haskell at Target

Bay Area Haskell

Subscribe 257

1,675 views

Add to Share More

16 0

dire

The Caml Language

caml.inria.fr/

Home About Download Resources Contact us Version française

This site is updated infrequently. For up-to-date information, please visit the new OCaml website at ocaml.org.

News Archives

The Caml Language

2017/06
OCaml 4.04.2 released

2017/04
OCaml 4.04.1 released

2017/01
Docker joins the Caml Consortium

2016/11
OCaml 4.04.0 released

2016/08
Kernelyze LLC joins the Caml Consortium

2016/05
Ahrefs joins the Caml Consortium

2016/04
OCaml 4.03.0 released

2016/04
Facebook joins the Caml Consortium

2015/07
OCaml 4.02.3 released

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.

How does Sigma work?

Sigma is a *rule engine*, which means it runs a set of rules, called policies. Every interaction on Facebook — from posting a status update to clicking "like" — results in Sigma evaluating a set of

Related

Open-sourcing Haxl, a library for Haskell

Fighting spam with pure functions

Swift 4

The powerful programming language that is also easy to learn.

Swift is a powerful and intuitive programming language for macOS, iOS, watchOS and tvOS. Writing Swift code is interactive and fun, the syntax is concise yet expressive, and Swift includes modern features developers love. Swift code is safe by design, yet also produces software that runs lightning-fast.

Minds making markets

With offices in **New York**, **London**, and **Hong Kong**, Jane Street is a trading firm that operates around the clock and around the globe, trading a wide range of financial products, including:

Experience Report: Haskell as a Reagent

Results and Observations on the Use of Haskell in a Python Project

Iustin Pop
Google Switzerland
iustin@google.com

Abstract

In system administration, the languages of choice for solving automation tasks are scripting languages, owing to their flexibility, extensive library support and quick development cycle. Functional programming is more likely to be found in software development teams and the academic world.

This separation means that system administrators cannot use the most effective tool for a given problem; in an ideal world, we should be able to mix and match different languages, based on the problem at hand.

This experience report details our initial introduction and use of Haskell in a mature, medium size project implemented in Python. We also analyse the interaction between the two languages, and show how Haskell has excelled at solving a particular type of real-world problems.

Categories and Subject Descriptors D.2.12 [*Software engineering*]: Interoperability; D.3.2 [*Programming languages*]: Language Classifications—Applicative (functional) languages

General Terms Experimentation, Languages

Keywords Haskell, Python, Ganeti, System administration

type system contrasts markedly from Python's "laissez-faire" approach to types, and its cheap-persistence model is the opposite of Python's cheap-modification one, while both are in the same high-level language category where complex data-modification pipelines are readily available. Such diametrically opposite views on some topics were very good at highlighting differences, and neither language was delegated to the 'low-level' versus 'high-level' status.

We have observed gains from simply having two different languages exercise the same API/RPC endpoints; in our case, this meant that the effort spent to standardise the message types (useful for Haskell) can lead to a more sound framework on the Python side. Prototyping the same algorithm in both languages led to a better understanding and in a few cases optimisations in one language can be carried to the other side.

Not all was good, however; a few bumps appeared along the way, in the form of small issues with availability of libraries, performance for some operations, compatibility between different versions of the base libraries and the higher difficulty of advanced programming techniques in a functional language.

1.1 Our contribution

Past ICFP experience reports have focused on either conversion of software from an imperative language ([Newton and Ko 2009]), or

Qualification of a Model Checker for Avionics Software Verification

Lucas G. Wagner
Alain Mebsout
Cesare Tinelli
Darren D. Cofer
Konrad L. Slind

Simon Peyton Jones

Principal Researcher

Contact Info

+44 1223 479 848

Email

21 Station Road
Cambridge, CB1 2FB
United Kingdom

Groups

Programming Principles and Tools

About Publications Videos Pictures Awards & Honours Biography

I'm a researcher at Microsoft Research in Cambridge, England. I started here in Sept 1998. I'm also an Honorary Professor of the Computing Science Department at Glasgow University, where I was a professor during 1990-1998.

I am married to Dorothy, a priest in the Church of England. We have six children.

I'm interested in the design, implementation, and application of lazy functional languages. In practical terms, that means I spend a most of my time on the design and implementation of the language Haskell. In particular, much of my work is focused around the Glasgow Haskell Compiler, and its ramifications.

I am chair of Computing at School, the group at the epicentre of the reform of the national curriculum for Computing in England. Computer science is now a foundational subject, alongside maths and natural science, that every child learns from primary school onwards (background here).

Research skills +

Useful information and links +

ABOUT US

Building the compute for the next generation of high performance machine learning.

Groq delivers industry leading performance and sub-millisecond latency with efficient, software-driven solutions for compute-intensive applications. We're redefining compute by focusing on key technology innovations: software-defined compute, silicon innovation and developer velocity.

LEARN MORE

PERFORMANCE

Companies that use Haskell in production as

- IP - internal product
- PR - software product
- RD - research activities
- CO - consulting activities

table is sortable, default by numer

#	Name	Type	Location	Area
1	Microsoft	RD	UK, Cambridge	Compilers
2	Facebook	IP	USA, CA, Menlo Park	Advertising, Spam Filtering
3	Barclays Capital	IP	UK, London	Finance
4	Standard Chartered Bank	IP	UK, London	Finance
5	Tsuru Capital	IP	Japan, Tokyo	Finance
6	Galois	RD	USA, OR, Portland	Consulting, Research
7	FP Complete	CO	USA, CA, San Diego	Consulting
8	IOHK	RD	Hong Kong, Hong Kong	Blockchain
9	Tesla	IP	USA, CA, Palo Alto	Hardware
10	Skedge	PR	USA, NY, NYC	Productivity, SaaS

Graded Work

- (40%) 4 graded homework assignments.
- (30%) A take-home midterm exam.
- (30%) A final project: Implementation of a Domain Specific Language

No scores will be curved throughout the semester. Please use the following scoring tables:

- A : $88\% - 100\%$
- B : $78\% - 87\%$
- C : $68\% - 77\%$
- D : $58\% - 67\%$
- F : $0\% - 57\%$

Turing in Homework

- All homework will be released and turned in via a private Github repo.

- Please sign up for Github and send me a DM on slack with your username.

All homework is to be turned in via your private Git repository. There are two deadlines for this course:

- Deadline: By 11:59pm on the Friday after the homework is released.
- Hard Deadline: By 11:59pm on the Wednesday fives days after the deadline.

The precise dates will be on each homework assignment.

Getting Help

Office hours are by appointment.

Ask questions online using Slack in #office_hours.

All course discussion must go through Slack unless it is grade specific, then use my university email or ask via Teams, or after class.

Collaboration is important in all areas of study.

So I recommend talking to other students for help.

However, there is a rule that must be followed: **while discussing homework solutions in a group of two or more no one is allowed to write anything down, and <u>each student must prepare their solution in isolation.</u>**

Attendance

Attendance is optional, but highly encouraged. If you do not come to class then there is a high probability that you will fail this course.

The Project
Design and Implementation of a Domain Specific Language

- You get to choose the language you want to implement.
- Groups of 1 or 2 students.
- Project Meeting: Each project and group will have to be approved during the project meeting.
- Midpoint Report: At the halfway point of the project a report detailing the projects progress is due.
- Final Exam: Project presentations!

What are Domain Specific Languages?

A programming language specifically designed to make solving problems within a particular domain easier.

Example 1: Regular Expressions

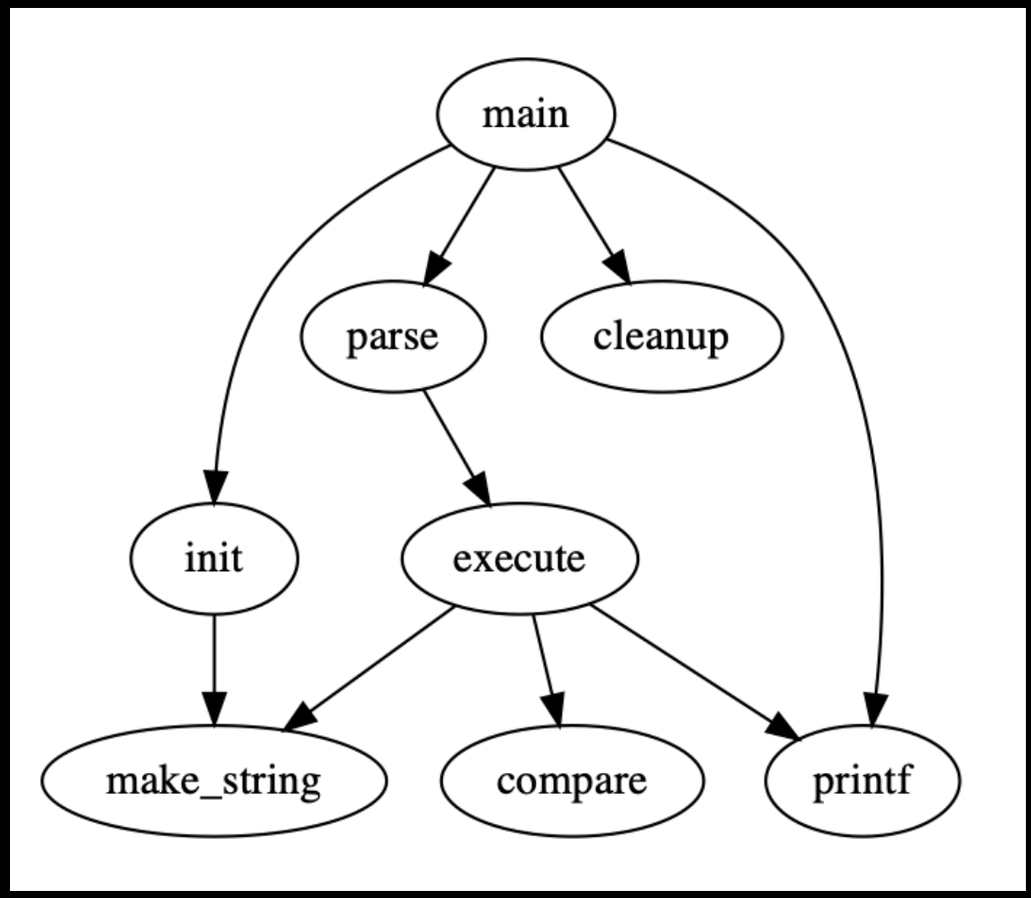
```
The following pattern matches email addresses:

^([a-zA-Z0-9_\-\.]+)@([a-zA-Z0-9_\-\.]+)\.([a-zA-Z]{2,5})$
```

Implementation:

Regular expressions compile to deterministic finite automata. That are used to conduct the matching.

Example 2: The Dot Language (Graphviz)

```
1: digraph G {
2:         main -> parse -> execute;
3:         main -> init;
4:         main -> cleanup;
5:         execute -> make_string;
6:         execute -> printf
7:         init -> make_string;
8:         main -> printf;
9:         execute -> compare;
10: }
```



Implementation:

Converts the input code to various output formats like PDF, Png, LaTeX, etc.

Example 3: Verilog

```
module toplevel(clock,reset);
  input clock;
  input reset;
  reg flop1;
  reg flop2;

  always @ (posedge reset or posedge clock)
    if (reset)
      begin
        flop1 <= 0;
        flop2 <= 1;
      end
    else
      begin
        flop1 <= flop2;
        flop2 <= flop1;
      end
endmodule
```

Implementation:

Structure and behavior are verified by a compiler, and then realized as an actual blueprint of an electronic circuit.