Semantic Foundations of Higher-Order Probabilistic Programs in Isabelle/HOL

Michikazu Hirata, Yasuhiko Minamide, Tetsuya Sato

Tokyo Institute of Technology

ITP2023
August 3, 2023

Probabilistic Programs

Programmers **write probabilistic distributions as programs**.

The languages infer **posterior distributions** after observing some events.
(conditional distributions)

Anglican, Church, Hakaru, WebPPL, …

Higher-order probabilistic programs

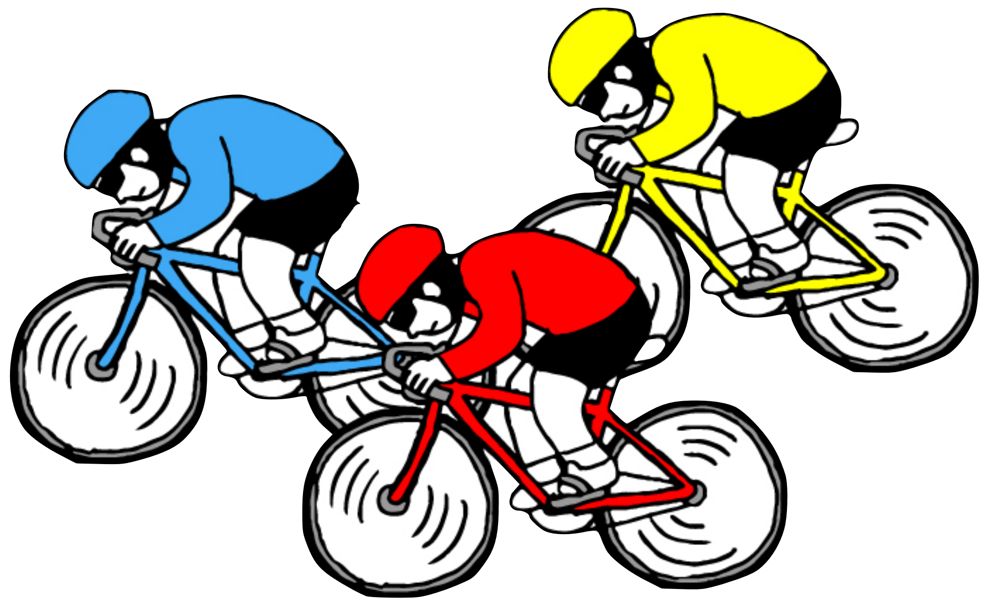
= Higher-order functions + probabilistic programs

higher-order functions, sampling, conditioning

Probabilistic Programs

Example (by Staton*)

1. We came to watch a road bicycle racing.
We want to know what time it is.



2. We know the rate of bikes per hour at each time.

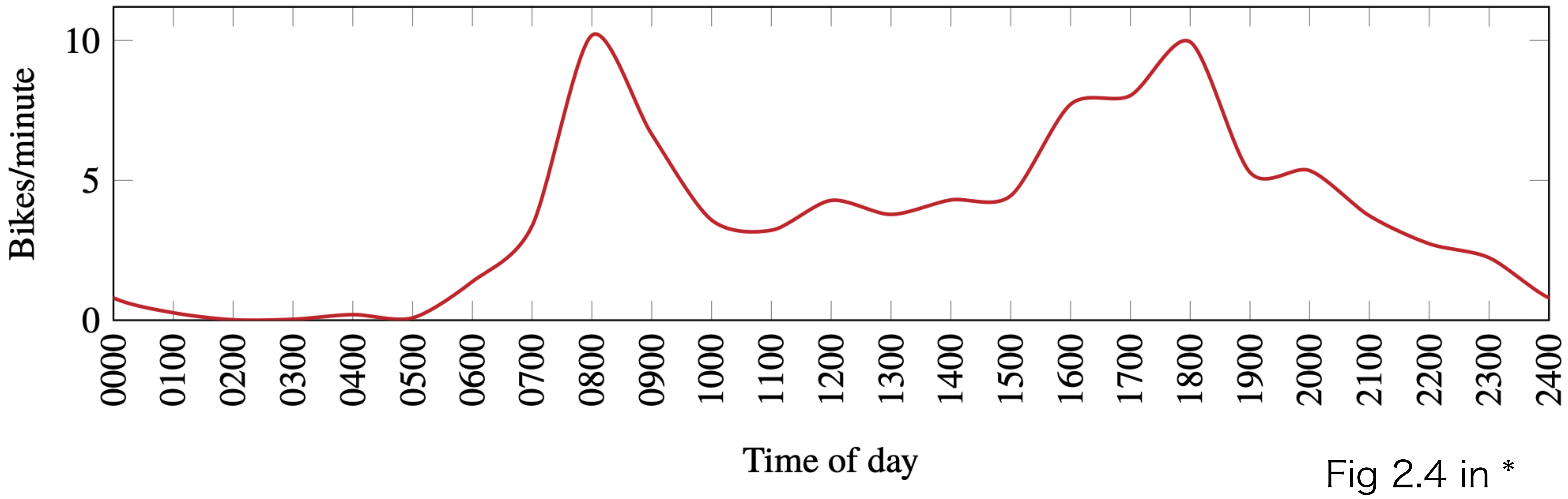
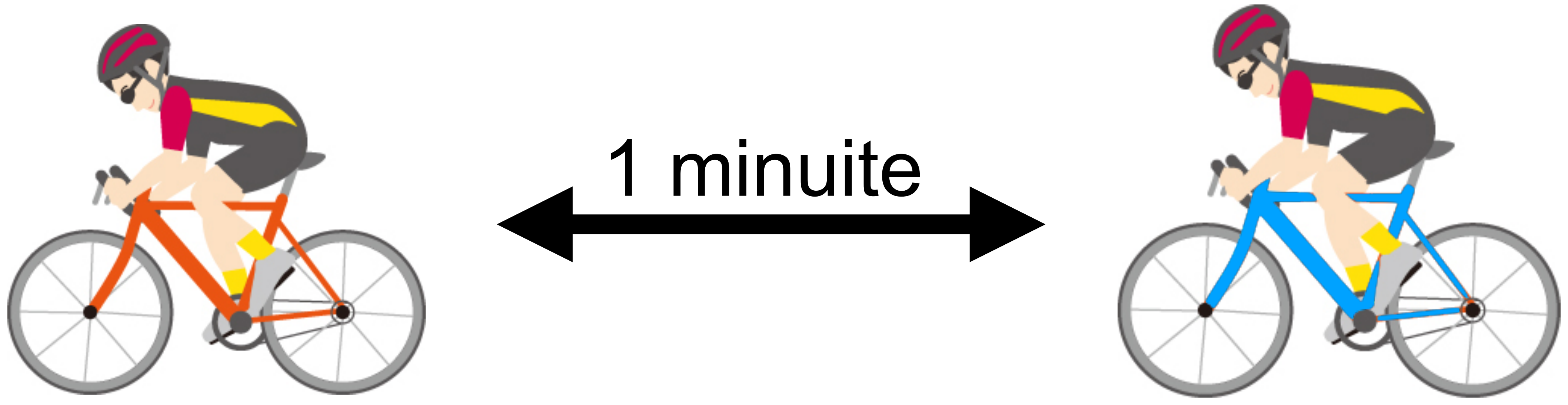


Fig 2.4 in *

3. We observed a 1 minute gap between two bikes.



4. What time is it?

* **Probabilistic Programs as Measures**, Sam Staton, Foundations of Probabilistic Programming, 2020

Probabilistic Programs

Posterior $\propto$ Likelihood $\times$ Prior

```
definition whattime :: "(real ⇒ real) ⇒ real qbs_measure" where
"whattime ≡ (λf. do {
                    let T = Uniform 0 24 in
                    query T (λt. let r = f t in
                            exponential_density r (1 / 60))
                  })"
```

The rate

Prior

Likelihood

1. We want to know what time it is.

2. We know the rate of bikes per hour.

3. We observed a 1 minute gap between two bikes.

4. What time is it ?

Semantics of Probabilistic Programs

Semantics based on measure theory

The probability monad (the Giry monad) or s-finite kernels

$\Gamma \vdash e : P(T)$ $\xrightarrow{\text{Interpretation}}$ A measurable function $[\![e]\!] : [\![\Gamma]\!] \to G([\![T]\!])$

Problem: Function spaces do **not** exist in general

Semantics based on quasi-Borel spaces [Heunen+, LICS 2017]

A suitable denotational model for higher-order probabilistic programs

Function spaces **always exist**

The s-finite measure monad on quasi-Borel spaces[Scibior+, POPL 2018]

(Probability measures $\subseteq$ σ-finite measures $\subseteq$ s-finite measures)

Previous Works

Many proof assistants have measure theory library (Isabelle/HOL, Lean, Coq, …)

Formalization of semantics of probabilistic programs

1. [Hirata+, FLOPS2022] Quasi-Borel spaces and the probability monad in Isabelle/HOL
2. [Affeldt+, CPP2023] S-finite kernels in Coq

- 1 does **not** support conditioning
- 2 does **not** support higher-order functions
- Both of 1 and 2 use de-Bruijn index to denote probabilistic programs

  Hard to read, write, and reason about programs

This Work(1)

Formalizing semantics for higher-order probabilistic programs with conditioning

Isabelle/HOL terms as probabilistic programs

```
definition whattime :: "(real ⇒ real) ⇒ real qbs_measure" where
"whattime ≡ (λf. do {
                        let T = Uniform 0 24 in
                        query T (λt. let r = f t in
                                     exponential_density r (1 / 60))
                     })"
```

Higher-order

Command for conditional distributions (based on the s-finite measure monad)

This Work(2)

Automated *type checking*

$\vdash$ whattime : (real $\Rightarrow$ real) $\Rightarrow$ M[real] (M[real]$\cdots$ Distribution on $\mathbb{R}$)

$$\vdash t : T \xrightarrow{\text{Interpretation}} [\![t]\!] \in \text{Quasi-Borel space } [\![T]\!]$$

In Isabelle/HOL

```
lemma "whattime ∈ (ℝQ ⇒Q ℝQ) ⇒Q monadM_qbs ℝQ"
  unfolding whattime_def by qbs
```

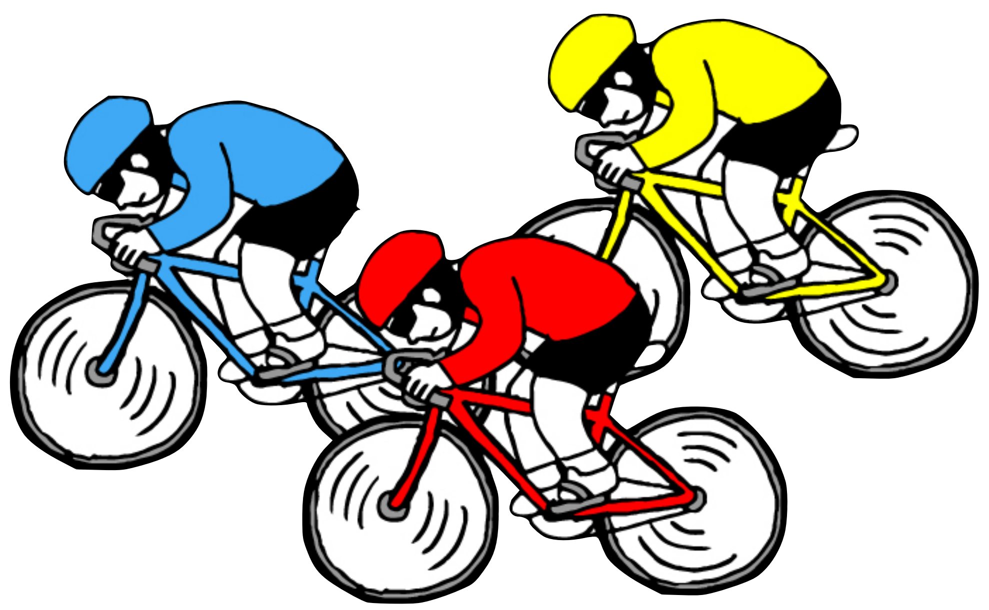
`(ℝQ ⇒Q ℝQ) ⇒Q monadM_qbs ℝQ` $\cdots$ A quasi-Borel space

Example: What time is it?

Example: What time is it?

Example (by Staton*)

1. We came to watch a road bicycle racing.
   We want to know what time it is.



2. We know the rate of bikes per hour at each time.

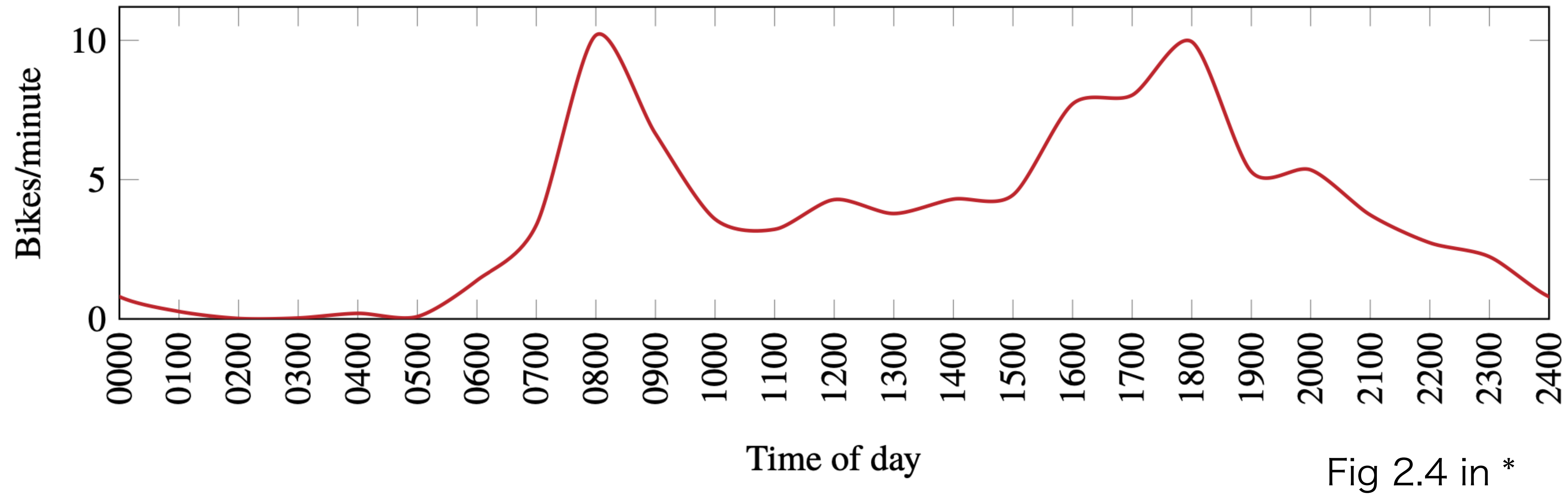


Fig 2.4 in *

3. We observed a 1 minute gap between two bikes.

1 minuite

4. What time is it?

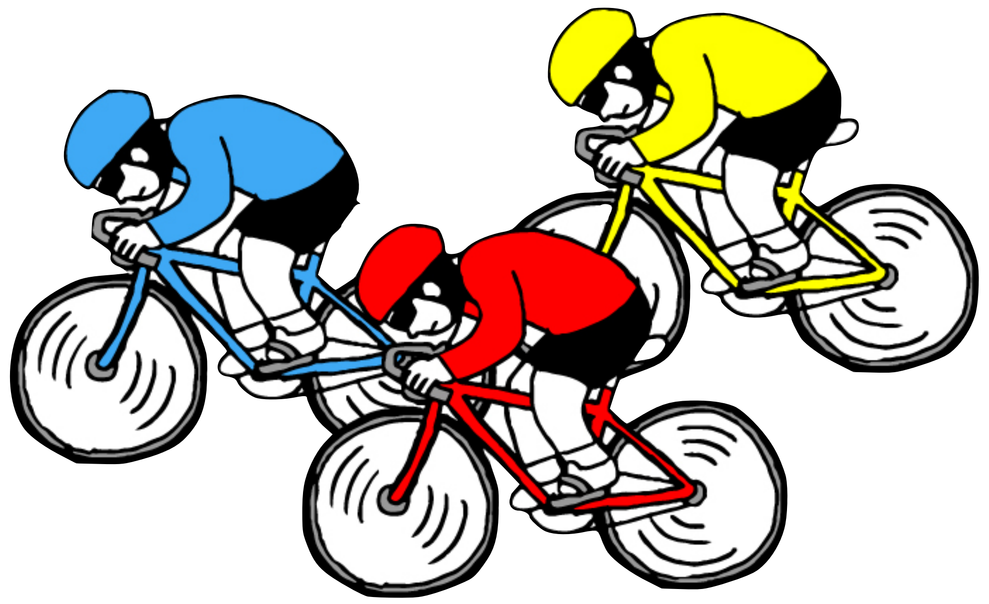
* **Probabilistic Programs as Measures**, Sam Staton, Foundations of Probabilistic Programming, 2020

Example: What time is it?

Example (by Staton*)

1. We came to watch a road bicycle racing.
We want to know what time it is.

Prior



2. We know the rate of bikes per hour at each time.

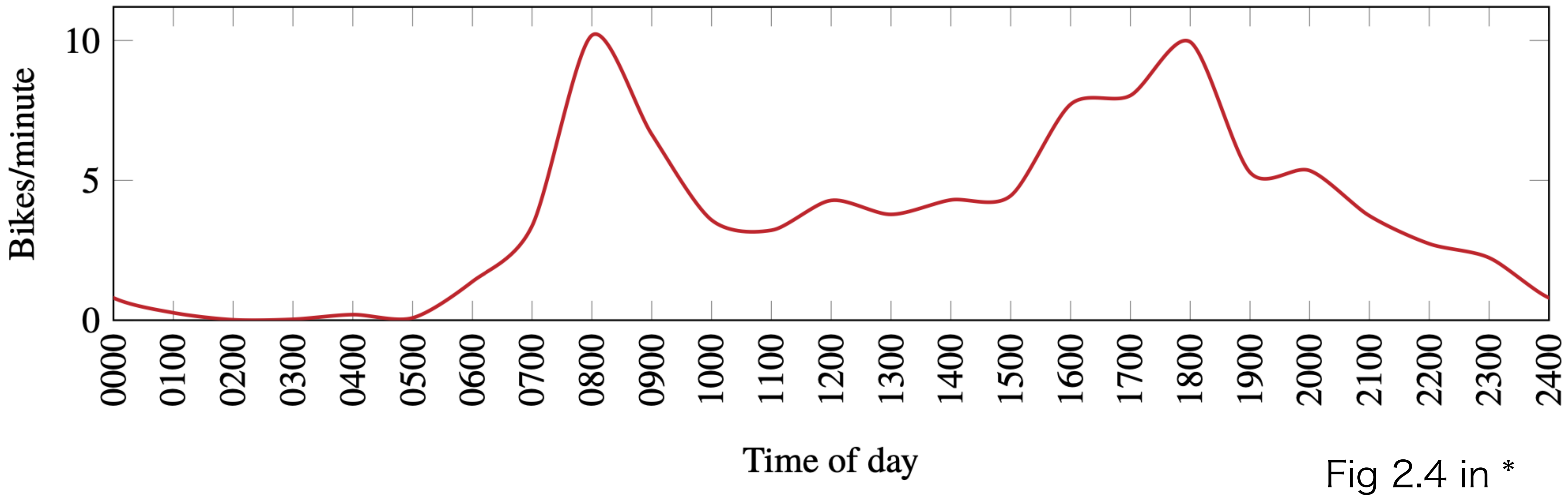
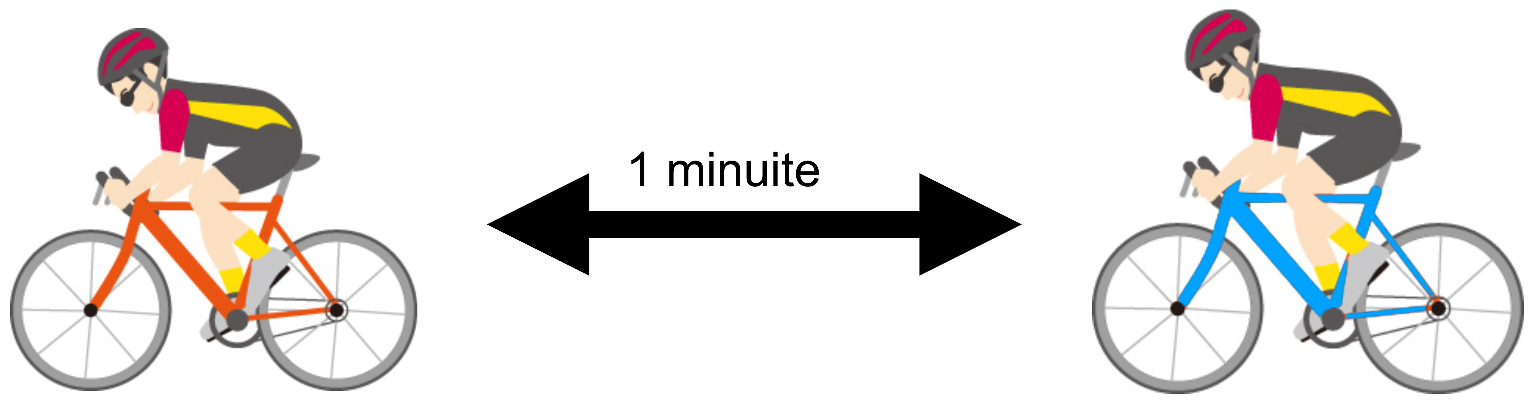


Fig 2.4 in *

3. We observed a 1 minute gap between two bikes.

Likelihood



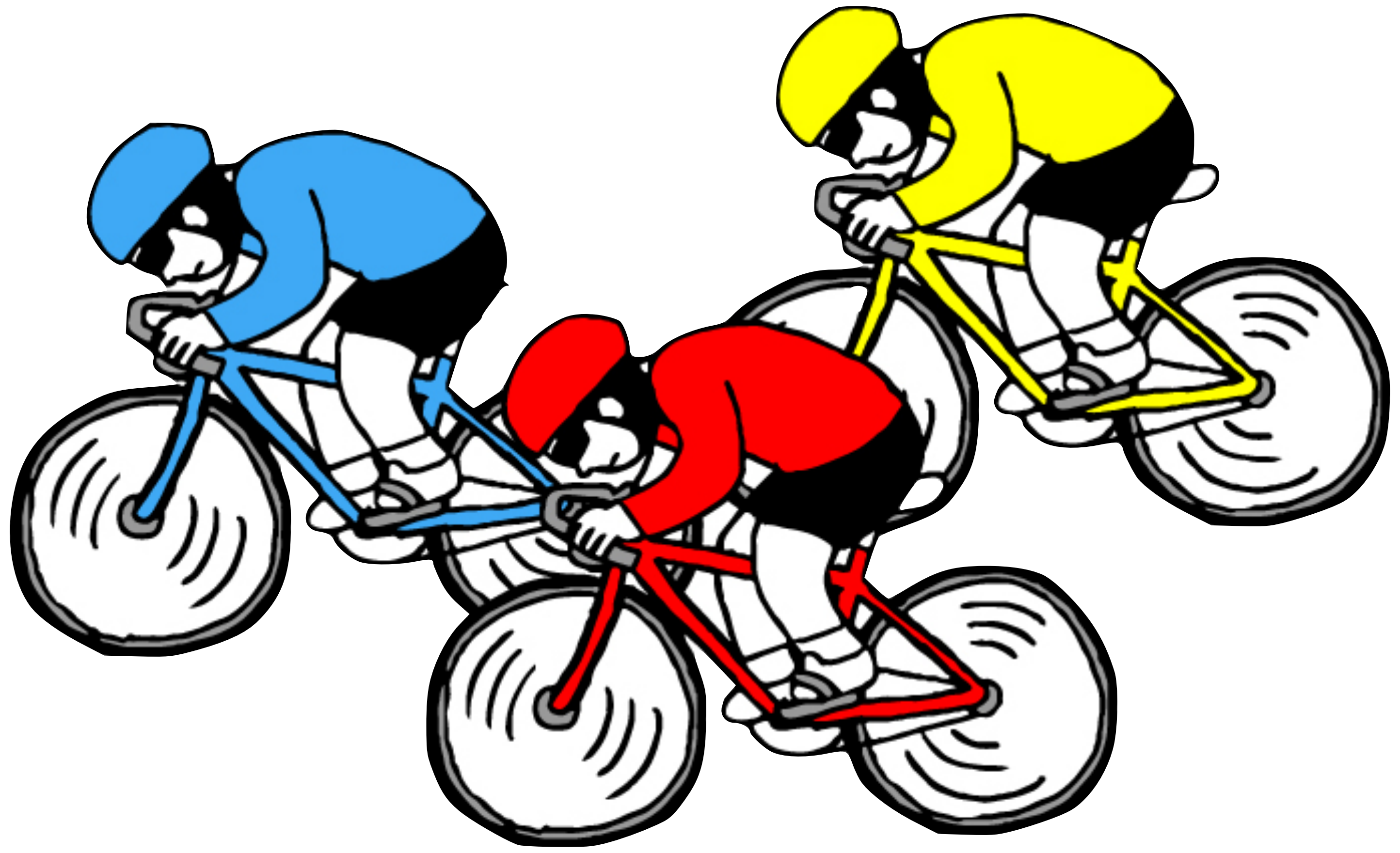
4. What time is it?

Posterior ∝ Likelihood × Prior

* **Probabilistic Programs as Measures**, Sam Staton, Foundations of Probabilistic Programming, 2020

Example: What time is it?

1. We came to watch a road bicycle racing.
   We want to know what time it is.



Prior: $\mathrm{Uniform}(0{,}24)$

pdf: $p(t) = \begin{cases} \frac{1}{24} & 0 < t < 24 \\ 0 & \text{o.w.} \end{cases}$

Example: What time is it?

2. We know the rate of bikes per hour at each time.

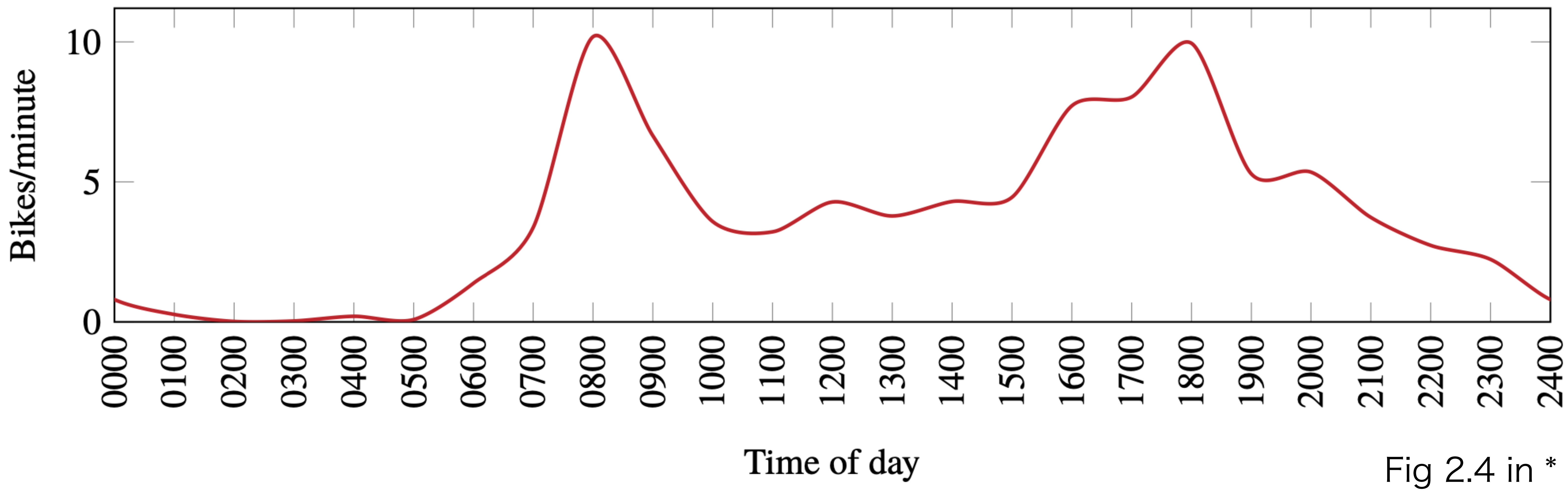


Fig 2.4 in *

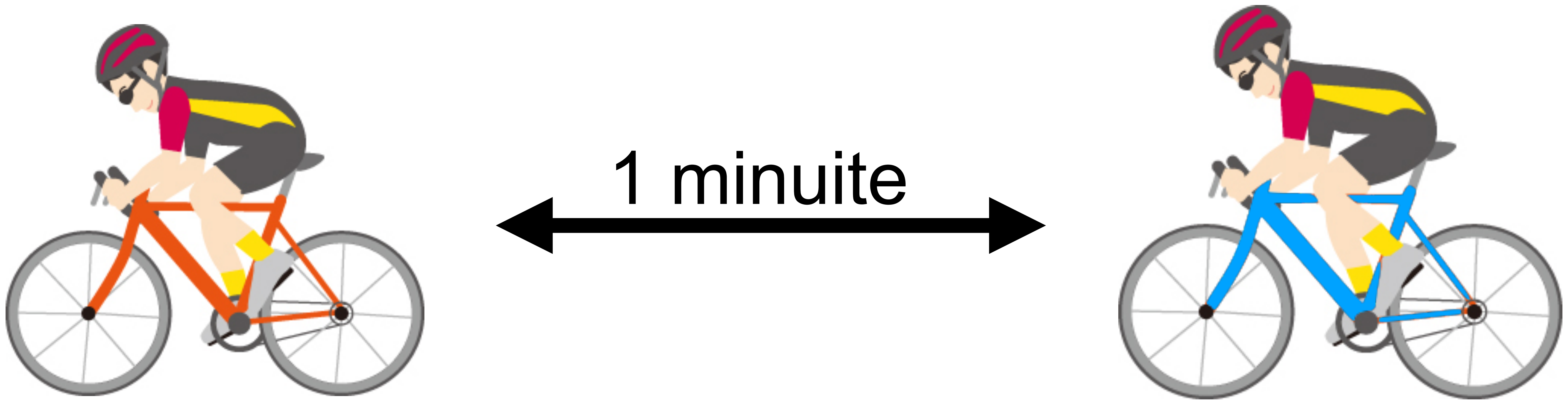
Given as an argument f

$f(t) \cdots$ The rate of bikes per hour at time t

* **Probabilistic Programs as Measures**, Sam Staton, Foundations of Probabilistic Programming, 2020

Example: What time is it?

3. We observed a 1 minute gap between two bikes



$\langle$The gap between two bikes at time $t\rangle \sim \text{Exponential}(f(t))$

$$\text{exp\_density}(x \mid f(t)) = f(t)e^{-xf(t)}$$

Likelihood function: $l(t) = \text{exp\_density}(1/60 \mid f(t)) = f(t)e^{-\frac{1}{60}f(t)}$

(1 minute = 1/60 hour)

Example: What time is it?

4. What time is it?

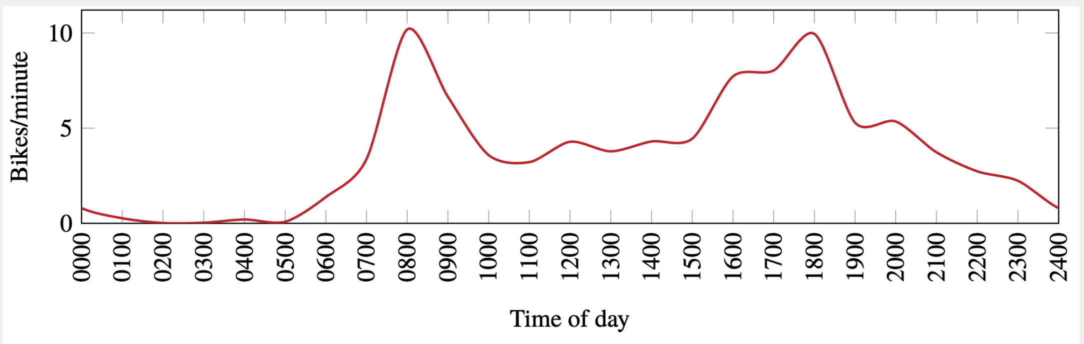
Posterior ∝ Likelihood × Prior

Prior: Uniform(0,24)

Likelihood function: $l(t) = f(t)e^{-\frac{1}{60}f(t)}$

```
definition whattime :: "(real ⇒ real) ⇒ real qbs_measure" where
"whattime ≡ (λf. do {
                    let T = Uniform 0 24 in
                    query T (λt. let r = f t in
                               exponential_density r (1 / 60))
                  })"
```

The rate



Prior

Likelihood

Example: What time is it?

Posterior ∝ Likelihood × Prior

$$\text{posterior\_pdf}(t) \propto f(t)e^{-\frac{1}{60}f(t)} \times \frac{1}{24}$$

$$\text{posterior\_pdf}(t) = \frac{f(t)e^{-\frac{1}{60}f(t)} \times \frac{1}{24}}{\int_{t=0}^{t=24} f(t)e^{-\frac{1}{60}f(t)} \times \frac{1}{24}\mathrm{d}t} = \frac{f(t)e^{-\frac{1}{60}f(t)}}{\underbrace{\int_{t=0}^{t=24} f(t)e^{-\frac{1}{60}f(t)}\mathrm{d}t}_{= N}}$$

$$\mathrm{P}_{T\sim\mathrm{Posterior}}[T \in U] = \frac{\int_{t\in(0,24)\cap U} f(t)e^{-\frac{1}{60}f(t)}\mathrm{d}t}{N}$$

```
lemma
  assumes "f ∈ ℝQ ⇒Q ℝQ" and "U ∈ sets borel" and "⋀r. f r ≥ 0"
  defines "N ≡ (∫t∈{0<..<24}. (f t * exp (- 1/ 60 * f t)) ∂lborel)"
  assumes "N ≠ 0" and "N ≠ ∞"
  shows "𝒫(t in whattime f. t ∈ U) = (∫t∈{0<..<24}∩U. (f t * exp (- 1/ 60 * f t)) ∂lborel) / N"
```

Around 100 lines for proof

Theory of Quasi-Borel Spaces and Implementing Probabilistic Programs

Quasi-Borel Spaces

A new denotational model for higher-order probabilistic programs[Heunen+, LICS2017]

- Function spaces always exist (For measurable spaces, do not in general)

  $\Longrightarrow$ Higher-order programs

- *Measures* on standard Borel spaces (e.g. $\mathbb{N}$, $\mathbb{R}$, and $\Pi_{i\in\mathbb{N}}\mathbb{R}$)

  $\Longrightarrow$ Probability distributions

- The s-finite measure monad[Scibior+, POPL 2018]

  $\Longrightarrow$ Semantics for probabilistic programs

Quasi-Borel Spaces in Isabelle/HOL

Our previous work[Hirata+, FLOPS2022]

Quasi-Borel Spaces

```
"X :: 'a quasi_borel"
```

consist of underlying set and “random variables”

```
"qbs_space X :: 'a set"
```

```
"qbs_Mx X :: (real ⇒ 'a) set"
```

Morphisms (structure preserving functions)

```
"X →Q Y = {f. ∀α∈qbs_Mx X. f ∘ α ∈qbs_Mx Y}"
```

Product Space

```
"X ⨂Q Y :: ('a × 'b) quasi_borel"
```

Function Space

```
"X ⇒Q Y :: ('a ⇒ 'b) quasi_borel"
```

```
"qbs_space (X ⇒Q Y) = X →Q Y"
```

List Space

```
"list_qbs X :: 'a list quasi_borel"
```

The S-Finite Measure Monad

Measures

```
"s :: 'a qbs_measure"
```

Space of Measures, Return and Bind

```
"monadM_qbs X :: 'a qbs_measure quasi_borel"
"return_qbs X ∈ X ⇒Q monadM_qbs X"
"(≫=) ∈ monadM_qbs X ⇒Q (X ⇒Q monadM_qbs Y) ⇒Q monadM_qbs Y"
```

The triple forms a commutative strong monad on **QBS**

Ex: Associativity

```
lemma
  assumes "s ∈ monadM_qbs X"
          "f ∈ X ⇒Q monadM_qbs Y"
      and "g ∈ Y ⇒Q monadM_qbs Z"
    shows "s ≫= (λx. f x ≫= g) = (s ≫= f) ≫= g"
```

Probabilistic Programming Language

Language

HPProg$\cdots$ A functional probabilistic programming language[Sato+, POPL2019]

$$T ::= \mathsf{nat} \mid \mathsf{bool} \mid \mathsf{real} \mid \mathsf{preal} \mid \mathsf{list}[T] \mid T \times T \mid T \Rightarrow T \mid M[T]$$

The type of distributions on T

$$e ::= x \mid c \mid e\ e \mid \lambda x\,.\,e$$

$$c ::= + \mid - \mid \ldots \mid \mathsf{Pair} \mid \mathsf{fst} \mid \mathsf{snd} \mid \mathsf{rec\_nat} \mid \mathsf{rec\_list} \mid$$
$$\mathsf{return} \mid \mathsf{bind} \mid \mathsf{query} \mid \mathsf{Uniform} \mid \mathsf{Gauss} \mid \ldots$$

$\vdash \mathsf{return} : X \Rightarrow M[X]$

$\vdash \mathsf{bind} : M[X] \Rightarrow (X \Rightarrow M[Y]) \Rightarrow M[Y]$

$\vdash \mathsf{Uniform} : \mathsf{real} \Rightarrow \mathsf{real} \Rightarrow M[\mathsf{real}]$

$\vdash \mathsf{Gauss} : \mathsf{real} \Rightarrow \mathsf{real} \Rightarrow M[\mathsf{real}]$

Isabelle/HOL Terms as Programs

Semantics

Type T $\Longrightarrow$ Quasi-Borel space T

Typed term $\vdash t : T \Longrightarrow$ `"t ∈ qbs_space T"`

We regard Isabelle/HOL terms as probabilistic programs

Examples (deterministic terms):

$\vdash + :$ real $\Rightarrow$ real $\Rightarrow$ real

`"(+) ∈ ℝQ ⇒Q ℝQ ⇒Q ℝQ"`

$\vdash \lambda x\,.\,[x, x+1, x+2] :$ real $\Rightarrow$ List[real]

`"(λx. [x, x+1, x+2]) ∈ ℝQ ⇒Q list_qbs ℝQ"`

$\vdash \lambda(f, x)\,.\,f\,x :$ (real $\Rightarrow$ real) $\times$ real $\Rightarrow$ real

`"(λ(f,x). f x) ∈ (ℝQ ⇒Q ℝQ) ⨂Q ℝQ ⇒Q ℝQ"`

Isabelle/HOL Terms as Programs

Probabilistic Computations

$[\![M[X]]\!] =$ `monadM_qbs X`

`"return_qbs X ∈ X ⇒Q monadM_qbs X"`

`"(≫=) ∈ monadM_qbs X ⇒Q (X ⇒Q monadM_qbs Y) ⇒Q monadM_qbs Y"`

Example:

`"(Uniform 0 5 ≫= (λx. return_qbs ℝQ (x + 1))) ∈ monadM_qbs ℝQ"`

1. Pick a sample x from $\mathrm{Uniform}(0,5)$
2. Return $x + 1$

Equal to `"Uniform 1 6"`

Conditioning

`"query ∈ monadM_qbs X ⇒Q (X ⇒Q ℝQ≥0) ⇒Q monadM_qbs X "`

$\underbrace{\texttt{monadM\_qbs X}}_{\text{Prior}}$ $\underbrace{\texttt{(X ⇒Q ℝQ≥0)}}_{\text{Likelihood}}$ $\underbrace{\texttt{monadM\_qbs X}}_{\text{Posterior}}$

Automated Type Checking

Type check $\vdash t : T \Longleftrightarrow$ Prove `"t ∈ qbs_space T"`

Automate proofs for *type checking* `"t ∈ qbs_space T"` by tactic **qbs**

- implemented in ML
- similar to *measurability prover* in HOL-Analysis
- The algorithm is based on type inference in functional programming languages

```
lemma
  assumes [qbs]: "f ∈ ℝQ ⇒Q ℝQ"
  shows "(λx. f x + 1) ∈ ℝQ ⇒Q ℝQ"
  by qbs
```

```
lemma "(Uniform 0 5 ⤜ (λx. return_qbs ℝQ (x + 1)))
        ∈ monadM_qbs ℝQ"
  by qbs
```

Measurability Prover VS QBS Prover

Measurability Prover (in HOL-Analysis)

Try to solve measurability

Measurable functions

"$f \in M \rightarrow_M N$"

Measurable sets

"$A \in \mathrm{sets}\ M$"

QBS prover

Try to solve membership

"$t \in \mathrm{qbs\_space}\ X$"

- "$f \in X \rightarrow_Q Y$" $\Longrightarrow$ use "$\mathrm{qbs\_space}\ (X \Rightarrow_Q Y) = X \rightarrow_Q Y$"
- "$\alpha \in \mathrm{qbs\_Mx}\ X$" $\Longrightarrow$ use "$\mathrm{qbs\_Mx}\ X = \mathbb{R}_Q \rightarrow_Q X$"

Conclusion

- Formalizing the s-finite measure monad on quasi-Borel spaces

  $\Longrightarrow$ Higher-order probabilistic programs with conditioning

- Isabelle/HOL terms as programs + Developing proof automation for *type checking*

  $\Longrightarrow$ Easier to write, read, and reason about probabilistic programs

- Implementing several examples of probabilistic programs and proving their properties
  - What time is it?
  - Convergence and stability of the Gaussian mean learning

  etc.

Appendix

The Adjunction

Conversions

$$\begin{tikzcd} \mathbf{QBS} \arrow[r, bend left, "L"] \arrow[r, phantom, "\perp"] & \mathbf{Meas} \arrow[l, bend left, "R"] \end{tikzcd}$$

- `lemma "X →Q R M = L X →M M"`

Morphism Measurable

- If M is a standard Borel space, then $L\ (R\ M) = M$

Morphism iff measurable, when we use only standard Borel spaces

Measures on Quasi-Borel Spaces

Measures on quasi-Borel spaces

- S-finite measures on standard Borel spaces

  Ex: the Lebesgue measure, probability measures on $\mathbb{R}$

- Integration

Quasi-Borel theory　　　　　　　　Measure theory

```
lemma "(∫Q x. f x ∂lborelQ) = (∫x. f x ∂lborel)"
```

The Lebesgue measure on quasi-Borel spaces