Advances in Programming Languages

APL4: Coursework Assignment

Ian Stark

School of Informatics
The University of Edinburgh

Monday 21 January 2008
Semester 2 Week 3

Outline

Outline

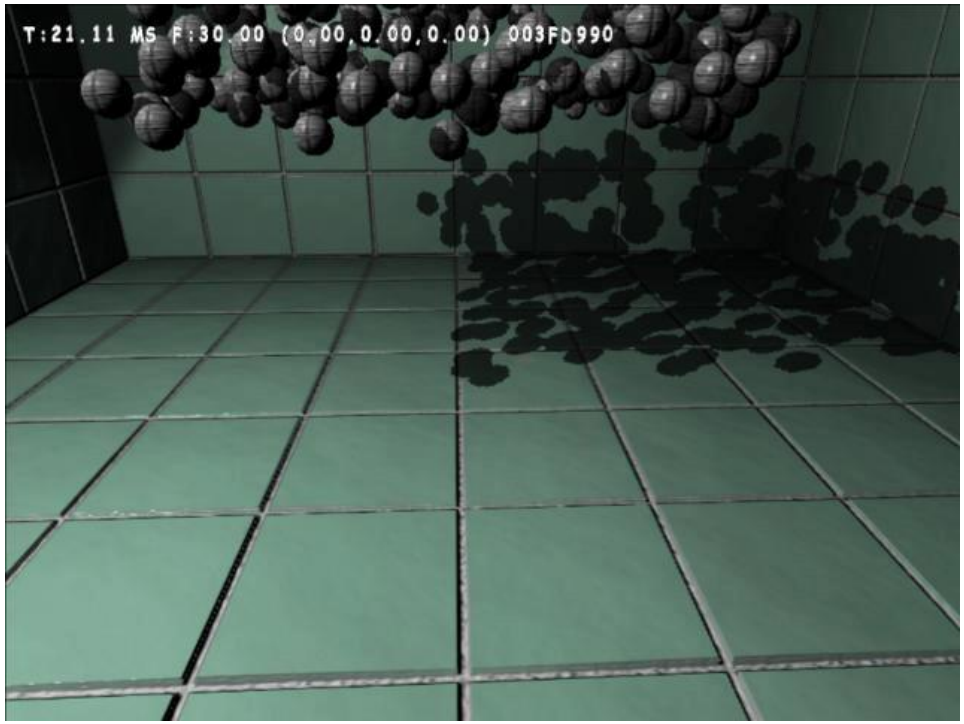
Quadtree example?

Video by Handkor

`http://handkor.googlepages.com/`

Early test sequence for "Hippocrates's Dilemma"

Octree example



Video by Handkor

`http://handkor.googlepages.com/`

Collision test — 256 marbles & gravity

Aims of exercises

The aims of the homework exercises set in lectures include:

- To review lecture material
- To give some context for understanding the lectures
- To provide other sources and views on the lecture topic
- To help with learning by exploring the subject

Crucially, these effects arise from doing the exercises. They are not assessed, nor would this necessarily be useful: they are intended to be self-validating (i.e. you can tell when you have succeeded)

Although your coursework reports will be assessed, most of this also applies there: the purpose is for you to find out and learn new things.

Outline

Deputy

Memory annotations in Deputy

The *Deputy* project at Berkeley is developing a C compiler that can prevent a number of common programming errors.

In particular, Deputy provides type annotations with which programmers can describe the intended behaviour of pointers. The compiler will then apply suitable static and run-time checks to make sure these intentions are satisfied.

CDuce

Regular expression types and patterns in CDuce

The *CDuce* programming language is designed for processing XML. It is an ML-style functional language, with special features for reading, transforming, and creating XML trees.

Notably, it uses *regular expressions* in both types and patterns to describe code that queries and manipulates XML trees, allowing the compiler to statically check that programs only ever generate valid XML.

Haskell STM library

Software Transactional Memory in Haskell

The *STM* library for the Glasgow Haskell Compiler (GHC) provides high-level language support for coordinating concurrent computation, where multiple threads act simultaneously on shared datastructures.

Remarkably, STM does this without using locks. Instead, it uses efficient and optimistic *software transactions*, giving freedom from deadlock and promoting non-interfering concurrency. These transactions are modular and composable: small transactions can be glued together to make larger ones. Moreover, implementing this within the Haskell type system gives static guarantees that transactions are used correctly.

Futures and promises

Futures and promises in Alice ML

The *Alice ML* language is based on Standard ML, with several extensions to support distributed concurrent programming.

In particular it provides *futures* and *promises* for lightweight concurrency: a future represents the result of a computation that may not yet be available, and a promise is a handle to build your own future.

Information flow in Jif

The *Jif* compiler extends the Java language with annotations for static analysis of security properties relating to the flow of information.

These annotations describe restrictions on how information is to be used: which *principals* control which information, and what they trust other principals to do with it. This gives increased assurance that trusted and untrusted information is used only according to explicit security policies.

Outline

Dates and submission

Week 3 Monday 21 January: Topic announcement

Week 4 Friday 1 February: Choose topic

Submit a file `choice.txt` containing the following:

- Which topic you have chosen
- Three suitable references

One reference must be to a published paper; the other two may be too, but could also be white papers, web tutorials, manuals, or similar. In all cases provide enough information for someone else to obtain the document.

Week 9 Friday 7 March: Report due

Submit a file `apl.pdf` containing your report in PDF. The recommended method for creating this is pdflatex with the article document class.

In addition, OpenOffice is freely available for Windows and Linux, installed on Informatics machines, and can write PDF. Mac OS X natively creates PDF. Microsoft provide PDF output as a plugin for Word 2007.

Suggested outline

Heading	Title, date, author
Abstract	This report describes ...
Introduction	Content summary, overview of report structure
Context	The problem domain
⟨Main topic⟩	What it is, how it works, advantages and limitations
Example	Annotated code, explanation, screenshot
	Salt: the example must in some way concern travel or transport (e.g. bicycle shop bills, tracking trucks, . . .)
Resources	For notable resources used (article, tutorial, manual), give a summary in your own words of what it contains
Related work	Other approaches to the problem
Conclusion	What ⟨topic⟩ does, good and bad points
Bibliography	Full references for all resources used

Total 8–10 A4 pages. See course web pages for further details.

Outline

Plagiarism

University of Edinburgh
Undergraduate Assessment Regulations 2007/08

Regulation 14

14.1 Plagiarism is the act of copying or including in one's own work, without adequate acknowledgement, intentionally or unintentionally, the work of another.

`http://www.acaffairs.ed.ac.uk/Regulations/Assessment/07-08/UG.htm#Reg14`

See also:

- University guidance
  `http://www.aaps.ed.ac.uk/regulations/Plagiarism/Intro.htm`
- Informatics policy
  `http://www.inf.ed.ac.uk/teaching/plagiarism.html`

Suitable working practices

Working practices

- Start with a blank document; all the words must be yours.
- Do not cut and paste from other documents.
  - Except for direct quotations, which must have source declared.
- Do not let others read your text; nor read theirs.

Aims of this coursework

- To learn about the chosen topic
- To improve researching and learning skills
- To demonstrate said knowledge and skills

The tangible outcome is a document, composed and written by you, demonstrating what you have learnt.

Outline

Summary

Topic choices

- Memory annotations in Deputy
- Regular expression types and patterns in CDuce
- Software Transactional Memory in Haskell
- Futures and promises in Alice ML
- Information flow in Jif

Coursework and learning

- Lecture exercises are there to be done
- Note the essay plan
- All your own work
- The aims of the coursework are to support learning