Programming the Internet of Uncertain <T>hings

James Bornholt University of Washington

Na Meng University of Texas at Austin

Todd Mytkowicz Microsoft Research

Kathryn S. McKinley Microsoft Research

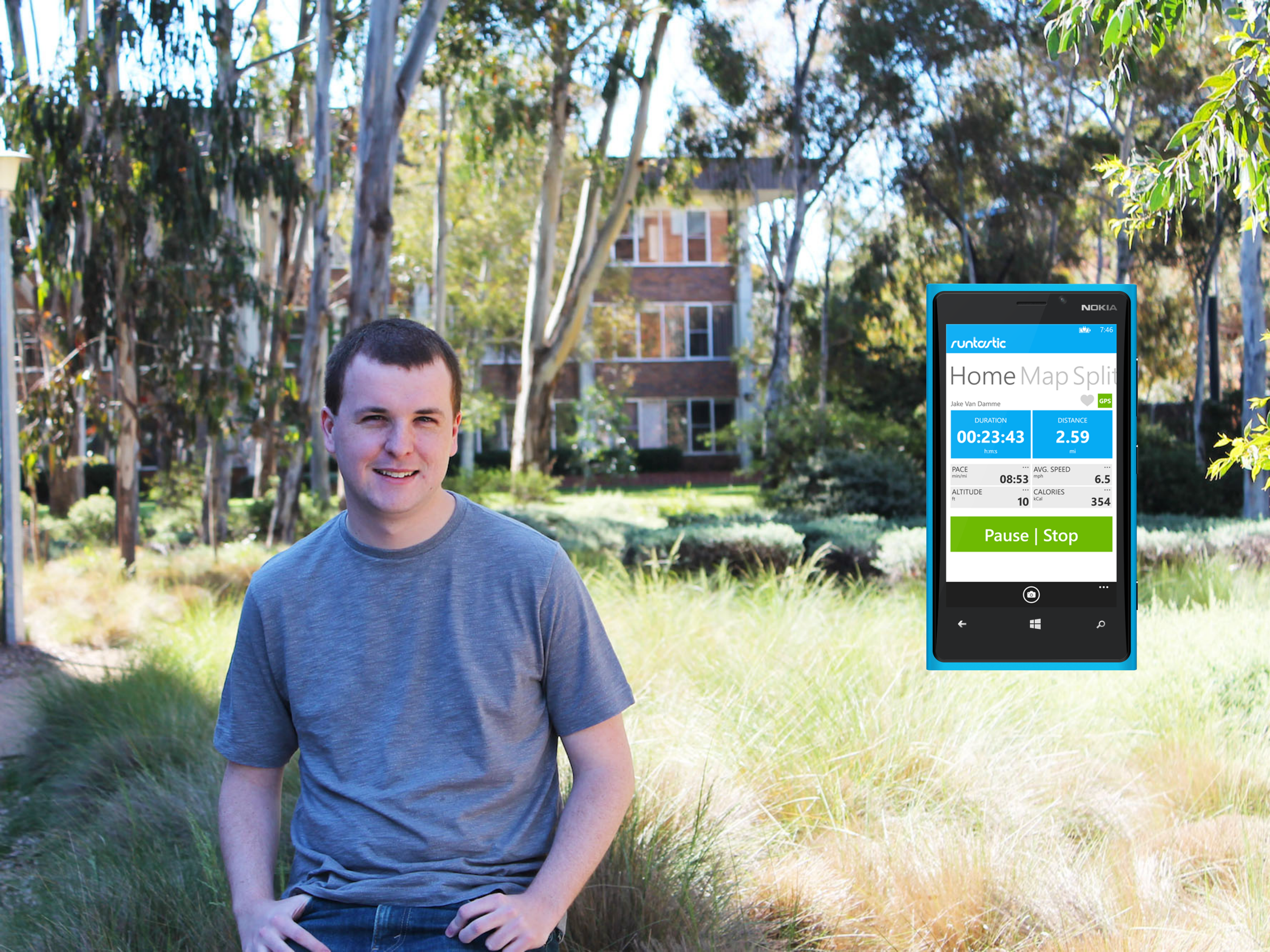

Sainsbury's
BOLT

24 mph

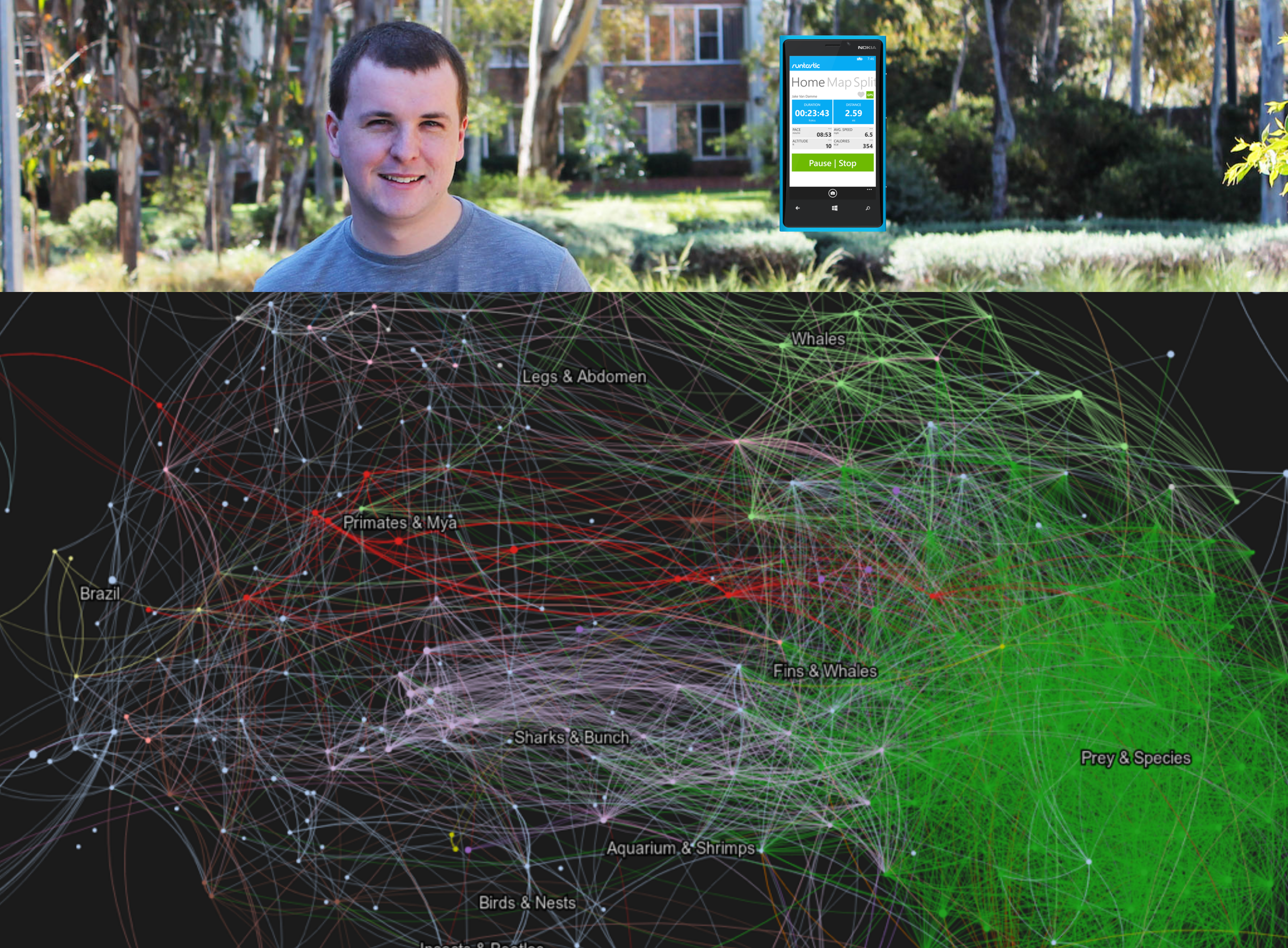
```
GeoCoordinate PrevLocn = Get();
Sleep(5);
GeoCoordinate Location = Get();
```

```
GeoCoordinate PrevLocn = Get();
Sleep(5);
GeoCoordinate Location = Get();
double Dist =
                  Distance(LastLocn, Location);
double Speed = Dist / 5;
```

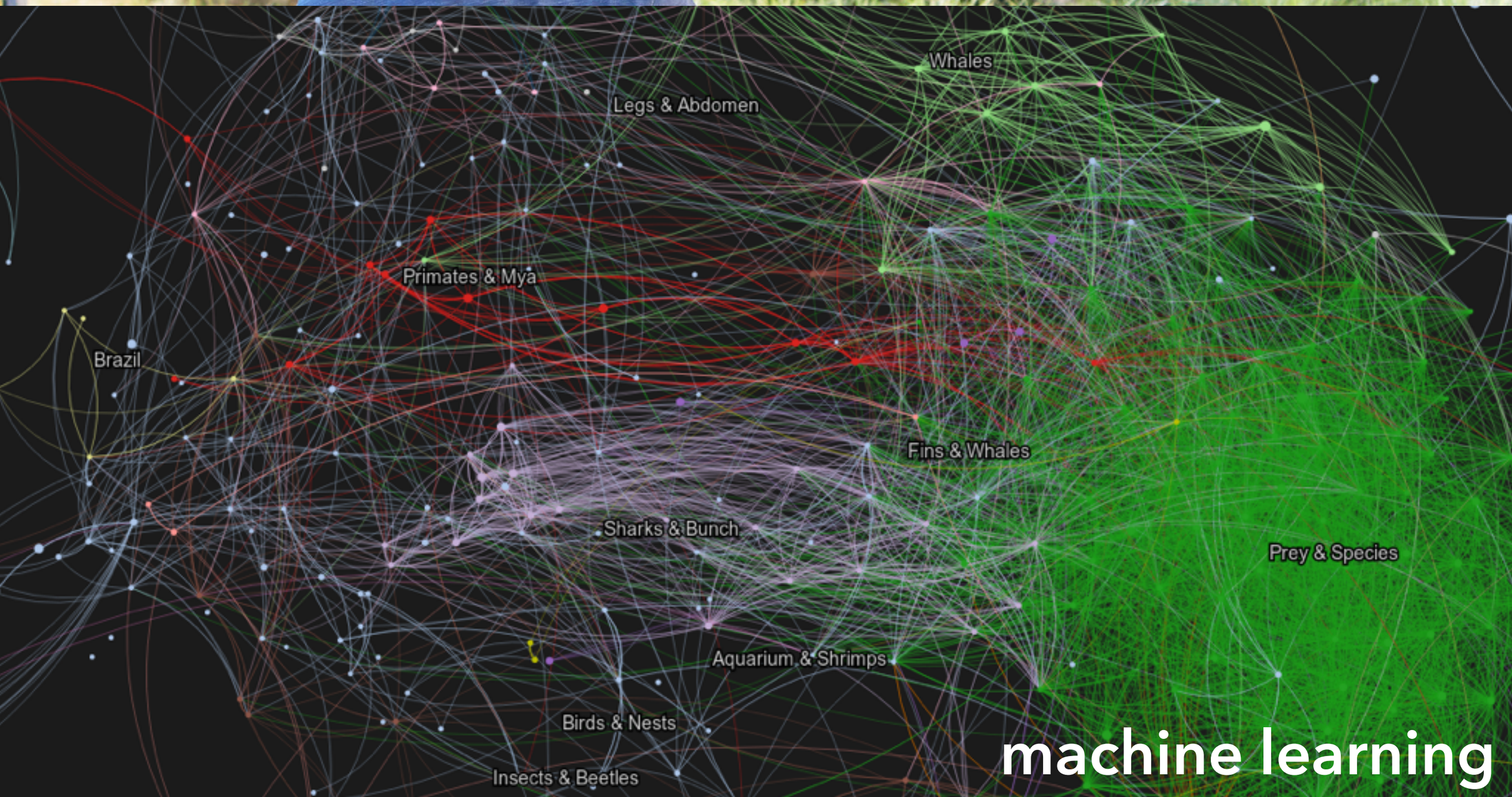
```
GeoCoordinate PrevLocn = Get();
Sleep(5);
GeoCoordinate Location = Get();
double Dist =
                  Distance(LastLocn, Location);
double Speed = Dist / 5;

if (Speed > 4)
    Alert("Keep it up!");
```


NOKIA
7:46
runtastic
Home Map Split
Jake Van Damme
GPS
DURATION
00:23:43
h:m:s
DISTANCE
2.59
mi
PACE
min/mi
08:53
AVG. SPEED
mph
6.5
ALTITUDE
ft
10
CALORIES
kCal
354
Pause | Stop


NOKIA
7:46
runtastic
Home Map Split
Jake Van Damme
GPS
DURATION
00:23:43
h:m:s
DISTANCE
2.59
mi
PACE
min/mi
08:53
AVG. SPEED
mph
6.5
ALTITUDE
ft
10
CALORIES
kCal
354
Pause | Stop
59 mph

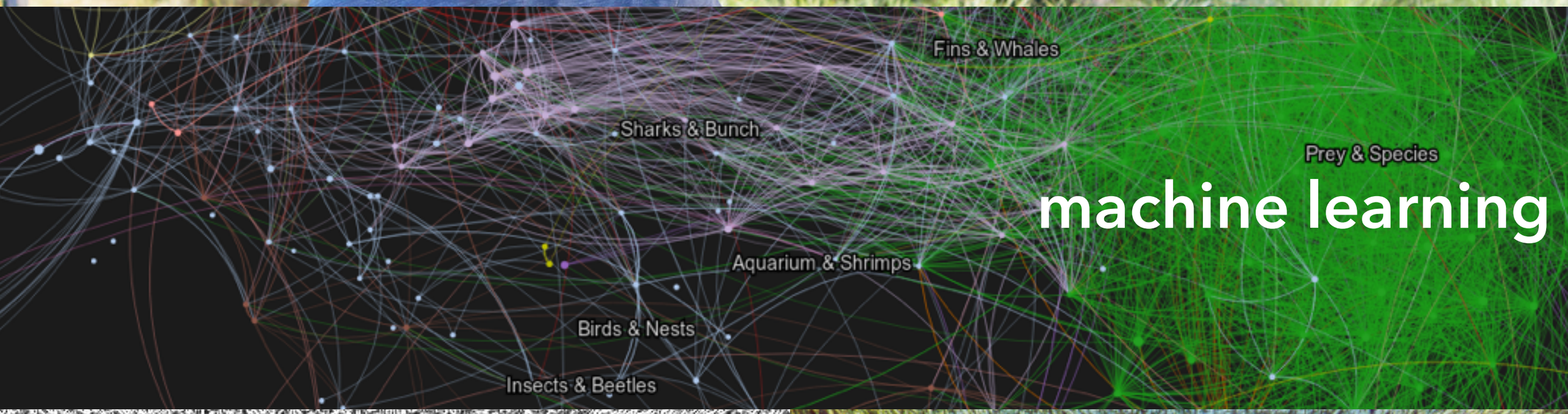

NOKIA
runtastic
Home Map Split
Jake Van Damme
GPS
DURATION
00:23:43
DISTANCE
2.59
PACE
08:53
AVG. SPEED
6.5
ALTITUDE
10
CALORIES
354
Pause | Stop
Whales
Legs & Abdomen
Primates & Mya
Brazil
Fins & Whales
Sharks & Bunch
Prey & Species
Aquarium & Shrimps
Birds & Nests
Insects & Beetles

sensors



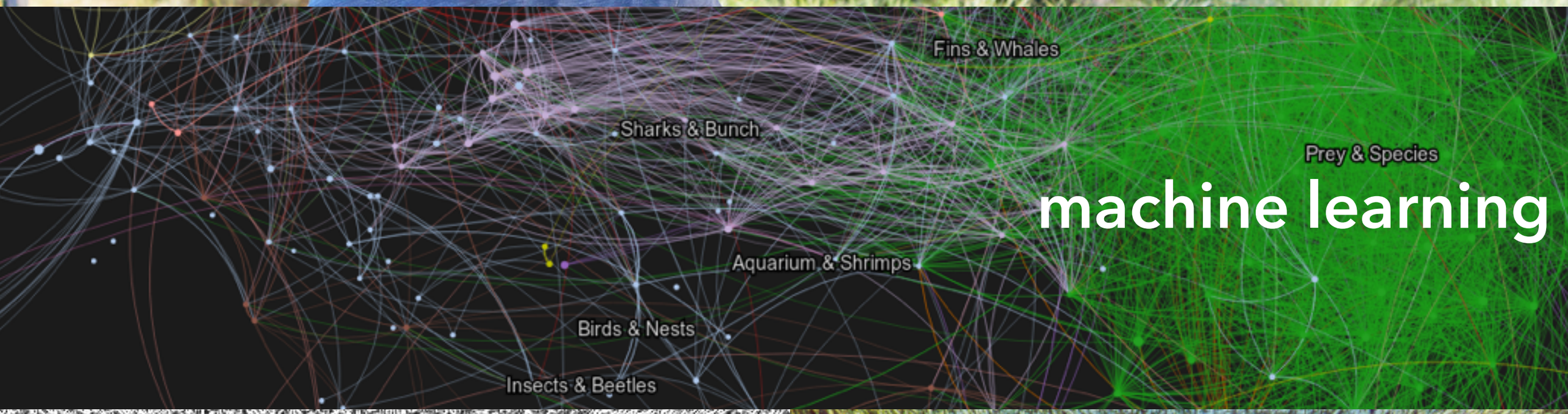
machine learning

sensors



machine learning

sensors



machine learning

approximate computing

Uncertain<*T*>

an abstraction for reasoning about noise [ASPLOS'14]

exploiting context

language constructs to make data more accurate

Uncertain<*T*>

an abstraction for reasoning about noise [ASPLOS'14]

exploiting context

language constructs to make data more accurate

```
GeoCoordinate PrevLocn = Get();
Sleep(5);
GeoCoordinate Location = Get();
double Dist =
                  Distance(LastLocn, Location);
double Speed = Dist / 5;

if (Speed > 4)
    Alert("Keep it up!");
```

```
Uncertain<GeoCoordinate> PrevLocn = Get();
Sleep(5);
Uncertain<GeoCoordinate> Location = Get();
Uncertain<double> Dist =
                    Distance(LastLocn, Location);
Uncertain<double> Speed = Dist / 5;

if (Speed > 4)
    Alert("Keep it up!");
```

```
Uncertain<GeoCoordinate> PrevLocn = Get();
Sleep(5);
Uncertain<GeoCoordinate> Location = Get();
Uncertain<double> Dist =
                    Distance(LastLocn, Location);
Uncertain<double> Speed = Dist / 5;

if (Speed > 4)
     Alert("Keep it up!");
```

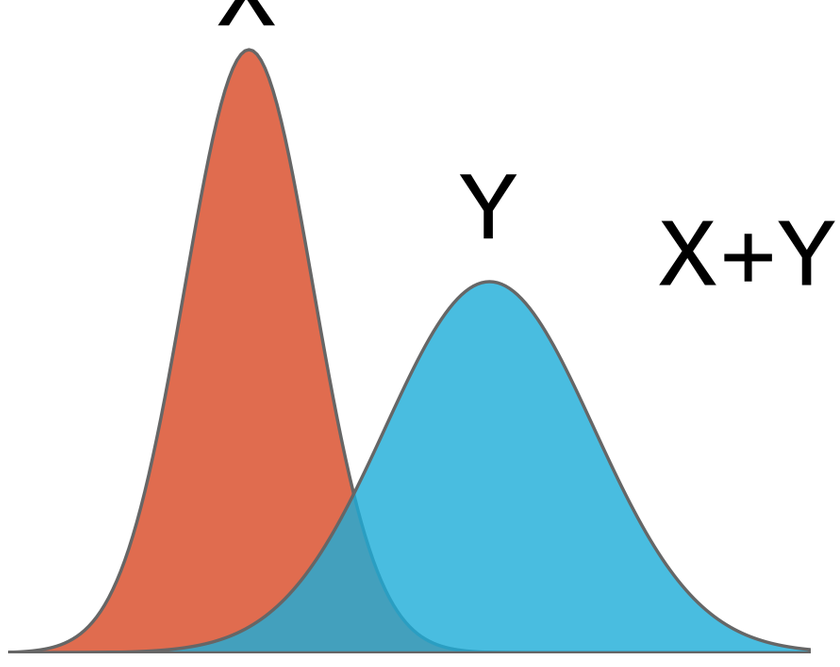
Semantics

Uncertain<T> encapsulates probability distributions and hides statistical complexity.

- Computing over random variables
- Deciding conditionals

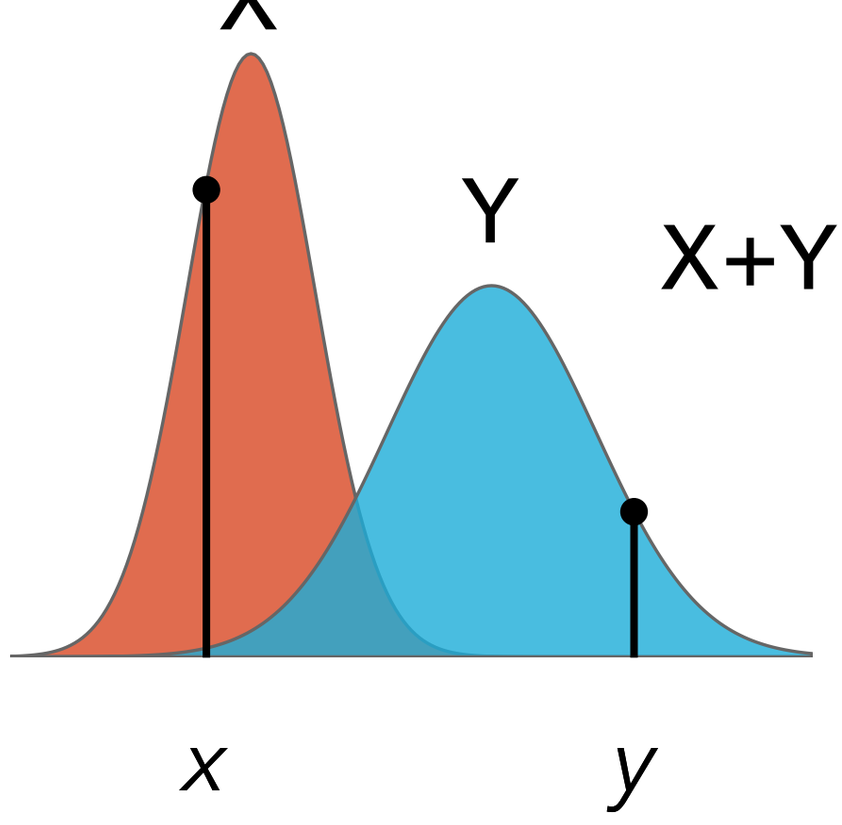
Computations

Represent distributions by random samples



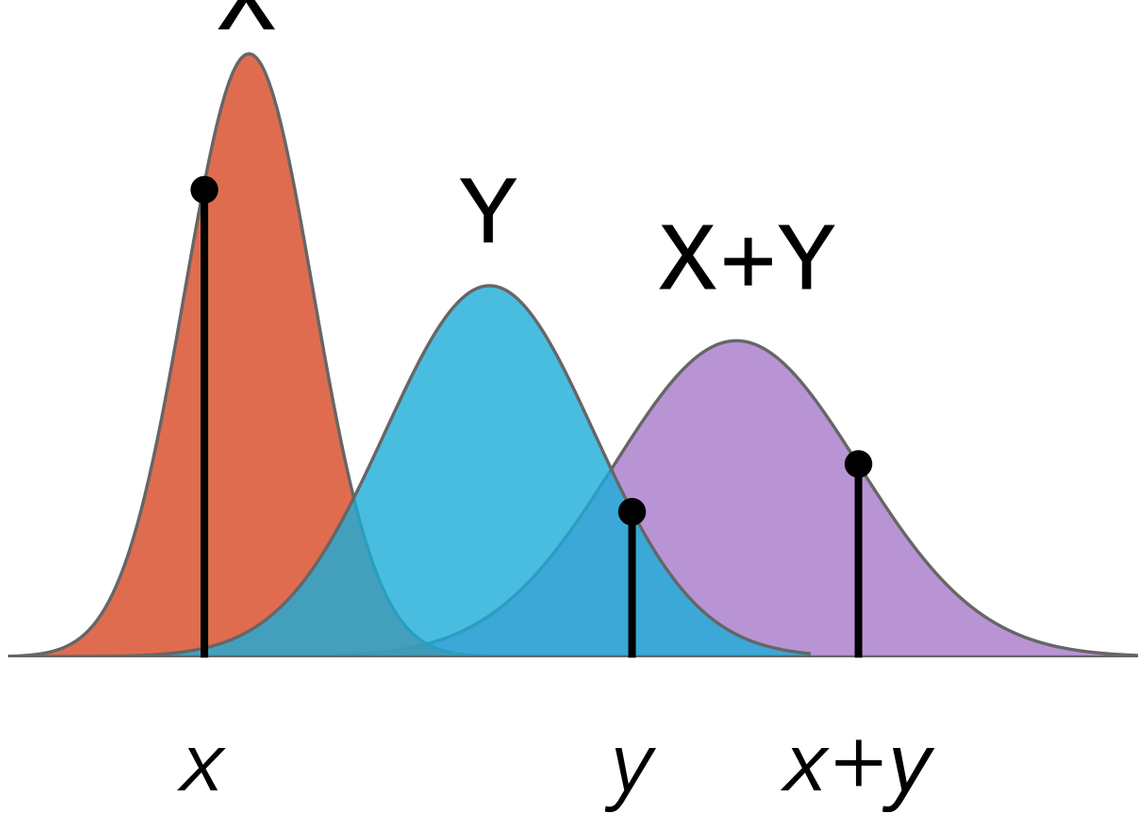
Computations

Represent distributions by random samples



Computations

Represent distributions by random samples



Computations

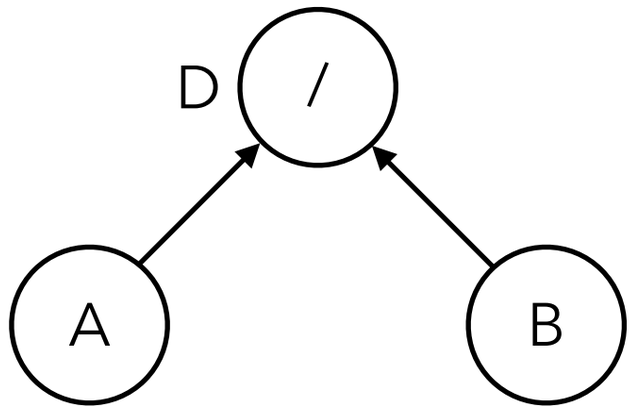
Operators build a Bayesian network rather than evaluating immediately.

```
D = A / B
```

Computations

Operators build a Bayesian network rather than evaluating immediately.

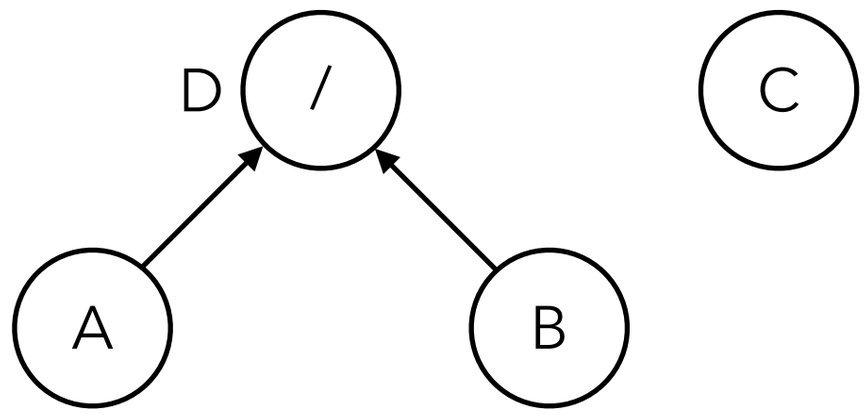
```
D = A / B
```



Computations

Operators build a Bayesian network rather than evaluating immediately.

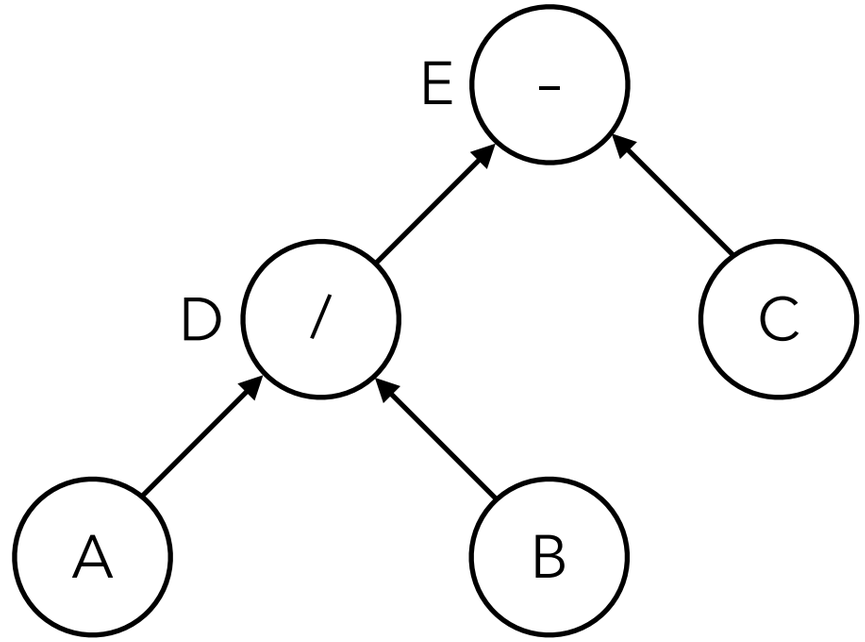
```
D = A / B
E = D - C
```



Computations

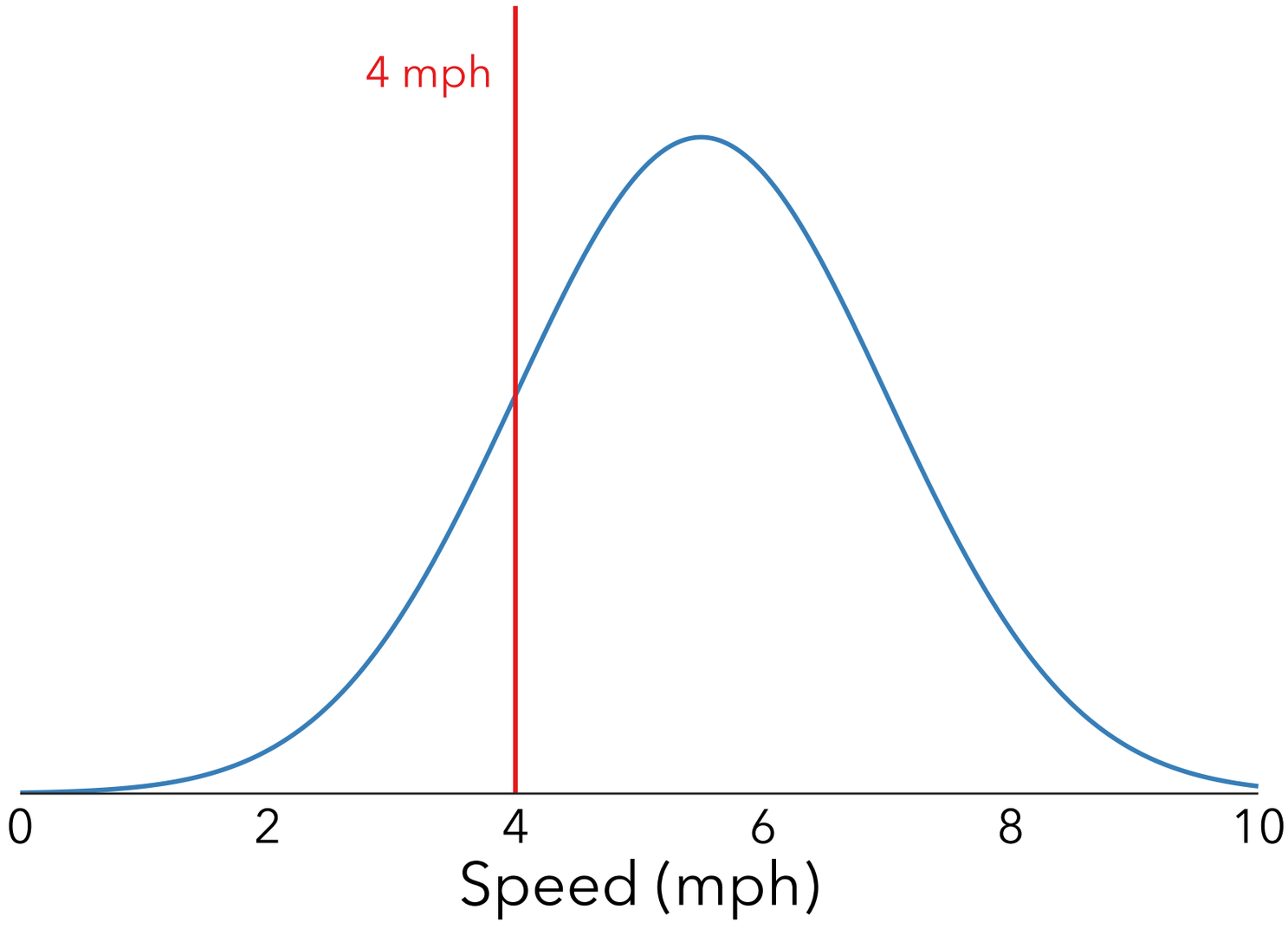
Operators build a Bayesian network rather than evaluating immediately.

```
D = A / B
E = D - C
```



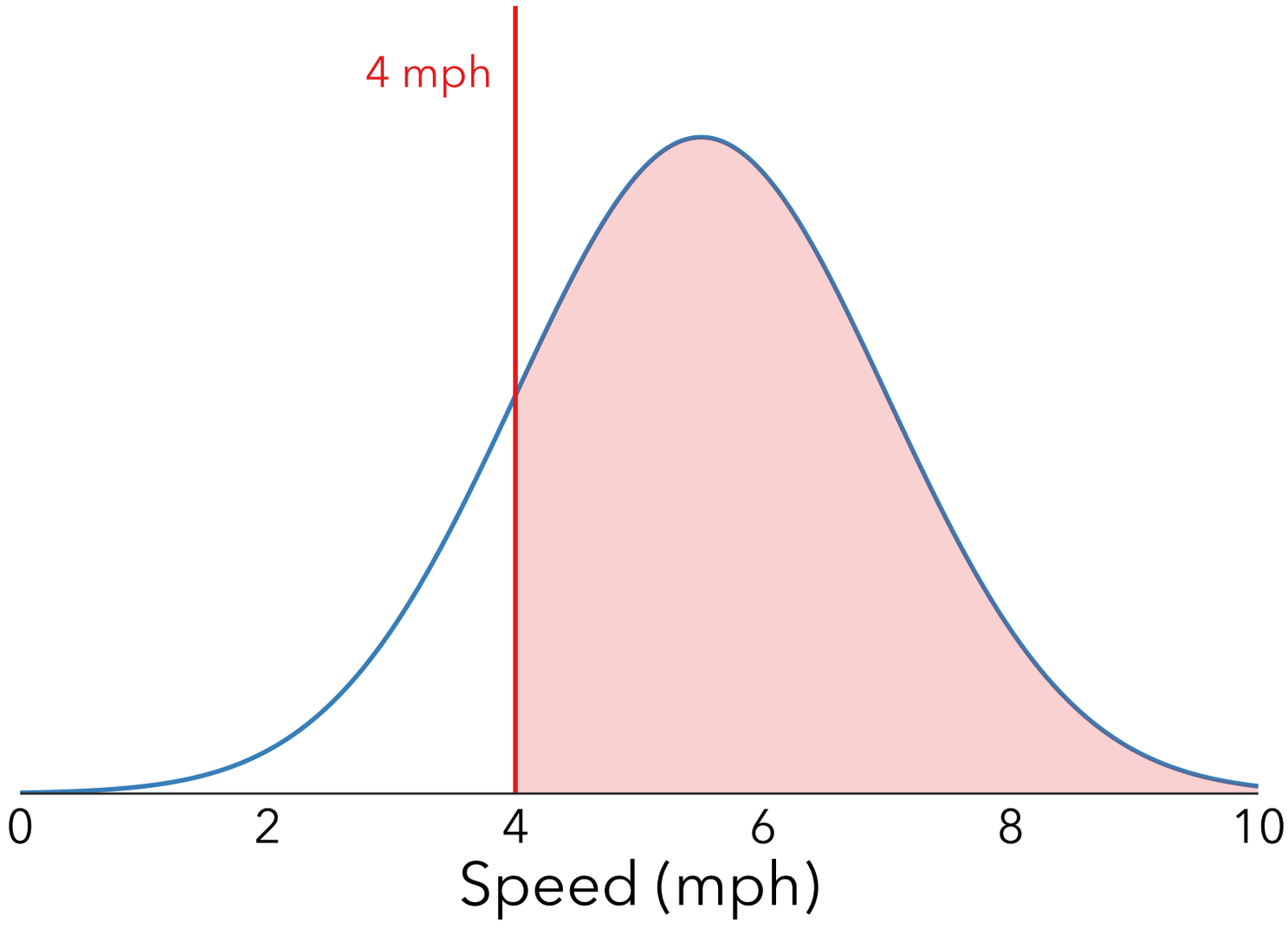
Deciding conditionals

```
if (Speed > 4)
     Alert("Keep it up!");
```



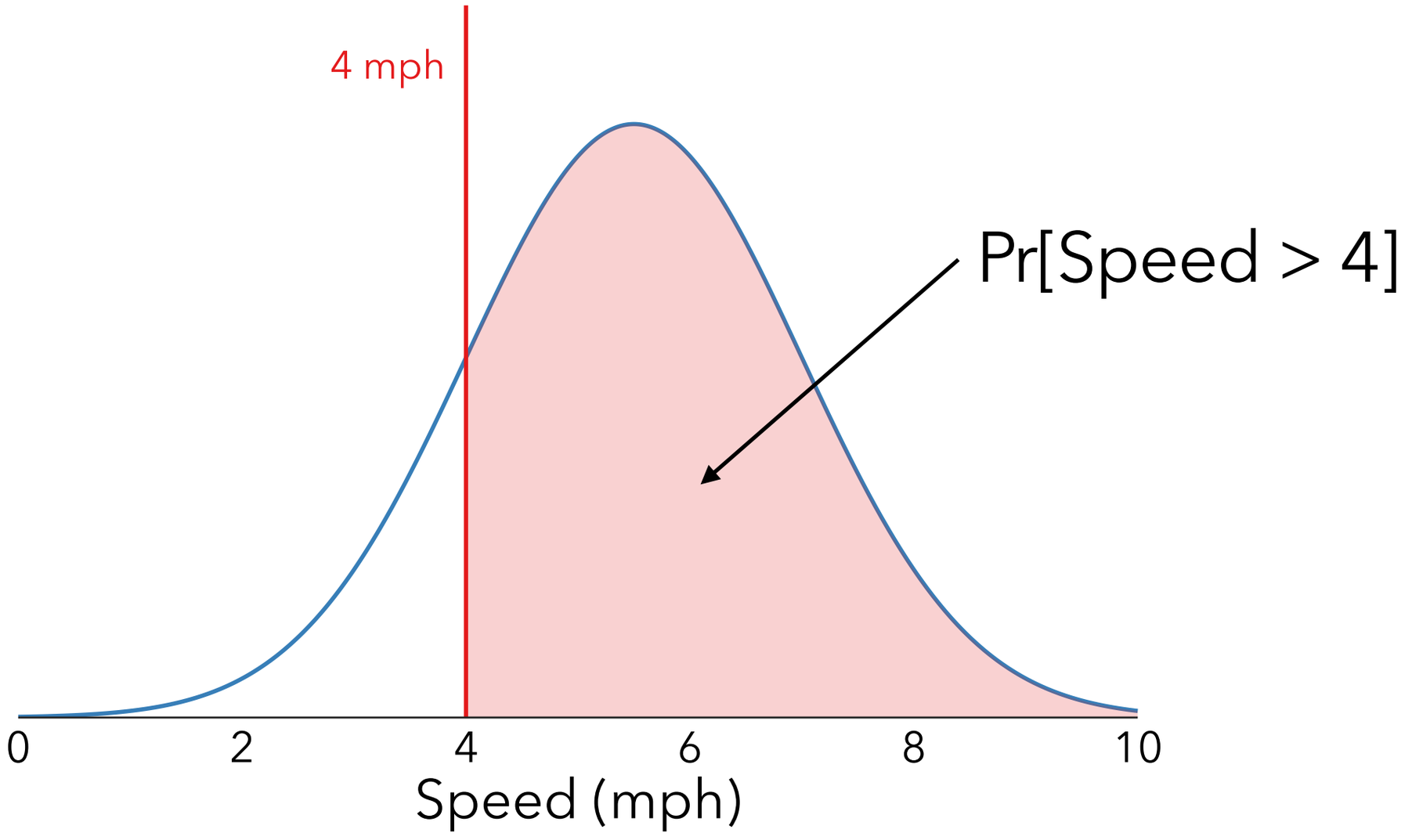
Deciding conditionals

```
if (Speed > 4)
     Alert("Keep it up!");
```



Deciding conditionals

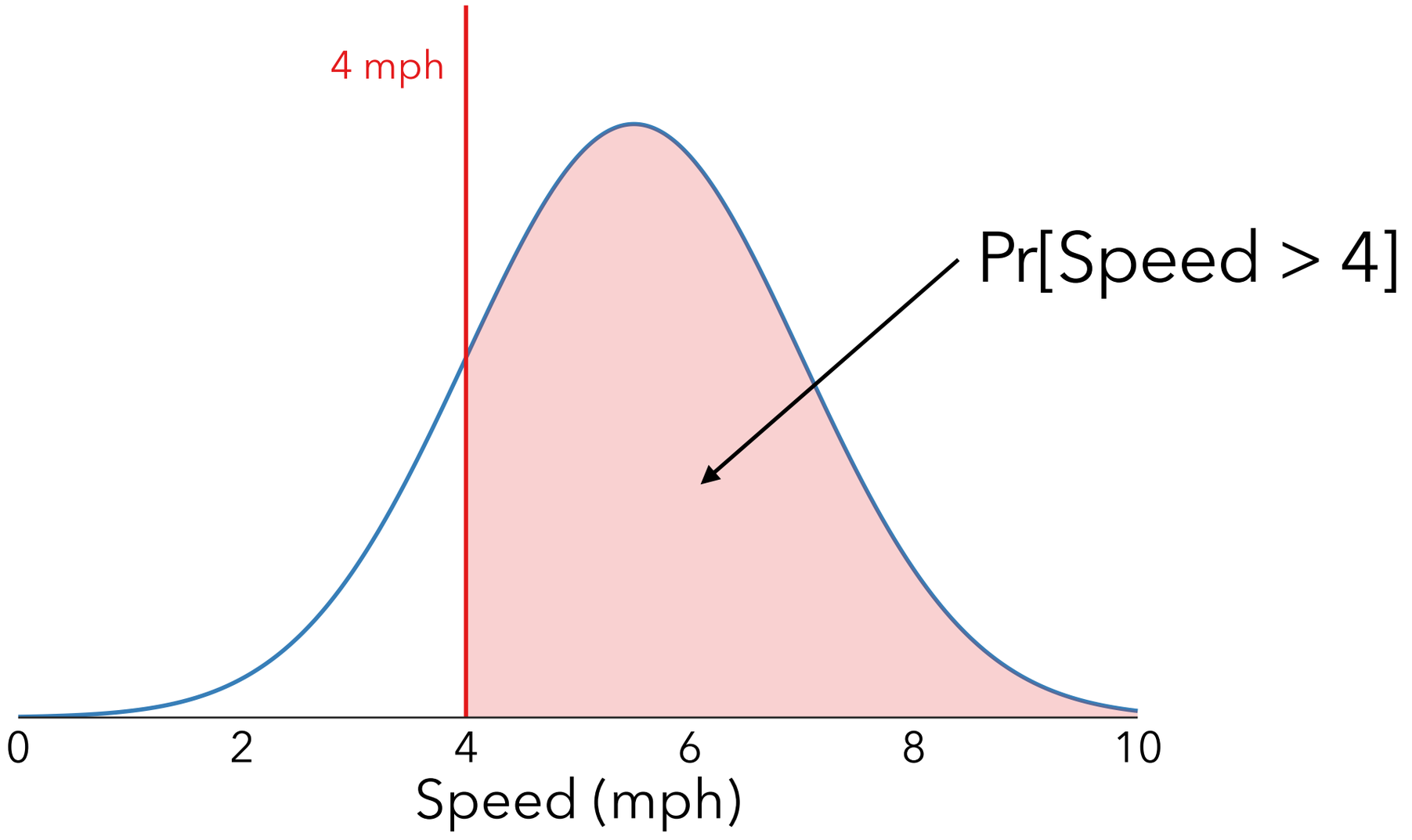
```
if (Speed > 4)
     Alert("Keep it up!");
```



Deciding conditionals

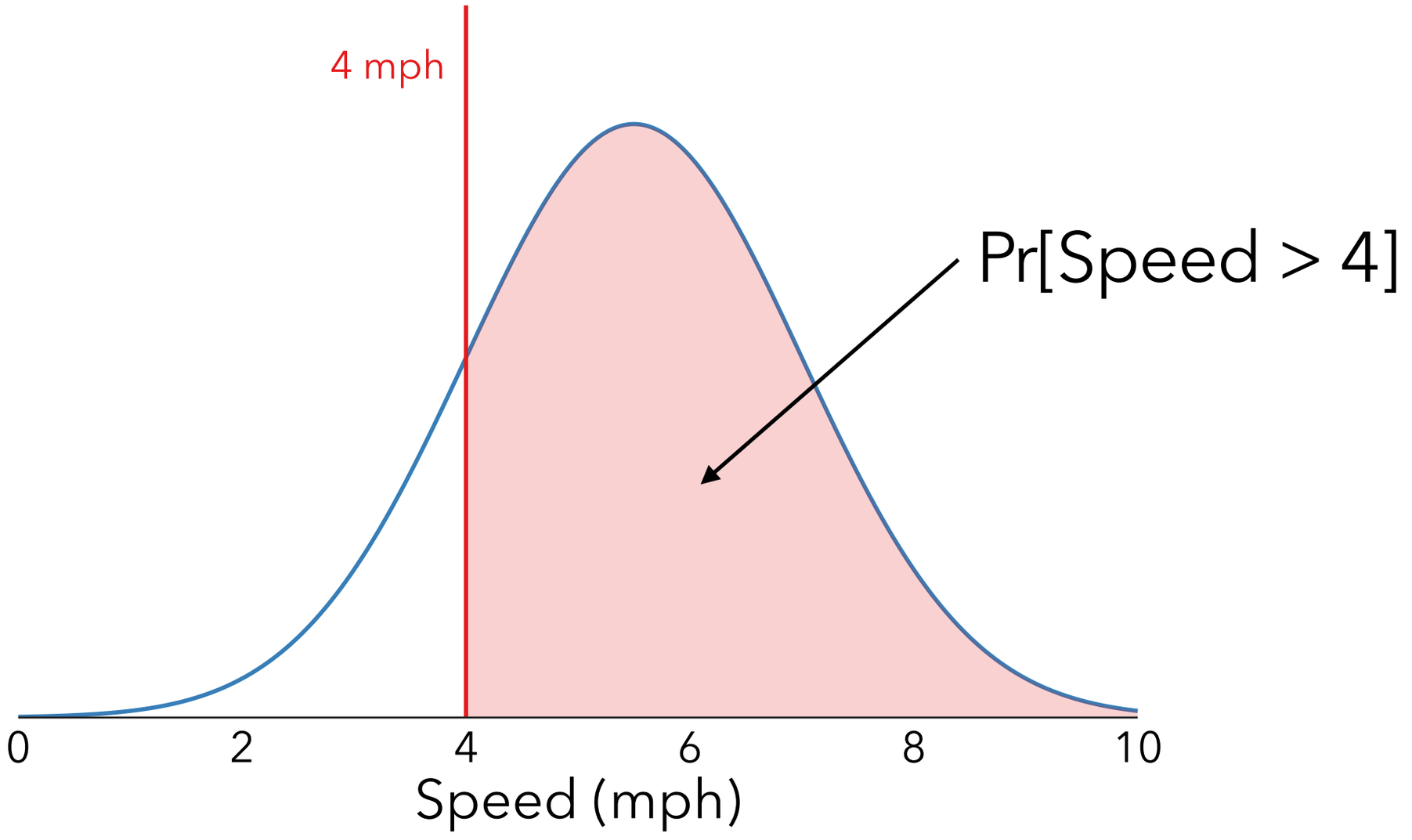
```
if (Speed > 4)
     Alert("Keep it up!");
```

More likely than not that Speed > 4?



Deciding conditionals

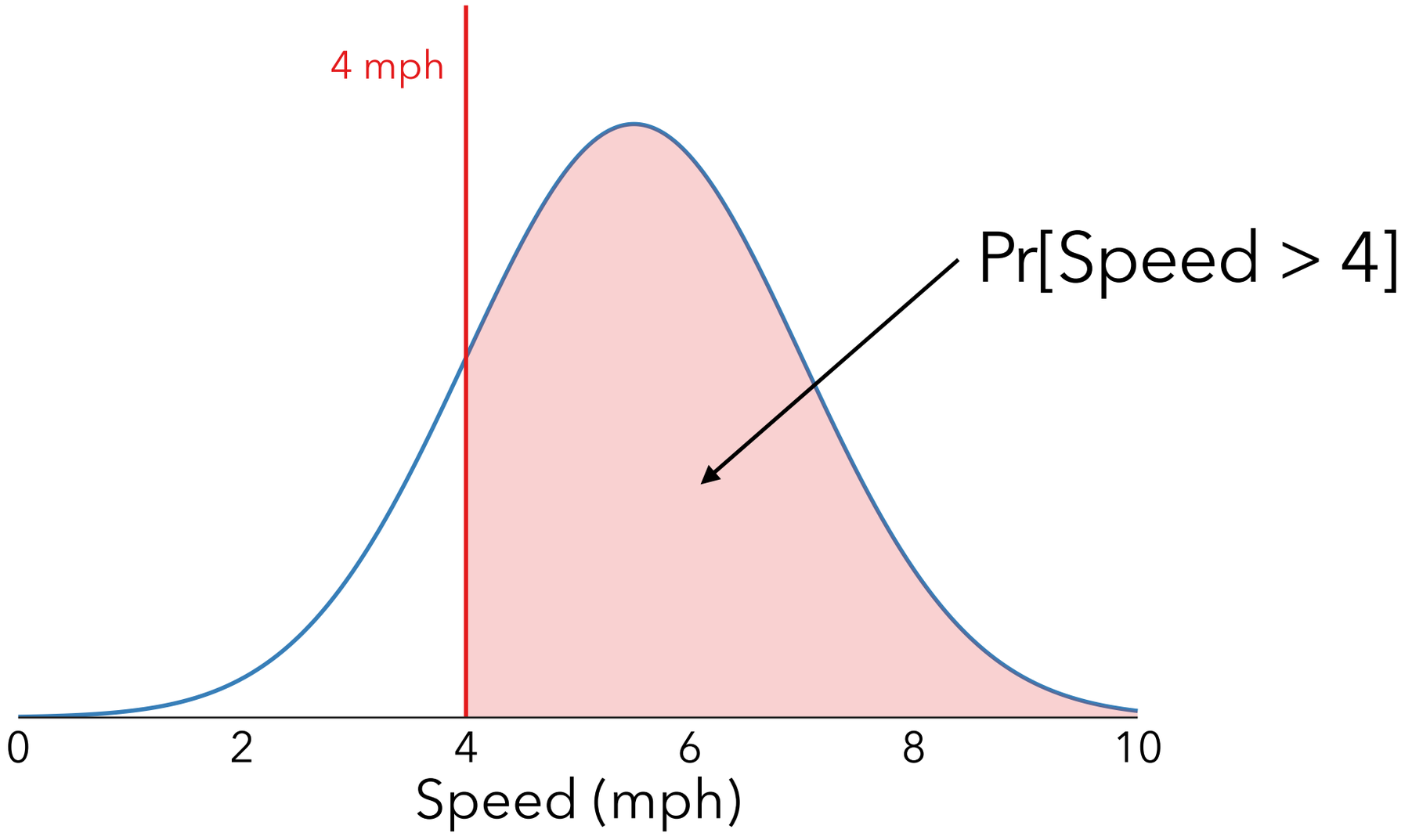
```
if ((Speed > 4).Pr(0.9))
    Alert("Keep it up!");
```



Deciding conditionals

```
if ((Speed > 4).Pr(0.9))
    Alert("Keep it up!");
```

At least 90% likely that Speed > 4?



Identifying absurd data

```
GeoCoordinate PrevLocn = Get();
Sleep(5);
GeoCoordinate Location = Get();
double Dist =
                  Distance(LastLocn, Location);
double Speed = Dist / 5;

if (Speed > 4)  // 7 mph
    Alert("That's crazy!");
```

Identifying absurd data

```
GeoCoordinate PrevLocn = Get();
Sleep(5);
GeoCoordinate Location = Get();
double Dist =
                Distance(LastLocn, Location);
double Speed = Dist / 5;

if (Speed > 4)  // 7 mph
    Alert("That's crazy!");
```

Naive: 30 times

Identifying absurd data

```
Uncertain<GeoCoordinate> PrevLocn = Get();
Sleep(5);
Uncertain<GeoCoordinate> Location = Get();
Uncertain<double> Dist =
                    Distance(LastLocn, Location);
Uncertain<double> Speed = Dist / 5;

if (Speed > 4)  // 7 mph
    Alert("That's crazy!");
```

Naive: 30 times

Identifying absurd data

```
Uncertain<GeoCoordinate> PrevLocn = Get();
Sleep(5);
Uncertain<GeoCoordinate> Location = Get();
Uncertain<double> Dist =
                  Distance(LastLocn, Location);
Uncertain<double> Speed = Dist / 5;

if (Speed > 4)  // 7 mph
    Alert("That's crazy!");
```

Naive: 30 times

50%: 4 times

Identifying absurd data

```
Uncertain<GeoCoordinate> PrevLocn = Get();
Sleep(5);
Uncertain<GeoCoordinate> Location = Get();
Uncertain<double> Dist =
                  Distance(LastLocn, Location);
Uncertain<double> Speed = Dist / 5;

if ((Speed > 4).Pr(0.9))
    Alert("That's crazy!");
```

Naive: 30 times

50%: 4 times

Identifying absurd data

```
Uncertain<GeoCoordinate> PrevLocn = Get();
Sleep(5);
Uncertain<GeoCoordinate> Location = Get();
Uncertain<double> Dist =
                  Distance(LastLocn, Location);
Uncertain<double> Speed = Dist / 5;

if ((Speed > 4).Pr(0.9))
    Alert("That's crazy!");
```

Naive: 30 times

50%: 4 times

90%: never

Uncertain<*T*>

an abstraction for reasoning about noise [ASPLOS'14]

exploiting context

language constructs to make data more accurate

Uncertain<

an abstraction for reasoning about noise

exploiting context

language constructs to make data more accurate


2013β


2013β


XBOX 360
KINECT


```
if (RecognizeBeard(photo))
    AddBeardToAvatar();
```

```
if (RecognizeBeard(photo))
    AddBeardToAvatar();
```

```
City userCity = ...;

if (RecognizeBeard(photo))
    AddBeardToAvatar();
```

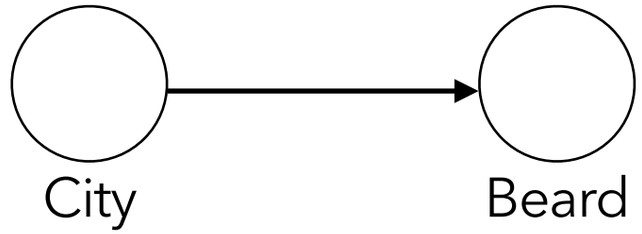
```
City userCity = ...;

if (RecognizeBeard(photo))
    AddBeardToAvatar();
```

context

application-specific

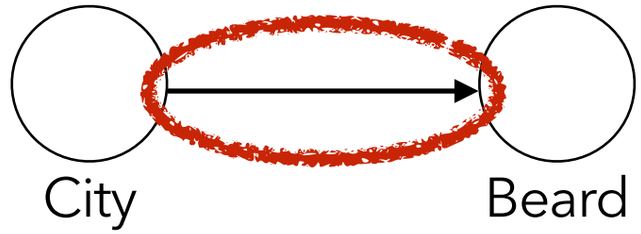
domain knowledge

Static context



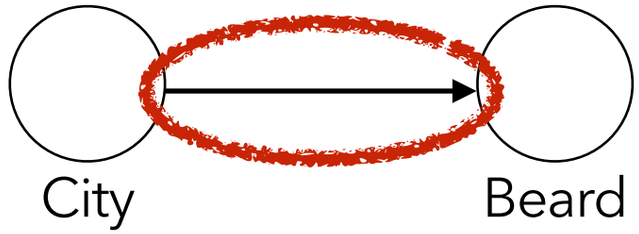
- How does city influence beards?

Static context



- How does city influence beards?

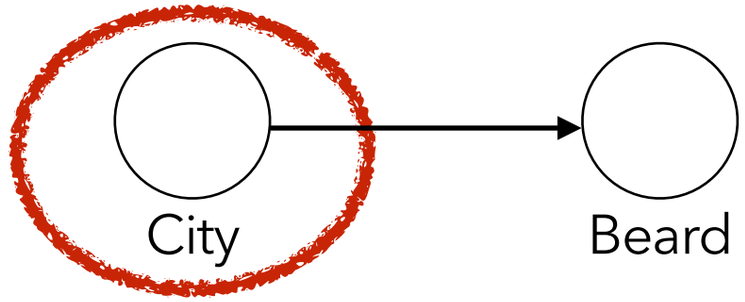
Static context



- How does city influence beards?

```
Bernoulli HasBeard_City(City c) {
    if (c == "Seattle")
        return new Bernoulli(0.4);
    else
        return new Bernoulli(0.2);
}
```

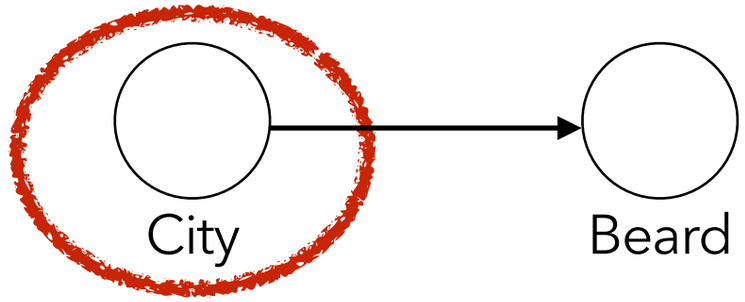
Static context



- How does city influence beards?

```
Bernoulli HasBeard_City(City c) {
    if (c == "Seattle")
        return new Bernoulli(0.4);
    else
        return new Bernoulli(0.2);
}
```

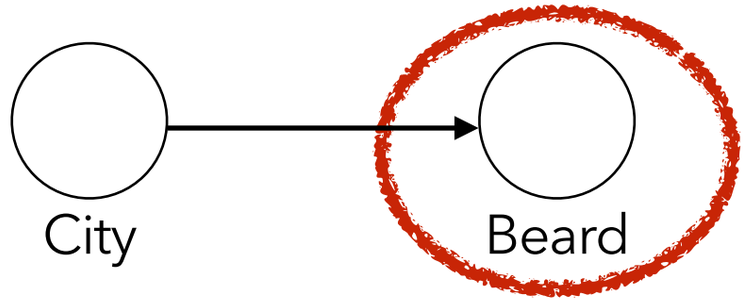
Static context



- How does city influence beards?

```
Bernoulli HasBeard_City(City c) {
    if (c == "Seattle")
        return new Bernoulli(0.4);
    else
        return new Bernoulli(0.2);
}

var Cities = Uniform(...);
```

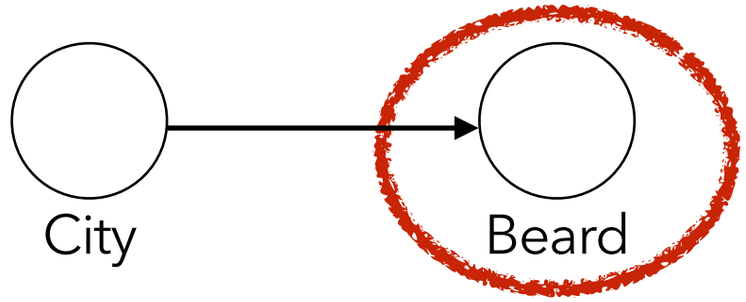
Static context



- How does city influence beards?

```
Bernoulli HasBeard_City(City c) {
    if (c == "Seattle")
        return new Bernoulli(0.4);
    else
        return new Bernoulli(0.2);
}

var Cities = Uniform(...);
```

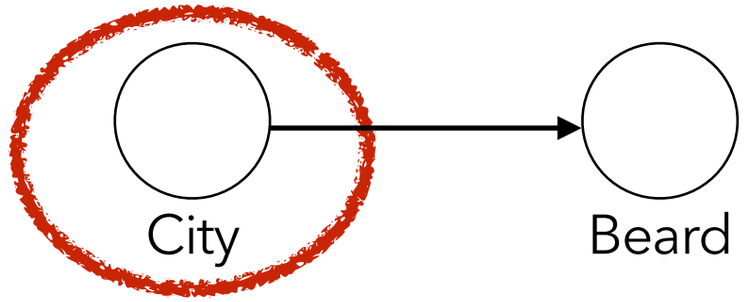
Static context



- How does city influence beards?

```
Bernoulli HasBeard_City(City c) {
    if (c == "Seattle")
        return new Bernoulli(0.4);
    else
        return new Bernoulli(0.2);
}

var Cities = Uniform(...);

Bernoulli HasBeard =
    HasBeard_City <| Cities;
```

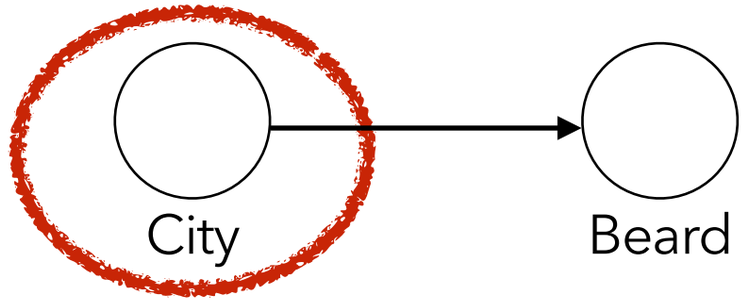
Dynamic context



- Exploit knowledge about *this user*

```
var Cities = Uniform(...);
Bernoulli HasBeard =
    HasBeard_City <| Cities;
```

Dynamic context

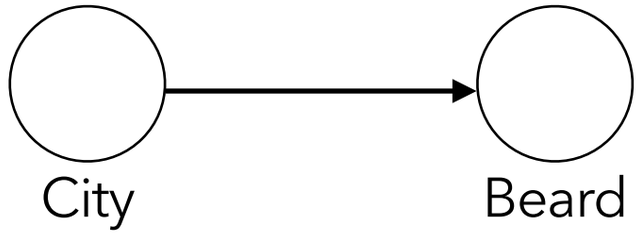


- Exploit knowledge about *this user*

```
var Cities = Uniform(...);
Bernoulli HasBeard =
     HasBeard_City <| Cities;
```

```
Cities.Value = "Seattle";
```

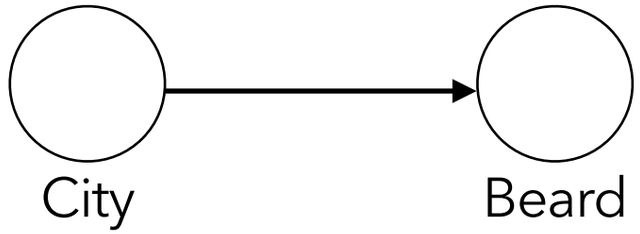
Dynamic context



- Exploit knowledge about *this user*

```
var Cities = Uniform(...);
Bernoulli HasBeard =
    HasBeard_City <| Cities;

Cities.Value = "Seattle";

Bernoulli oldHasBeard = BeardRecognizer(photo);
```

Dynamic context

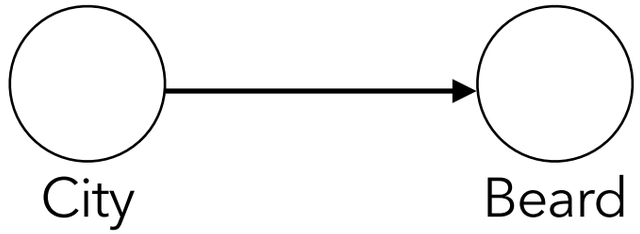


- Exploit knowledge about *this user*

```
var Cities = Uniform(...);
Bernoulli HasBeard =
    HasBeard_City <| Cities;
```

```
Cities.Value = "Seattle";

Bernoulli oldHasBeard = BeardRecognizer(photo);

Bernoulli newHasBeard = oldHasBeard # HasBeard;
```

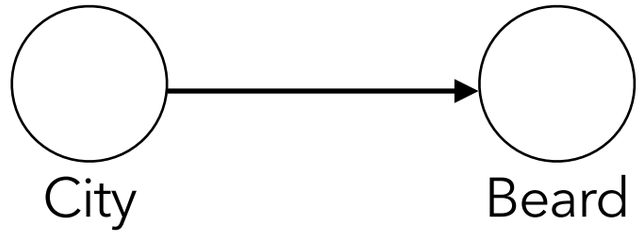
Dynamic context



- Exploit knowledge about *this user*

```
var Cities = Uniform(...);
Bernoulli HasBeard =
     HasBeard_City <| Cities;

Cities.Value = "Seattle";

Bernoulli oldHasBeard = BeardRecognizer(photo);

Bernoulli newHasBeard = oldHasBeard # HasBeard;

if (newHasBeard)
    AddBeardToAvatar();
```

Dynamic context



- Exploit knowledge about *this user*

```
var Cities = Uniform(...);
Bernoulli HasBeard =
     HasBeard_City <| Cities;
```

```
Cities.Value = "Seattle";

Bernoulli oldHasBeard = BeardRecognizer(photo);

Bernoulli newHasBeard = oldHasBeard # HasBeard;

if (newHasBeard.Pr(0.9))
    AddBeardToAvatar();
```

Two new constructs

<|

Building probability distributions (conditional probability)

#

Composing context and estimates (Bayesian inference)

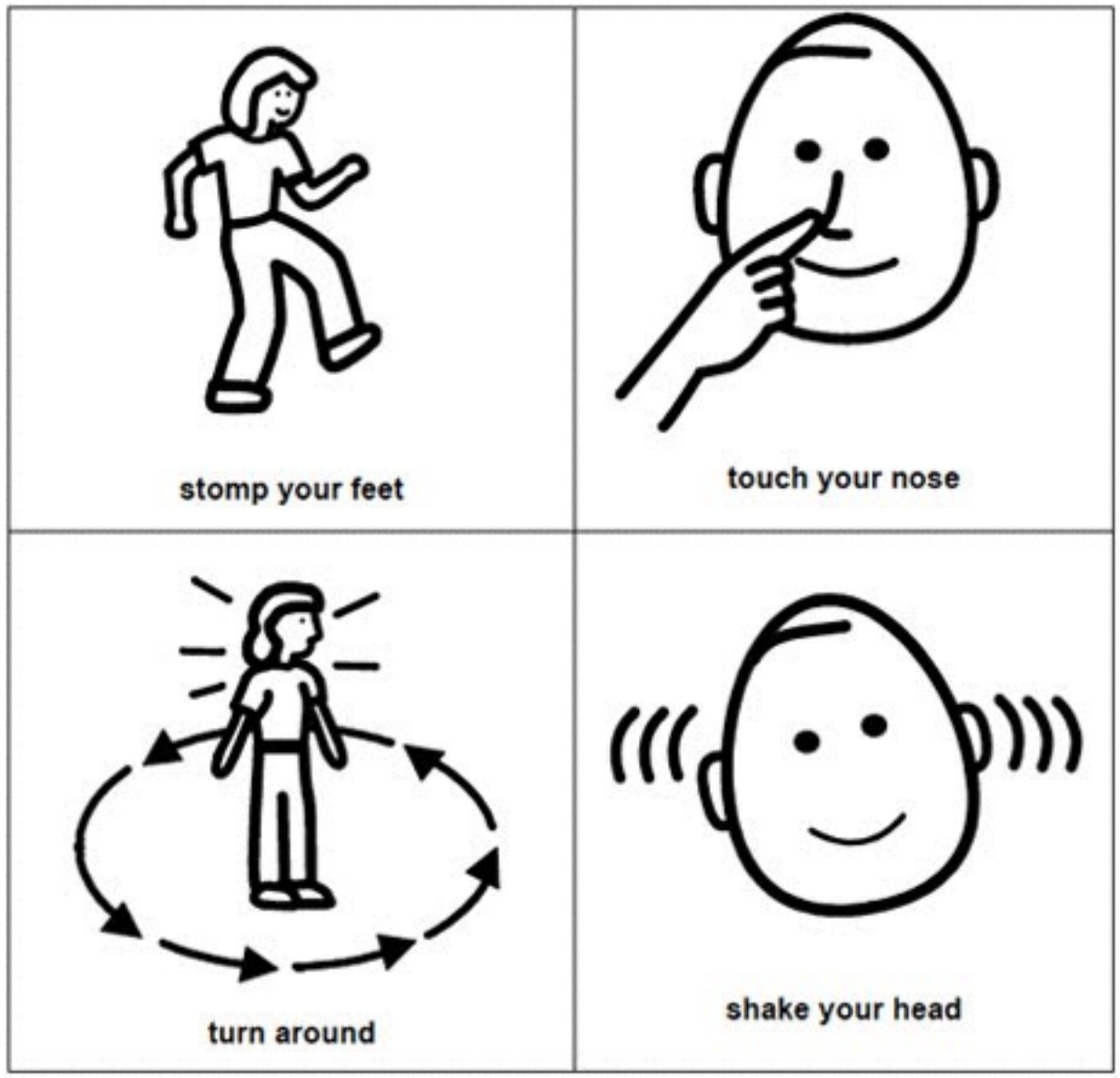
Implementation

- We designed a *sequential likelihood reweighting* algorithm to implement this abstraction
- But we'd like to compile down to probabilistic programming languages, which have better inference

figaro

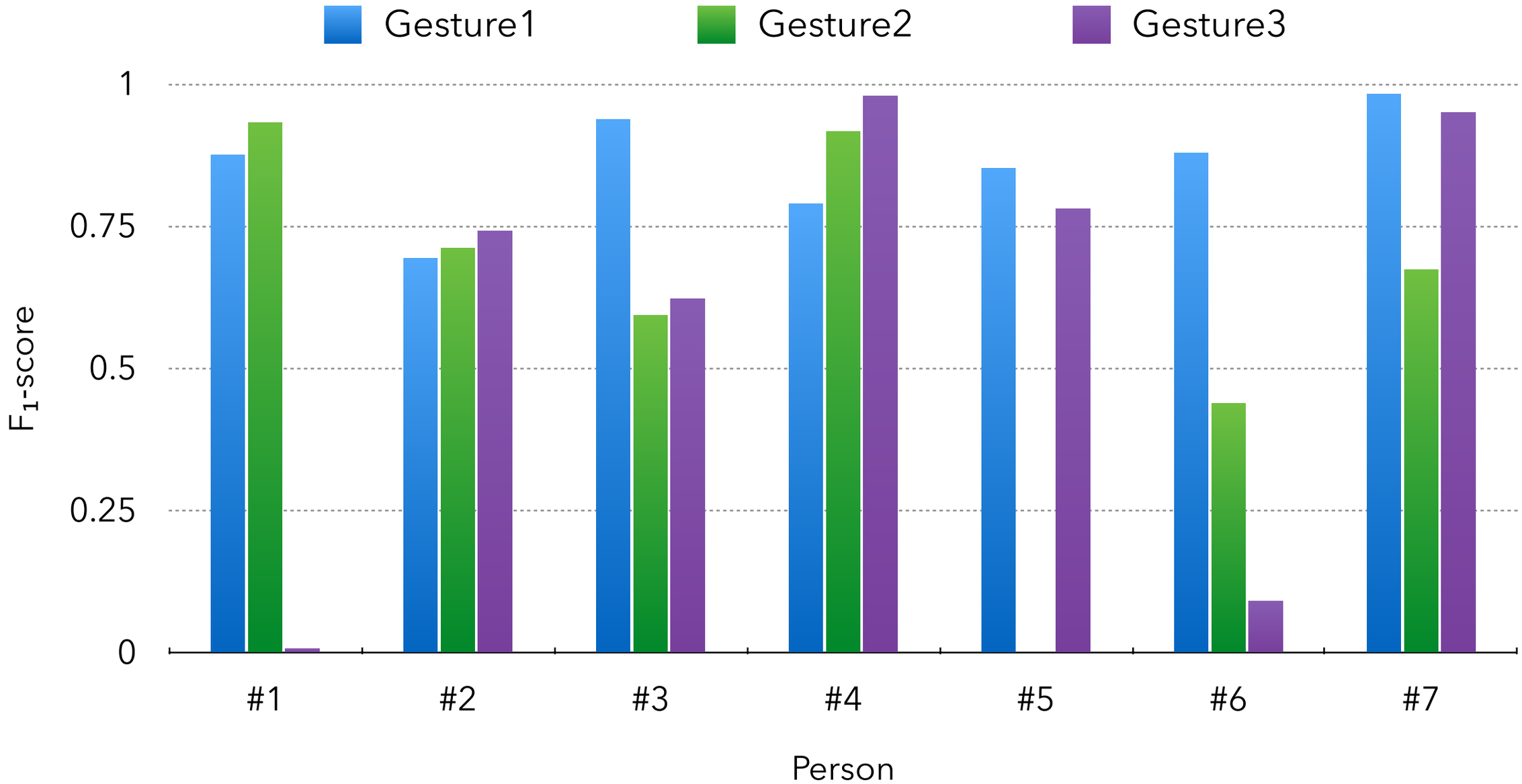
STAN

Simon Says



Simon Says

- Xbox Kinect gesture recognition API



Simon Says

Local model

- Specific to our user
- Need lots of examples to avoid noise

Simon Says

Local model

- Specific to our user
- Need lots of examples to avoid noise

Global (API) model

- Trained over many users
- Generalises well

Simon Says

Local model

- Specific to our user
- Need lots of examples to avoid noise

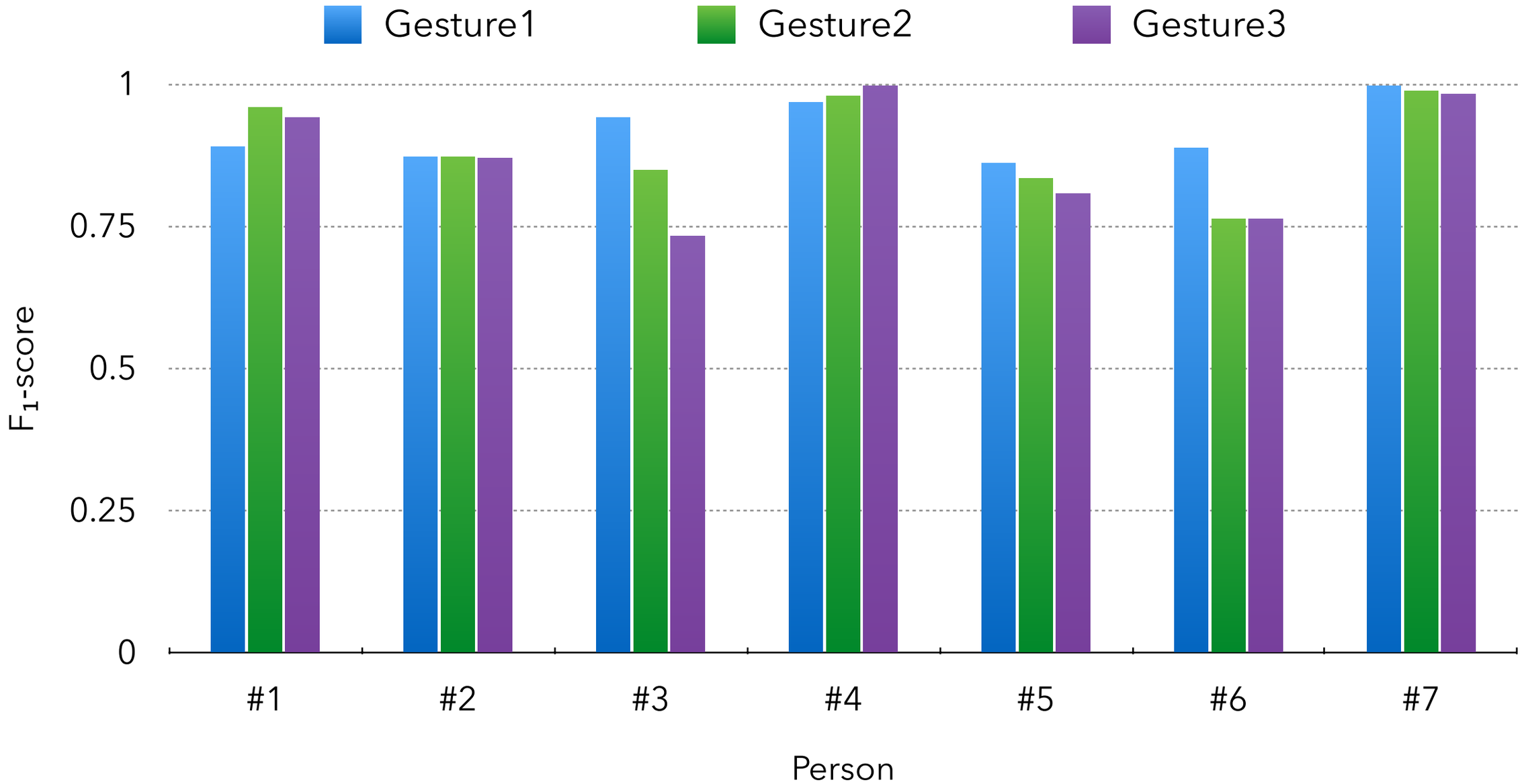
Global (API) model

- Trained over many users
- Generalises well

We’d like to keep both!

Simon Says

- Personalised gesture recognition model



Uncertain<*T*>

an abstraction for reasoning about noise [ASPLOS'14]

exploiting context

language constructs to make data more accurate

Uncertain<*T*>

an abstraction for reasoning about noise [ASPLOS'14]

Thanks!

exploiting context

language constructs to make data more accurate