How (not) to Give a Bad Research Talk

Leonidas Lampropoulos

University of Maryland

Outline

- The Research Talk
  - Why?
  - How?
- Preparing for a Research Talk
- General Tips and Tricks

Outline

- The Research Talk

Outline

- The Research Talk
  - Why?

Outline

- The Research Talk
  - Why?
  - How?

Outline

- The Research Talk
  - Why?
  - How?
- Preparing for a Research Talk

Outline

- The Research Talk
  - Why?
  - How?
- Preparing for a Research Talk
- General Tips and Tricks

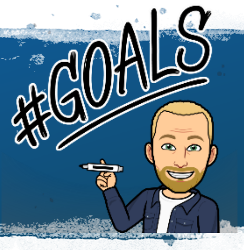
Waste time on outlines!

Annoy your audience with tedious animations!

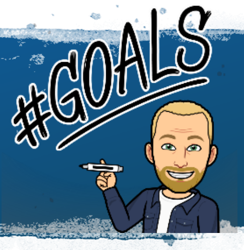
Need some signposting...

The Research Talk

The Research Talk



The Research Talk



Advertise your work	Explain all details of your work
Convey intuition	Show how smart you are
High-level contribution	Hear yourself speak
Conference requirement	Advisor made me
	Diss on related work

The Research Talk #GOALS

Examples!

Advertise your work

Convey intuition

High-level contribution

Conference requirements

Explain what you did

Explain all details of your work

Show how smart you are

Hear yourself speak

Advisor made me

Diss on related work

Explain too much

An Example Example

An Example Example

Property-Based Testing

Property-Based Testing and Random Generation

- Programmers write properties of software system or component as a function from sample inputs to Booleans.
- Tool generates many random inputs and applies the function to each one.
- Famously embodied in Haskell's QuickCheck by Koen Claessen and John Hughes.
- The Problem: properties with preconditions make generation hard.

Property-Based Testing and Random Generation

- Programmers write properties of software system or component as a function from sample inputs to Booleans.
- Tool generates many random inputs and applies the function to each one.
- Famously embodied in Haskell's QuickCheck by Koen Claessen and John Hughes.
- The Problem: properties with preconditions make generation hard.

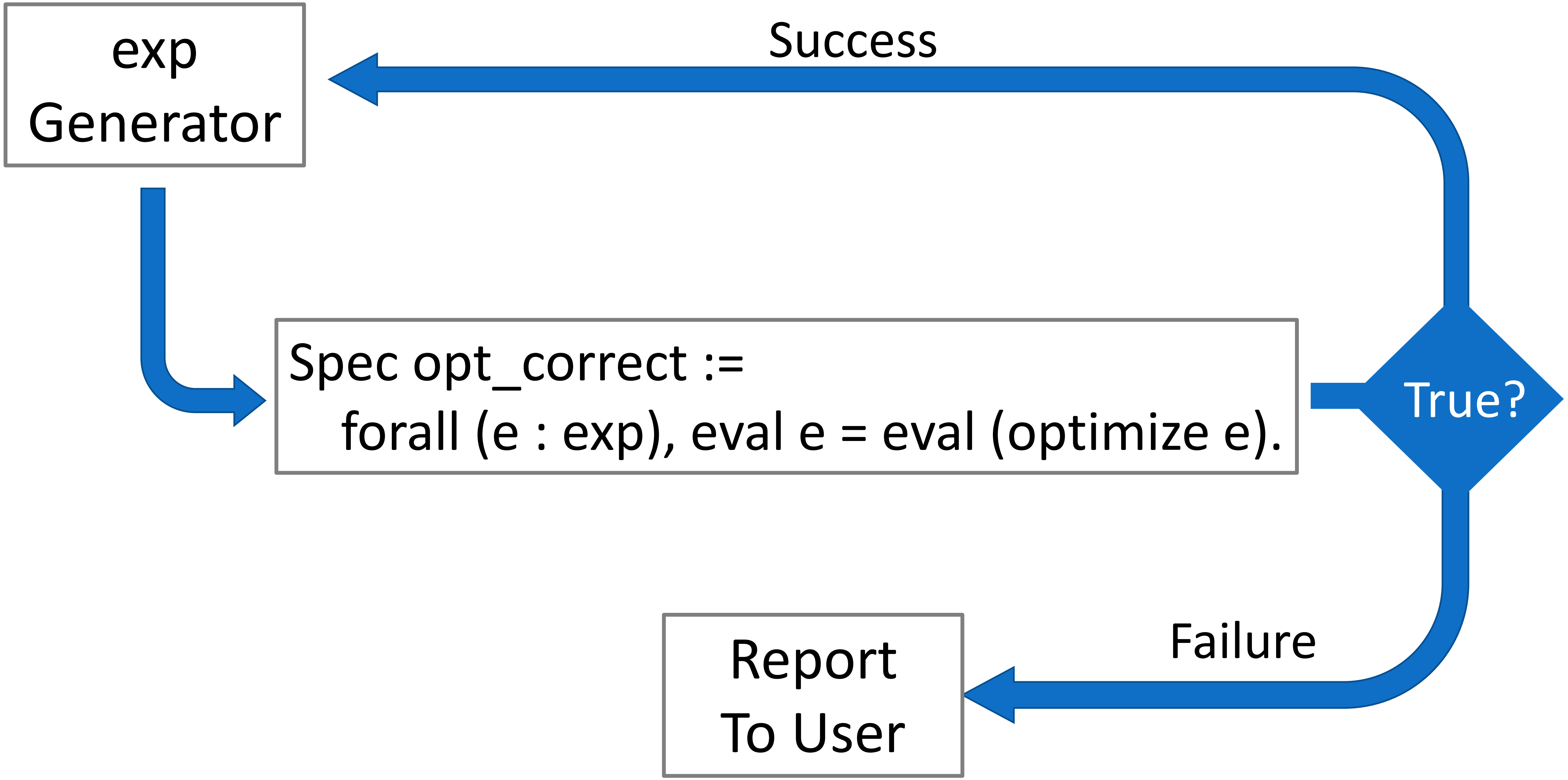
How do we test this?

```
Spec opt_correct :=
  forall (e : exp), eval e = eval (optimize e).
```

Property-Based Testing and Random Generation

➡

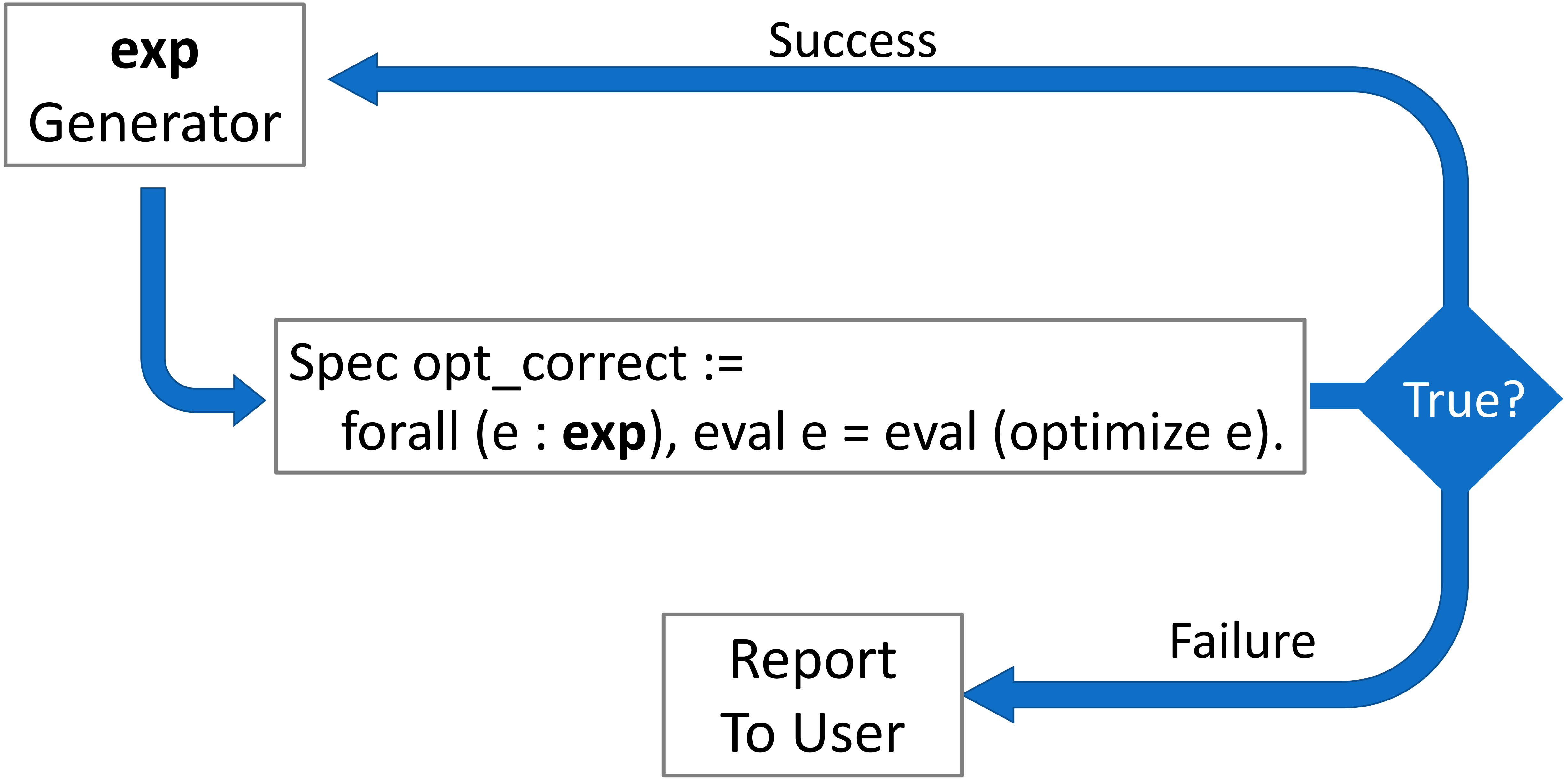
- Programmers write properties of software system or component as a function from sample inputs to Booleans.
- Tool generates many random inputs and applies the function to each one.
- Famously embodied in Haskell's QuickCheck by Koen Claessen and John Hughes.
- The Problem: properties with preconditions make generation hard.



```
Spec opt_correct :=
   forall (e : exp), eval e = eval (optimize e).
```

Property-Based Testing and Random Generation

- Programmers write properties of software system or component as a function from sample inputs to Booleans.
- Tool generates many random inputs and applies the function to each one.
- ~~Fa[illegible]usl[illegible]mbodied in Haskell's QuickCheck by Koen Cl[illegible]sen[illegible]d John Hughes.~~
- The Problem: properties with preconditions make generation hard.



```
Spec opt_correct :=
   forall (e : exp), eval e = eval (optimize e).
```

$$\forall x{:}\,T.\,p(x)$$

1. Generate x based on type information
2. Test $p(x)$

The Problem: Properties with preconditions

$$\forall x{:}\,T.\ p(x) \rightarrow q(x)$$

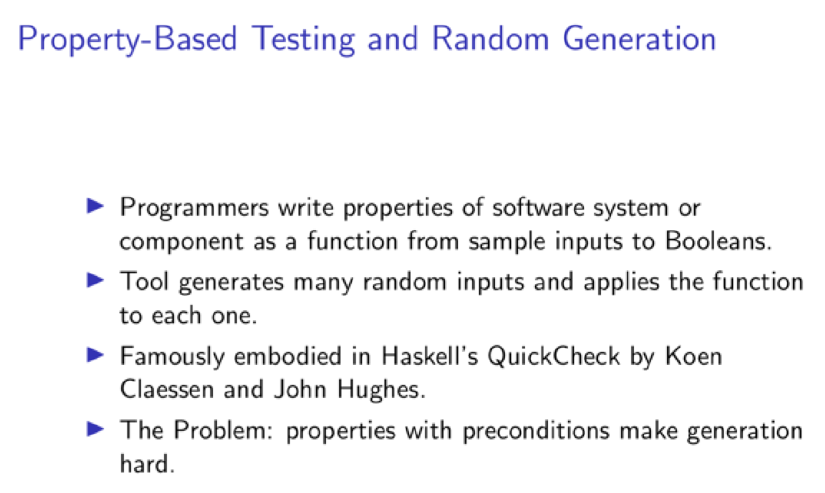
1. Generate x based on type information
2. Check $p(x)$
3. If $p(x)$ holds, test $q(x)$
4. If not, start over

The Problem: Properties with preconditions

$$\forall x{:}\,T.\ p(x) \rightarrow q(x)$$

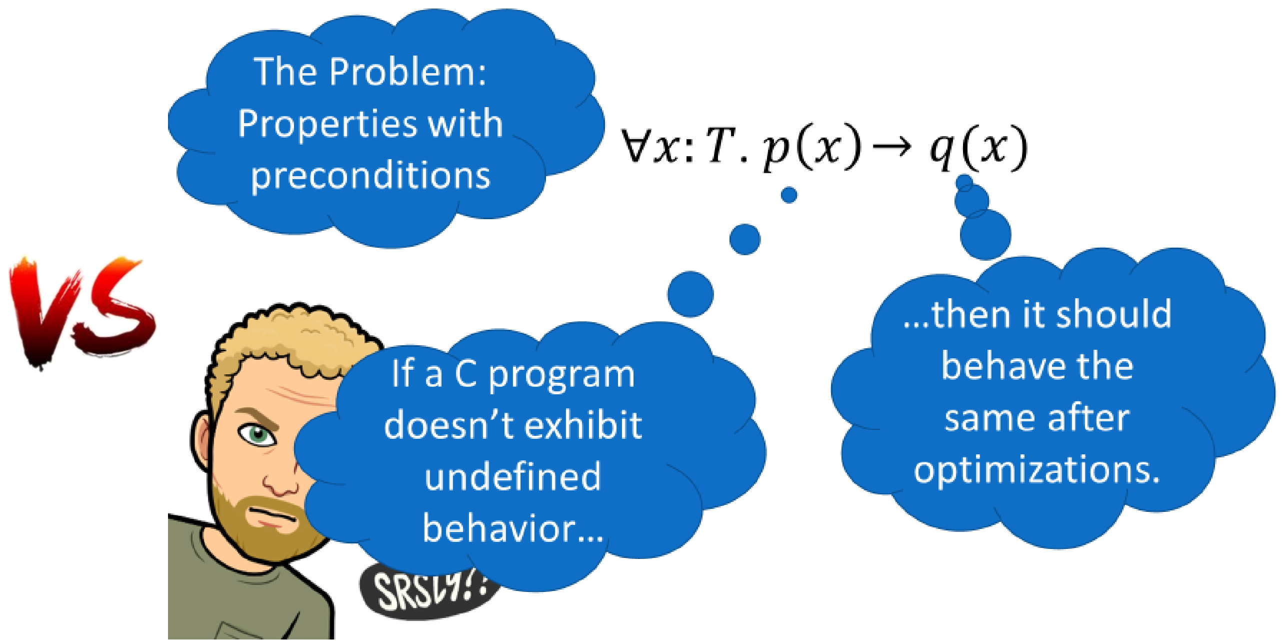
If a C program doesn't exhibit undefined behavior…

…then it should behave the same after optimizations.



Property-Based Testing and Random Generation

- Programmers write properties of software system or component as a function from sample inputs to Booleans.
- Tool generates many random inputs and applies the function to each one.
- Famously embodied in Haskell's QuickCheck by Koen Claessen and John Hughes.
- The Problem: properties with preconditions make generation hard.


VS
The Problem: Properties with preconditions
$\forall x\colon T.\ p(x) \rightarrow q(x)$
If a C program doesn't exhibit undefined behavior…
…then it should behave the same after optimizations.
SRSLY?!

Intuition through example

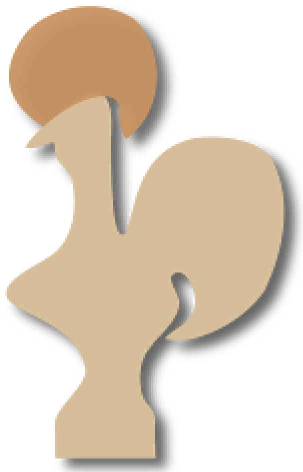
Less time for technical details

The Research Talk

Explain what you did

Explain too much

```
Spec opt_correct :=
    forall (e : exp), eval e = eval (optimize e).
```

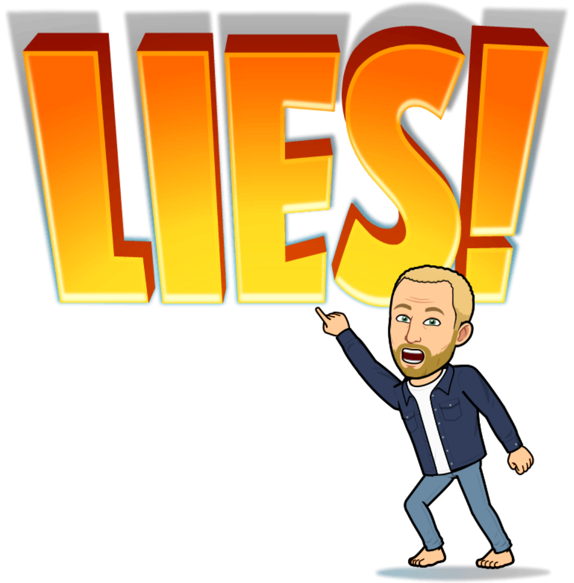


```
Spec opt_correct :=
  forall (e : exp), eval e = eval (optimize e).
```

What is Spec?

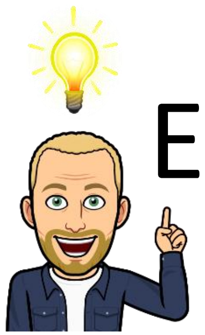
Propositional equality?

Familiar, but not off-putting



The Research Talk

LIES!
Explain what you did
Explain too much



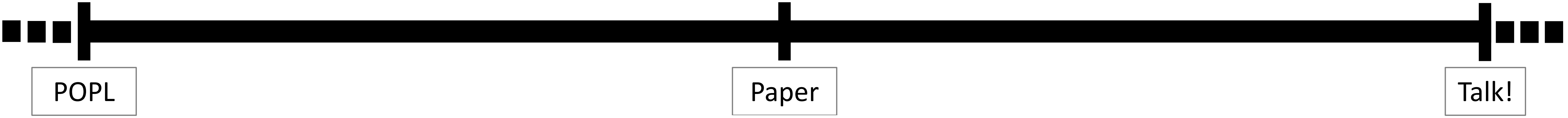
Examples!

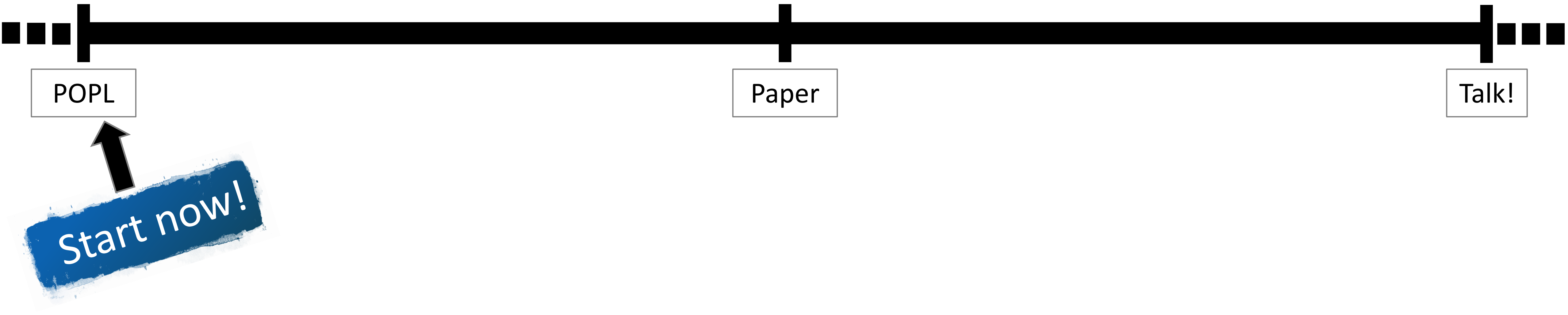
Preparing for a Research Talk

The Timeline

The Timeline

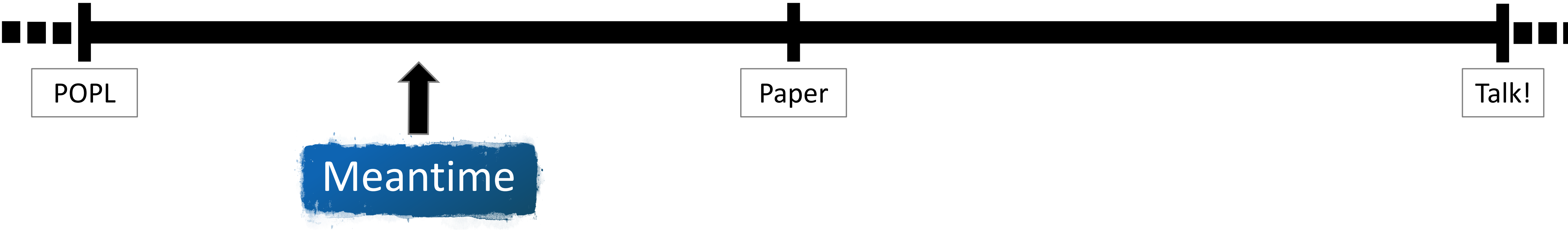
When does preparing start?





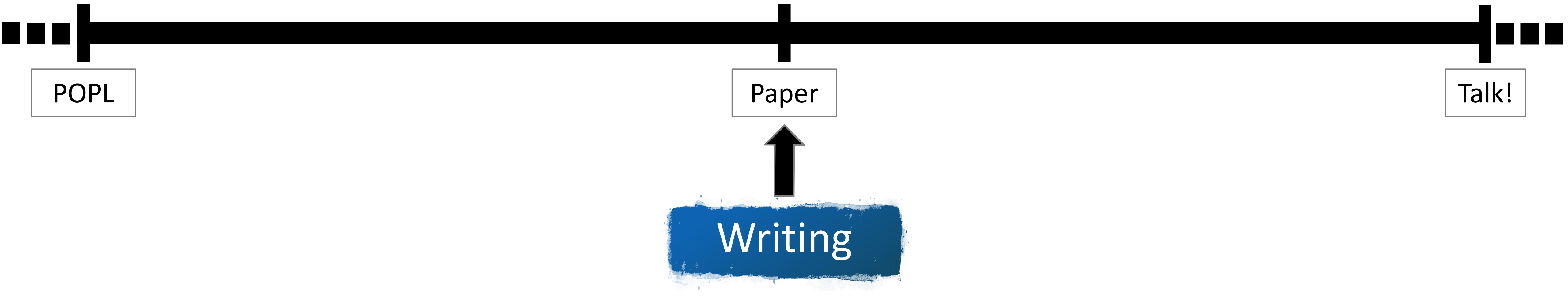
Go to other people's talks!

- Research talks at conferences (not all of them!)
- Job talks at your university
  - Look at people's reactions!
  - See what works and what doesn't

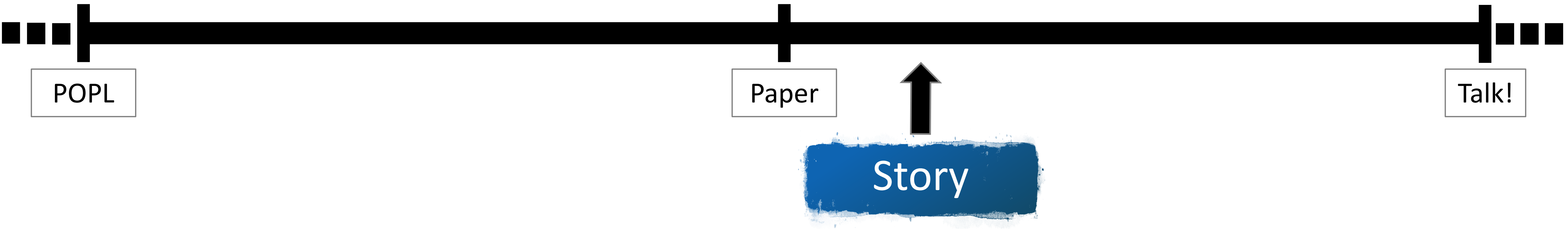


Give other talks!

- Reading group presentations at your university
- Final project presentations
- Regional seminars (NJPLS/MAPLS/…)



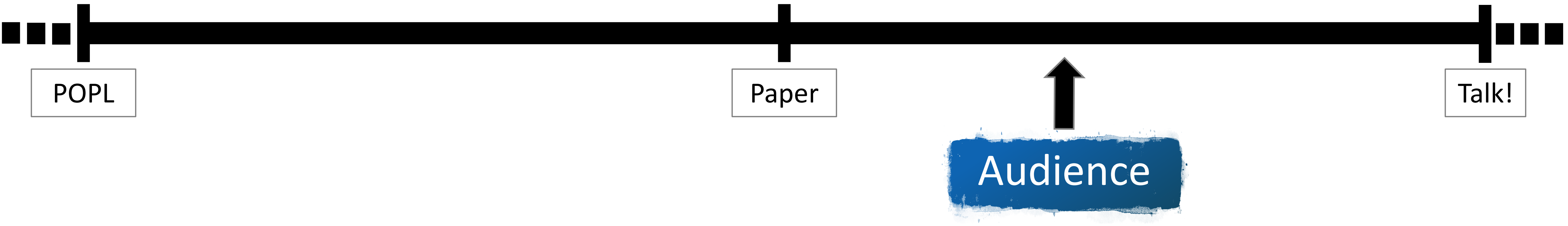
Writing PLMW talks!



Story of the paper NOT the talk!

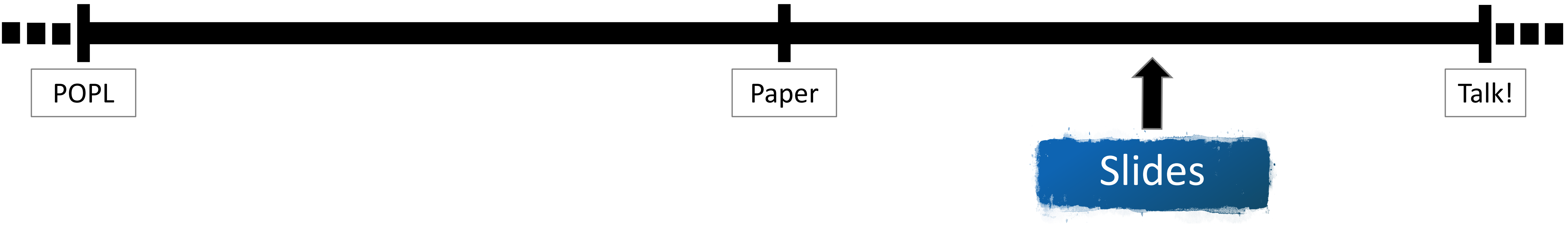
Motivation + Hig on

…but it’s a good starting point!



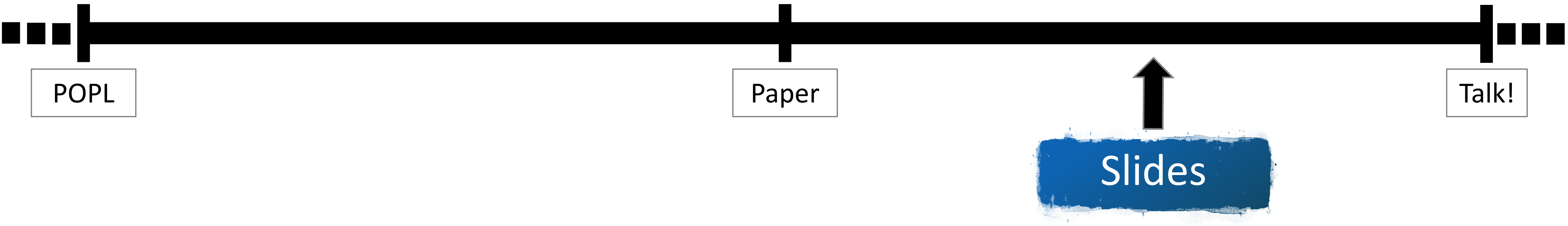
Curse of Knowledge

- Write for you *now*, not your expert reviewers!
- Don’t be defensive



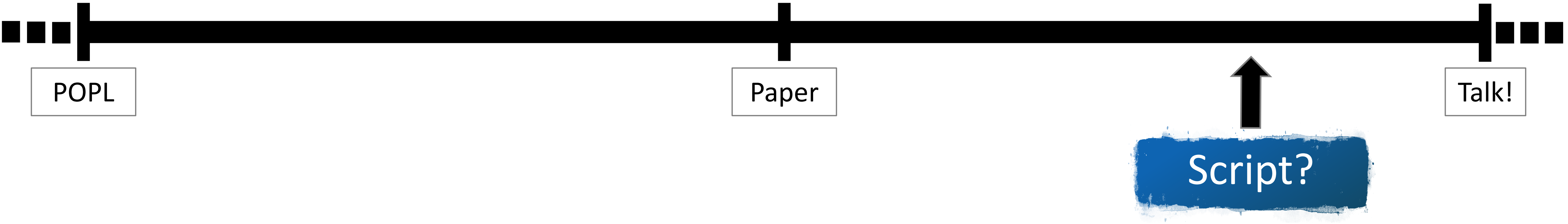
Use Presentation Software

- Good beamer talks *are* possible.
- Beamer makes it too easy to fall into bad habits.

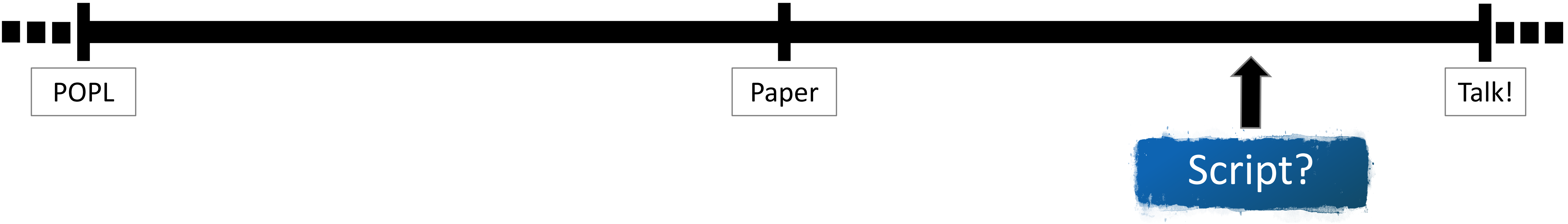


Use Presentation Software

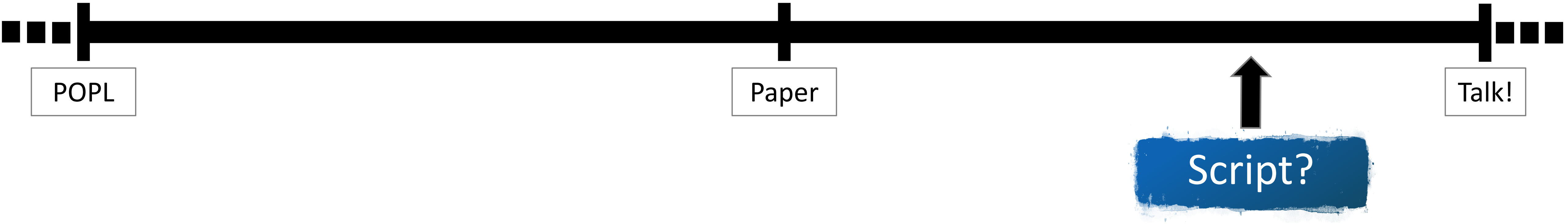
- Good beamer talks *are* possible.
- Beamer makes it too easy to fall into bad habits.
- *Beamer puts people off…*



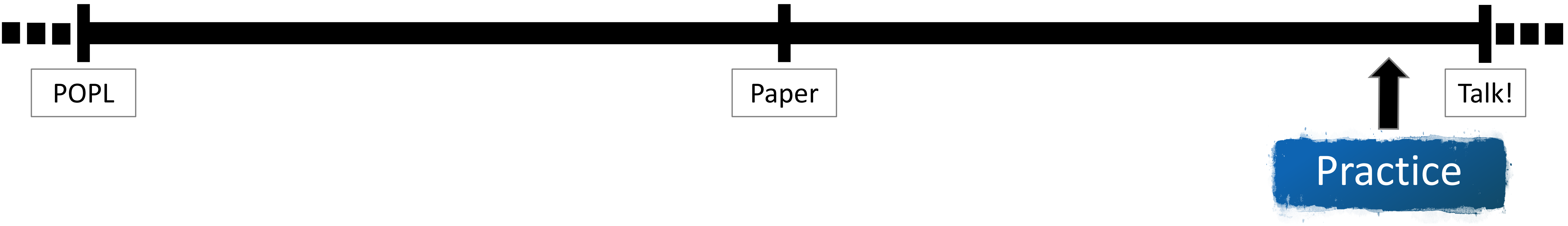
Script the entire thing



Script the intro



Script the transitions



Practice

- By yourself. Many times. Again and again.
- To people. **Listen** to feedback.
- Record yourself. Prepare to cringe.
- Timing…

Classical Processes

CP: a typed π-calculus. Type system: Classical Linear Logic (CLL).
Formulas of CLL are **session types**.

$$
\begin{array}{rlr}
A, B, \ldots ::= & \bot & \text{(receive end-of-session signal)} \\
& \mid \mathbf{1} & \text{(send end-of-session signal)} \\
& \mid A \parr B & \text{(input } A \text{ and continue as } B\text{)} \\
& \mid A \otimes B & \text{(output } A \text{ and continue as } B\text{)} \\
& \mid A \mathbin{\&} B & \text{(offer choice of } A \text{ or } B\text{)} \\
& \mid A \oplus B & \text{(select one of } A \text{ or } B\text{)} \\
& \mid ?A & \text{(replicated service consumer)} \\
& \mid !A & \text{(replicated service producer)}
\end{array}
$$

Duality:

$$(A \otimes B)^\perp \stackrel{\text{def}}{=} A^\perp \parr B^\perp \quad (A \oplus B)^\perp \stackrel{\text{def}}{=} A^\perp \mathbin{\&} B^\perp \quad (?A)^\perp \stackrel{\text{def}}{=} !A^\perp$$

We have $A^{\perp\perp} = A$.

Classical Processes

$\boxed{P \vdash x : A}$ "P will communicate along channel x according to A."

$$\frac{P \vdash \Gamma, x : A \qquad Q \vdash \Delta, x : A^{\perp}}{\nu x.\,(P \mid Q) \vdash \Gamma, \Delta} \qquad \frac{}{x \leftrightarrow y \vdash x : A^{\perp}, y : A}$$

$$\frac{P \vdash \Gamma, x : A, y : B}{y(x).\,P \vdash \Gamma, y : A \parr B} \qquad \frac{P \vdash \Gamma, x : A \qquad Q \vdash \Delta, y : B}{y[x].\,(P \mid Q) \vdash \Gamma, \Delta, y : A \otimes B}$$

$$\frac{P \vdash \Gamma, x : A}{x[\mathrm{inl}].\,P \vdash \Gamma, x : A \oplus B} \qquad \frac{Q \vdash \Gamma, y : B}{y[\mathrm{inr}].\,Q \vdash \Gamma, y : A \oplus B}$$

$$\frac{P \vdash \Gamma, x : A \qquad Q \vdash \Gamma, x : B}{x.\mathrm{case}\{P; Q\} \vdash \Gamma, x : A \,\&\, B}$$

$A \parr B$	input A and continue as B	**connected concurrency**
$A \otimes B$	output A and continue as B	**disjoint concurrency**

The Maypole Dance

$$P \vdash c_1 : A_1, \ldots, c_n : A_n$$

$$\frac{P \vdash \Gamma, x : A, y : B}{y(x).\, P \vdash \Gamma, y : A \parr B} \qquad \frac{P \vdash \Gamma, x : A \qquad Q \vdash \Delta, y : B}{y[x].\,(P \mid Q) \vdash \Gamma, \Delta, y : A \otimes B}$$

Expressivity I: Ad-hoc approaches

Atkey et al. [2016]: conflating connectives 'generates' concurrency.
E.g. to conflate $\otimes$ and $⅋$ add

$$\frac{P \vdash \Gamma \qquad Q \vdash \Delta}{P \mid Q \vdash \Gamma, \Delta}\ \textsc{Mix} \qquad\qquad \frac{}{\mathbf{0} \vdash \cdot}\ \textsc{Mix0}$$

$$\frac{P \vdash \Gamma, x : A^{\perp}, y : B^{\perp} \qquad Q \vdash \Delta, x : A, y : B}{\nu xy.\,(P \mid Q) \vdash \Gamma, \Delta}\ \textsc{BiCut}$$

Conflating ! and ? creates **access points**.

Balzer et al. [ICFP 2017]: recursive types + sharing modalities.

The price to pay: either **deadlock** or **livelock**.

Expressivity II: Exponentials

- Kokke et al. [LMCS 2020]: using techniques from **bounded linear logic**. Graded modalities $?_n A$ and $!_n A$.

$$\frac{\vdash \Gamma, A}{\vdash \Gamma, !_1 A} \qquad \frac{\vdash \Gamma, !_m A \qquad \vdash \Delta, !_n A}{\vdash \Gamma, \Delta, !_{m+n} A}$$

$$\frac{\vdash \Gamma, A}{\vdash \Gamma, ?_1 A} \qquad \frac{\vdash \Gamma, ?_m A, ?_n A}{\vdash \Gamma, ?_{m+n} A}$$

- Adding rules from Ehrhard's **differential linear logic**:

$$\frac{\vdash \Gamma, !A \qquad \vdash \Delta, !A}{\vdash \Gamma, \Delta, !A} \qquad \frac{\vdash \Gamma, A}{\vdash \Gamma, !A} \qquad \frac{}{\vdash !A} \qquad \frac{\vdash \Gamma \qquad \dots \qquad \vdash \Gamma}{\vdash \Gamma}$$

See e.g. Yoshida, Castellan and Stefanesco [arXiv:2011.05248].

Expressivity III: (Co)inductive types

Baelde [ToCL 2012], followed by Lindley and Morris [ICFP 2016]:
least and greatest fixed points μF and νF for positive functors F.

$$(\nu F)^\perp = \mu(F^\perp)$$

$$\frac{P \vdash \Gamma, x : F(\mu F)}{\mathrm{rec}\ x.\, P \vdash \Gamma, x : \mu F} \qquad \frac{P \vdash \Gamma, y : A \qquad Q \vdash y : A^\perp, x : F(A)}{\mathrm{corec}\ x[y].\,(P \mid Q) \vdash \Gamma, x : \nu F}$$

Qian, K, Birkedal [ICFP 2021]: custom-built cases work wonders.

Good properties are preserved!

Coexponentials

Specializing Baelde's system to

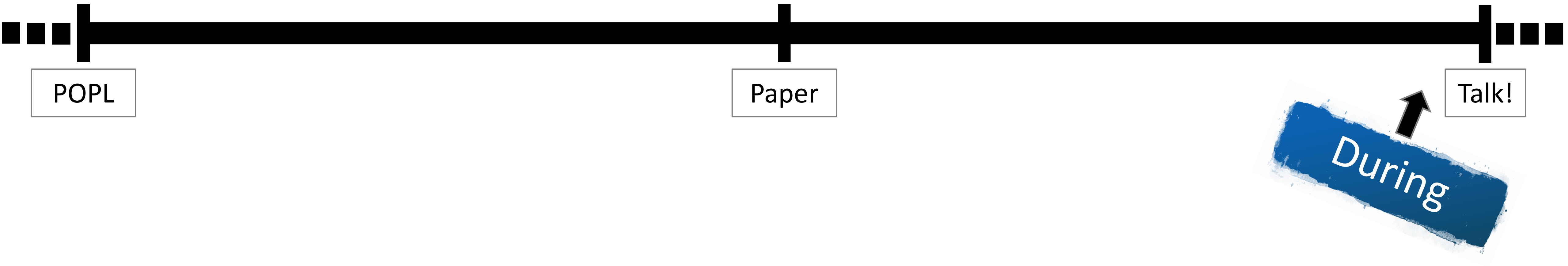
$$¿A \;\cong\; \mathbf{1} \oplus A \oplus (¿A \otimes ¿A) \qquad ¡A \;\cong\; \bot \mathbin{\&} A \mathbin{\&} (¡A \parr ¡A)$$

we obtain the following rules.

$$\frac{}{\vdash ¿A}\ ¿w \qquad \frac{\vdash \Gamma, A}{\vdash \Gamma, ¿A}\ ¿d \qquad \frac{\vdash \Gamma, ¿A \qquad \vdash \Delta, ¿A}{\vdash \Gamma, \Delta, ¿A}\ ¿c$$

$$\frac{\vdash \Gamma, B \qquad \vdash B^{\bot}, \bot \qquad \vdash B^{\bot}, A \qquad \vdash B^{\bot}, B \parr B}{\vdash \Gamma, ¡A}\ ¡$$

$\boxed{\begin{array}{c}¿ \text{ means client} \\ ¡ \text{ means server}\end{array}}$



- Enthusiasm vs calmness
- Enunciate
- Pacing
- Engagement – *Find the nodder!*

Break the rules!

Questions?

- Thank you!

How (not) to Give a Bad Research Talk

Go to other people's talks!

- Look at people's reactions!
- See what works and what doesn't

The Backup Slide

It’s good to have some!