Engaging, Large-Scale Functional Programming Education in Physical and Virtual Space

Kevin Kappelmann, Jonas Rädle, Lukas Stevens

March 16, 2022

Technical University of Munich

Supported by

- Lecturer: Tobias Nipkow
- Manuel Eberl
- 2019: 13 student assistants
- 2020: 22 student assistants

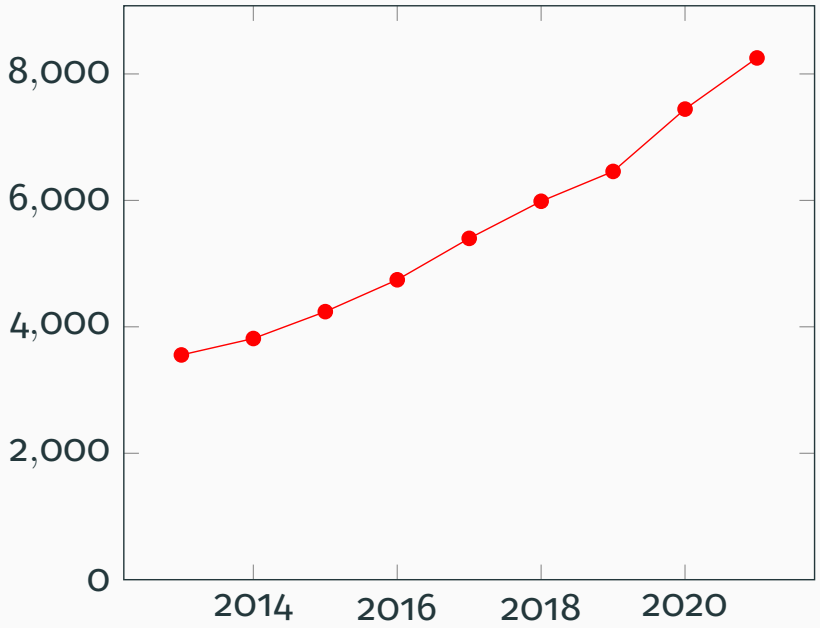
Challenges

Soaring Enrolments

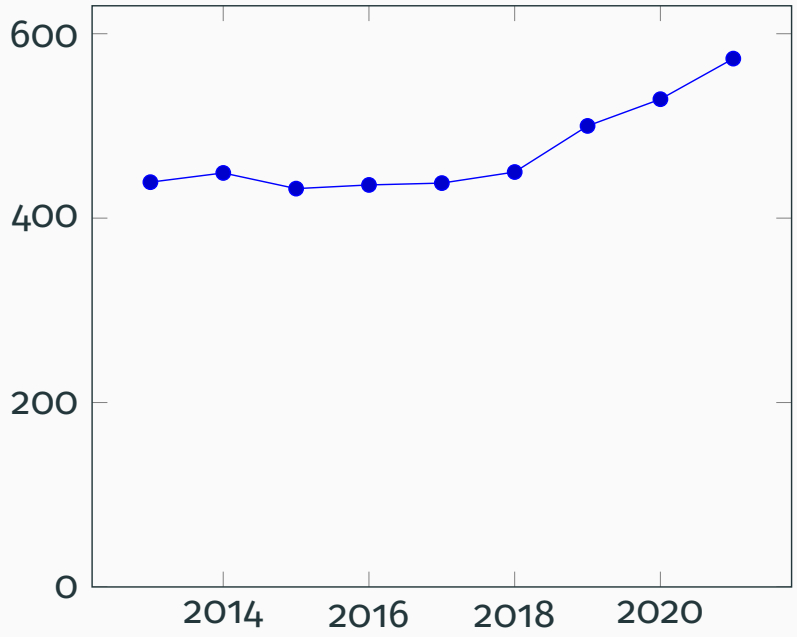
1. Number of Computer Science students exploded

Soaring Enrolments

Example: Computer Science at TU Munich



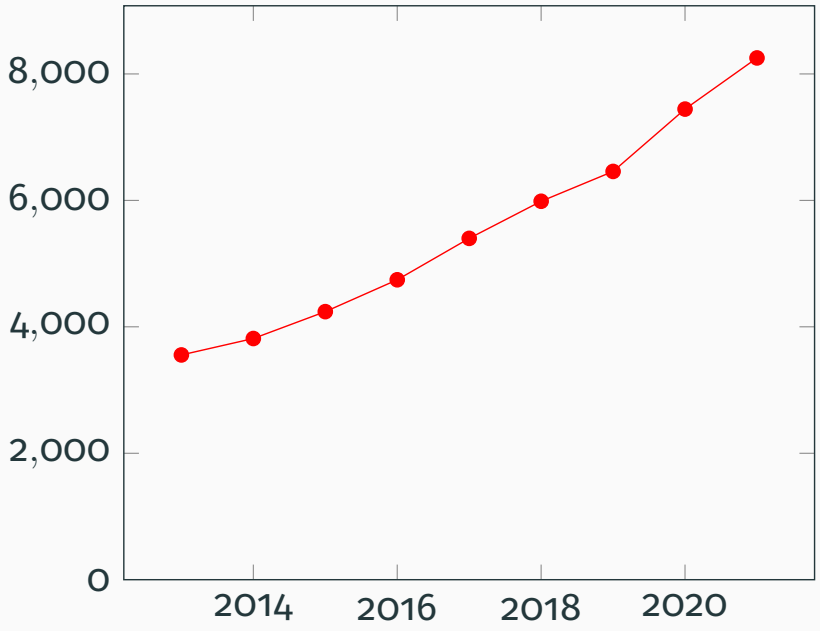
Number of CS students
(132% increase)



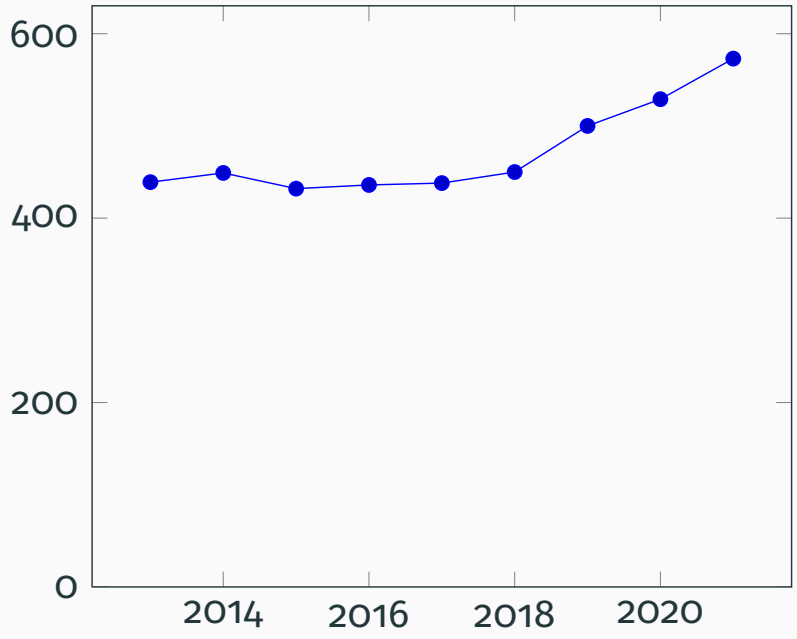
Number of CS academic staff
(31% increase)

Soaring Enrolments

Example: Computer Science at TU Munich



Number of CS students
(132% increase)



Number of CS academic staff
(31% increase)

1000+ students per course are the new normal

The Pandemic

2. Radical transition to online classes

The Pandemic

How can we go from here...

The Pandemic

to here...

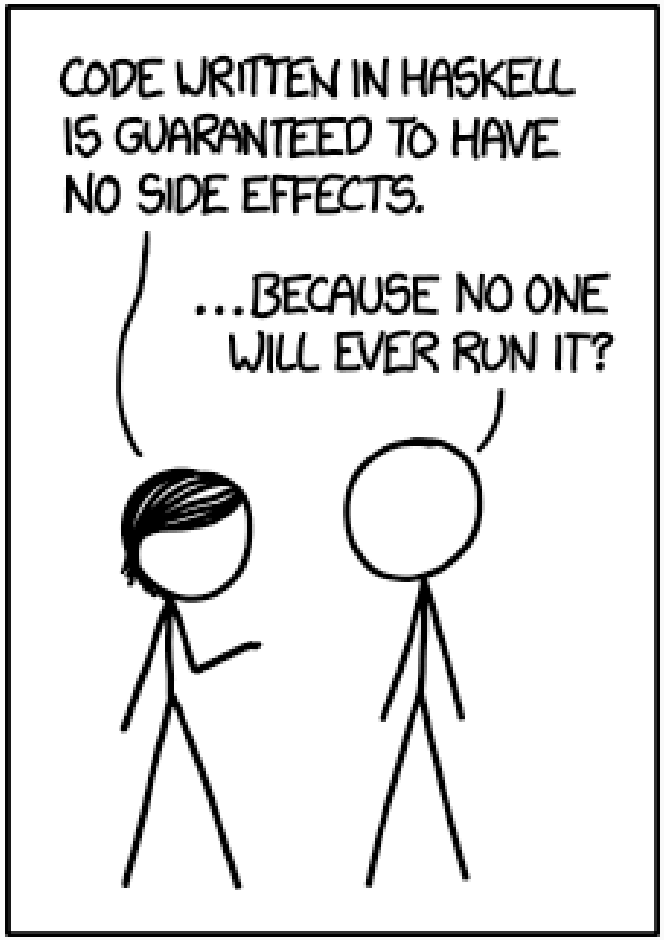
The Pandemic

without ending up here?

Usefulness of Functional Programming

3. Students question the usefulness of functional languages beyond academia

Usefulness of Functional Programming



xkcd.com/1312

xkcd.com/1270

There is hope!

- We managed to cope with all these challenges

There is hope!

- We managed to cope with all these challenges
- We share our insights, tools, and exercises for other educators

You can find our resources on:

thub.com/kappelmann/engaging-large-scale-functional-programmi

There is hope!

- We managed to cope with all these challenges
- We share our insights, tools, and exercises for other educators

You can find our resources on:

thub.com/kappelmann/engaging-large-scale-functional-programmi

Note: We used Haskell, but most ideas apply to any functional programming course

Practical Part

Practical Part

Engagement Mechanisms

Grade Bonus

Incentives do work!

Grade Bonus

Incentives do work!

- Bonus of one grade step on final exam if certain goals are achieved

Grade Bonus

Incentives do work!

- Bonus of one grade step on final exam if certain goals are achieved
- Offer multiple ways to obtain points: homework exercises, participation in workshops, programming contests,...

Grade Bonus

Incentives do work!

- Bonus of one grade step on final exam if certain goals are achieved
- Offer multiple ways to obtain points: homework exercises, participation in workshops, programming contests,...
- Was once abolished...

Grade Bonus

Incentives do work!

- Bonus of one grade step on final exam if certain goals are achieved
- Offer multiple ways to obtain points: homework exercises, participation in workshops, programming contests,...
- Was once abolished...

  Result: number of homework submissions severely decreased

Instant Feedback

Feedback must come fast!

Instant Feedback

Feedback must come fast!

- Asynchronous Q&A forum

Instant Feedback

Feedback must come fast!

- Asynchronous Q&A forum
- Automated testing and feedback

Instant Feedback

Feedback must come fast!

- Asynchronous Q&A forum
- Automated testing and feedback
  - *ArTEMiS* runs tests, manages scores, offers exam mode,...

Instant Feedback

Feedback must come fast!

- Asynchronous Q&A forum
- Automated testing and feedback
  - *ArTEMiS* runs tests, manages scores, offers exam mode,...
  - *Tasty* combines QuickCheck, SmallCheck, and HUnit tests

Instant Feedback

Feedback must come fast!

- Asynchronous Q&A forum
- Automated testing and feedback
  - *ArTEMiS* runs tests, manages scores, offers exam mode,…
  - *Tasty* combines QuickCheck, SmallCheck, and HUnit tests
  - *Check Your Proof* for automated proof checking

Instant Feedback

Feedback must come fast!

- Asynchronous Q&A forum
- Automated testing and feedback
  - *ArTEMiS* runs tests, manages scores, offers exam mode,...
  - *Tasty* combines QuickCheck, SmallCheck, and HUnit tests
  - *Check Your Proof* for automated proof checking
- Manual reviews turned out to be inefficient...

Instant Feedback

Feedback must come fast!

- Asynchronous Q&A forum
- Automated testing and feedback
  - *ArTEMiS* runs tests, manages scores, offers exam mode,...
  - *Tasty* combines QuickCheck, SmallCheck, and HUnit tests
  - *Check Your Proof* for automated proof checking
- Manual reviews turned out to be inefficient...
  - *HLint* offers feedback more directly

Instant Feedback

Feedback must come fast!

- Asynchronous Q&A forum
- Automated testing and feedback
  - *ArTEMiS* runs tests, manages scores, offers exam mode,...
  - *Tasty* combines QuickCheck, SmallCheck, and HUnit tests
  - *Check Your Proof* for automated proof checking
- Manual reviews turned out to be inefficient...
  - *HLint* offers feedback more directly
  - Student assistants create engaging exercises instead

Workshops With Industry Partners

Functional programming is practical!

Workshops With Industry Partners

Functional programming is practical!

- In 2020, we hosted 3 workshops with industry partners

Workshops With Industry Partners

Functional programming is practical!

- In 2020, we hosted 3 workshops with industry partners
  1. Design patterns for functional programs
  2. Networking and advanced IO
  3. User interfaces and functional reactive programming

Workshops With Industry Partners

Functional programming is practical!

- In 2020, we hosted 3 workshops with industry partners
  1. Design patterns for functional programs
  2. Networking and advanced IO
  3. User interfaces and functional reactive programming
- Great success: 120 registrations for 105 spots

Workshops With Industry Partners

Functional programming is practical!

- In 2020, we hosted 3 workshops with industry partners
  1. Design patterns for functional programs
  2. Networking and advanced IO
  3. User interfaces and functional reactive programming
- Great success: 120 registrations for 105 spots
- In some cases, workshop extended for multiple hours

Workshops With Industry Partners

Functional programming is practical!

- In 2020, we hosted 3 workshops with industry partners
  1. Design patterns for functional programs
  2. Networking and advanced IO
  3. User interfaces and functional reactive programming
- Great success: 120 registrations for 105 spots
- In some cases, workshop extended for multiple hours
- Little organisational work

Social Interactions

The social environment matters!

Social Interactions

The social environment matters!

- Online courses are isolating...

Social Interactions

The social environment matters!

- Online courses are isolating...
  so let us foster social interaction:

Social Interactions

The social environment matters!

- Online courses are isolating...

  so let us foster social interaction:

  - Pair-programming in tutorials
  - ACM-ICPC-like programming contest
  - Get-together hangout sessions
  - Award ceremonies

Competitions

Offer challenges to go beyond the syllabus!

Competitions

Offer challenges to go beyond the syllabus!

- Diverse, weekly competition exercises

Competitions

Code Golf

```
numberToEo = unwords <<< catMaybes <<< reverse <<< flip lookup `zipWith` table <<< chunksOf 3
  <<< pack <<< reverse

topCycle = concatMap snd . ap zip (iterate . fmap nub . concatMap . dominators <*> copeland)

traceFractran rs n = n : fromMaybe [] (traceFractran rs <$> numerator <$>
  (liftM2 eqInteger truncate numerator) `find` map (fromIntegral n *) rs)

bernoulli = genericIndex $ map head $ iterate ((*) `zipWith` enumFrom 1 <<< zipWith subtract
  $ recip `map` enumFrom 1
```

Competitions

Optimization

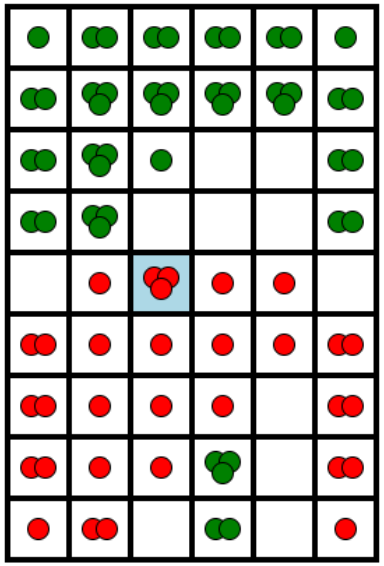
Rank	Name	Time [seconds]
1.	MC Jr	0.02
2.	Florian Hübler	0.13
2.	Luis Bahners	0.19
4.	Tobias Markus	0.35
4.	Robert Imschweiler	0.37
6.	Julian Pritzi	0.76

Competitions

Strategy

Tobias Markus vs. Severin Schmidmeier

Winner: Severin Schmidmeier

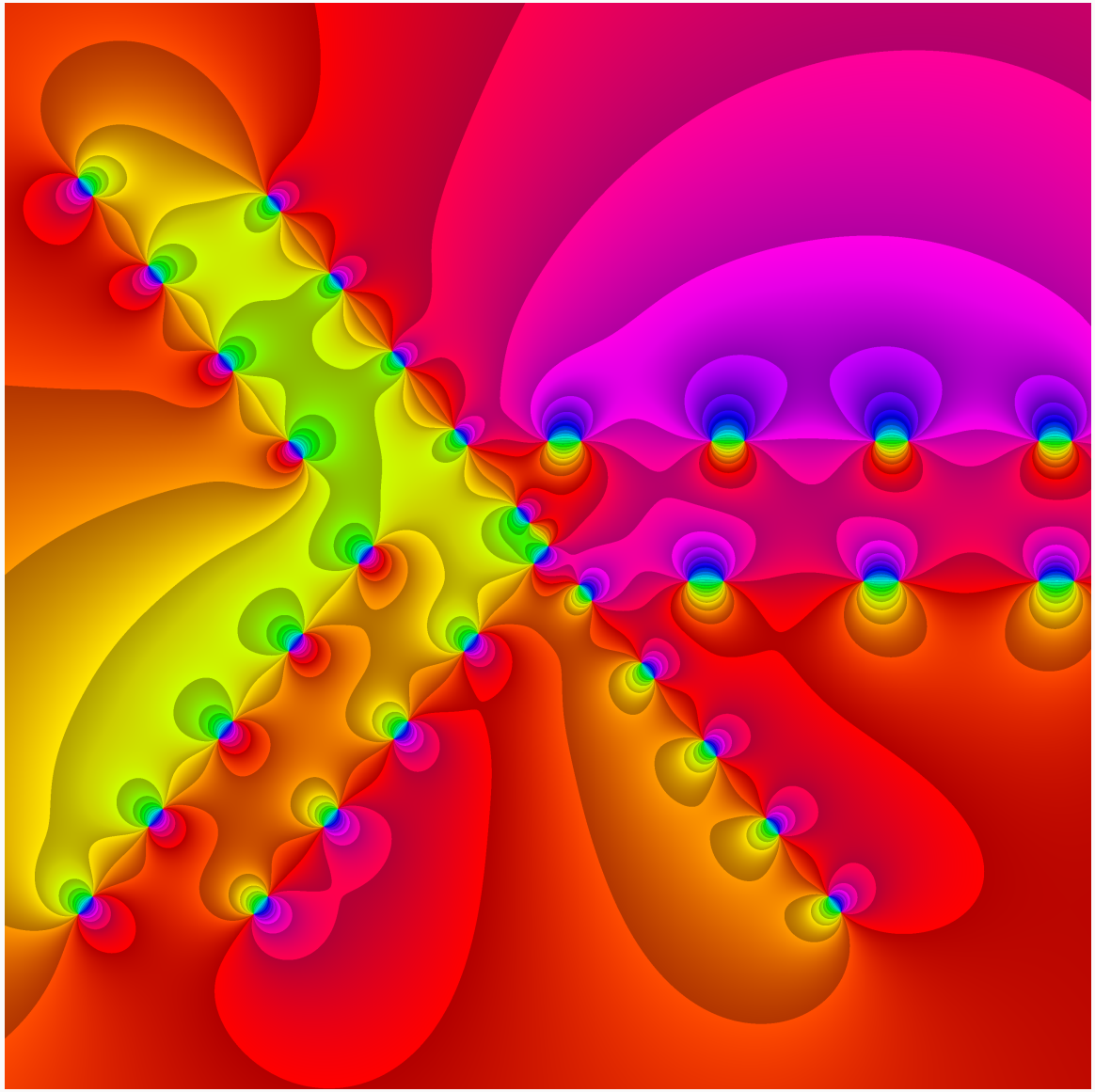


Stats

Statistic	Tobias Markus	Severin Schmidmeier
Moves made	49	49
Orbs captured	40	89
Capture/loss ratio	0.4494	2.2250

Competitions

Creativity



Competitions

Creativity

```haskell
module Exercise_13 where

import Data.Bool (bool)
import Data.Maybe (fromMaybe)
import Data.List (stripPrefix, isPrefixOf, findIndex, genericIndex)
import Data.Char (ord)
import Data.Word (Word8)
import qualified Data.ByteString as B
import Transform

animate :: [(String, Transform -> Transform)] -> String -> [String]
animate a s = map svg $ scanl (flip applyAnim) (parseInput s) $ map (:[]) a

paint :: String -> String
paint = svg . parseInput
```

Competitions

Offer challenges to go beyond the syllabus!

- Diverse, weekly competition exercises
- Awards for top 30 students

Competitions

Offer challenges to go beyond the syllabus!

- Diverse, weekly competition exercises
- Awards for top 30 students
- Works extremely well to motivate talented students...

Competitions

Offer challenges to go beyond the syllabus!

- Diverse, weekly competition exercises
- Awards for top 30 students
- Works extremely well to motivate talented students…

  but it is very time-consuming.

Check Your Proof

CYP In A Nutshell

- Operates on a strict, untyped subset of Haskell

CYP In A Nutshell

- Operates on a strict, untyped subset of Haskell
- Automatically checks simple inductive proofs

CYP In A Nutshell

- Operates on a strict, untyped subset of Haskell
- Automatically checks simple inductive proofs
- Provides instant feedback

CYP In A Nutshell

- Operates on a strict, untyped subset of Haskell
- Automatically checks simple inductive proofs
- Provides instant feedback
- Integrates with Tasty

Background Theory

```
data List a = [] | a : List a

[] ++ ys = ys
(x : xs) ++ ys = x : (xs ++ ys)

goal xs ++ (ys ++ zs) .=. (xs ++ ys) ++ zs
```

The [] Case

```
Lemma: xs ++ (ys ++ zs) .=. (xs ++ ys) ++ zs
Proof by induction on List xs
  Case []
    To show: [] ++ (ys ++ zs) .=. ([] ++ ys)
       ++ zs
    Proof
                          [] ++ (ys ++ zs)
      (by def ++) .=. ys ++ zs
      (by def ++) .=. ([] ++ ys) ++ zs
    QED
```

The Cons Case

```
Case x : xs
  To show: (x : xs) ++ (ys ++ zs)
        .=. ((x : xs) ++ ys) ++ zs
  IH: xs ++ (ys ++ zs) .=. (xs ++ ys) ++ zs
  Proof
                          (x : xs) ++ (ys ++ zs)
    (by def ++)     .=. x : (xs ++ (ys ++ zs))
    (by IH)         .=. x : ((xs ++ ys) ++ zs)

                          ((x : xs) ++ ys) ++ zs
    (by def ++)     .=. (x : (xs ++ ys)) ++ zs
    (by def ++)     .=. x : ((xs ++ ys) ++ zs)
  QED
QED
```

Our Experience With CYP

- Student feedback 18 positive, 3 negative

Our Experience With CYP

- Student feedback 18 positive, 3 negative
- Main criticism: lack of documentation

Our Experience With CYP

- Student feedback 18 positive, 3 negative
- Main criticism: lack of documentation
- Mostly well-structured inductive proofs in the exam

Find more in our repository!

- IO mocking framework
- ACM-ICPC-like programming contest framework
- A music synthesiser
- More engagement mechanisms and insights, our technical setup,...

`thub.com/kappelmann/engaging-large-scale-functional-programmi`

Future Work

Future Work

Preventing collaboration/cheating

Any questions?