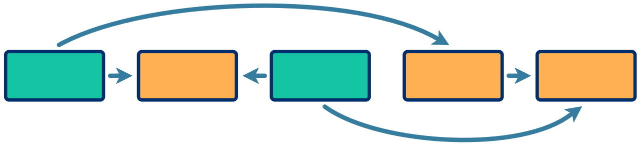
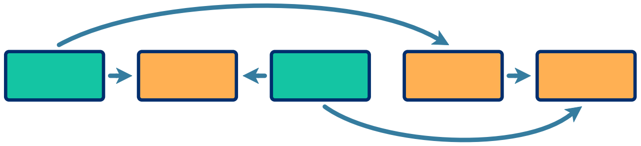
Vision one language with typed and untyped code



Vision one language with typed and untyped code

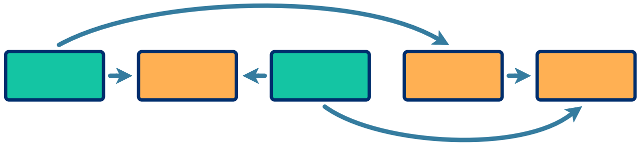


Problem tradeoffs!

- flexible mixes
- strong type guarantees
- run-time costs

Ben Greenman @ Brown

Vision one language with typed and untyped code

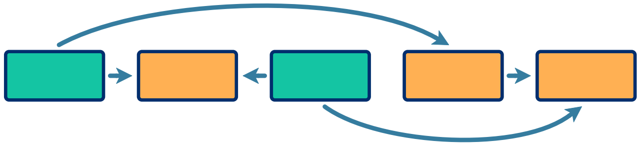


Problem tradeoffs!

- ➤ flexible mixes
- ➤ strong type guarantees
- ➤ run-time costs

Challenge develop methods to understand the tradeoffs (proofs + performance + **people**)

Vision one language with typed and untyped code



Problem tradeoffs!

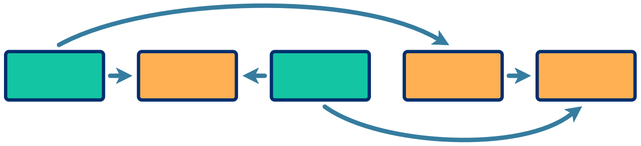
- flexible mixes
- strong type guarantees
- run-time costs

Challenge develop methods to understand the tradeoffs (proofs + performance + **people**)

Q . How to assess the pragmatics for working programmers?

Ben Greenman @ Brown

Vision one language with typed and untyped code



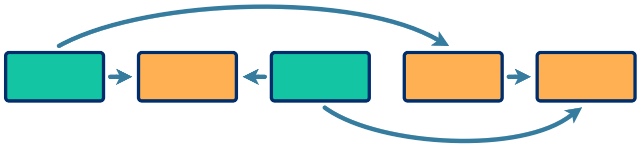
Problem tradeoffs!

- flexible mixes
- strong type guarantees
- run-time costs

Challenge develop methods to understand the tradeoffs
(proofs + performance + **people**)

Q . How to assess the pragmatics for working programmers?

Vision one language with typed and untyped code



Problem tradeoffs!

- ➤ flexible mixes
- ➤ strong type guarantees
- ➤ run-time costs

Challenge develop methods to understand the tradeoffs
(proofs + performance + **people**)

Q . How to assess the pragmatics for working programmers?

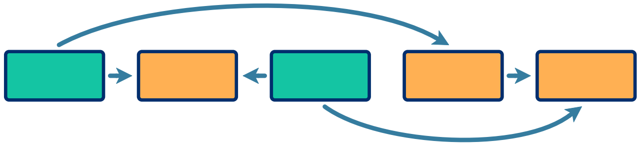
Surveys

qualtrics

+ easy to run ...

- for small q's

Vision one language with typed and untyped code



Problem tradeoffs!

- ➤ flexible mixes
- ➤ strong type guarantees
- ➤ run-time costs

Challenge develop methods to understand the tradeoffs
(proofs + performance + **people**)

Q . How to assess the pragmatics for working programmers?

Surveys

+ easy to run ...

- for small q's

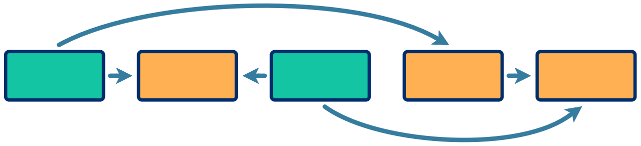
User Logs

+ realistic data

- hard to generalize

Ben Greenman @ Brown

Vision one language with typed and untyped code



Problem tradeoffs!

- ➤ flexible mixes
- ➤ strong type guarantees
- ➤ run-time costs

Challenge develop methods to understand the tradeoffs (proofs + performance + **people**)

Q . How to assess the pragmatics for working programmers?

Surveys

qualtrics

+ easy to run ...

- for small q's

User Logs

+ realistic data

- hard to generalize

The Rational Programmer

+ direct, scalable

- hard to design