Tutorial

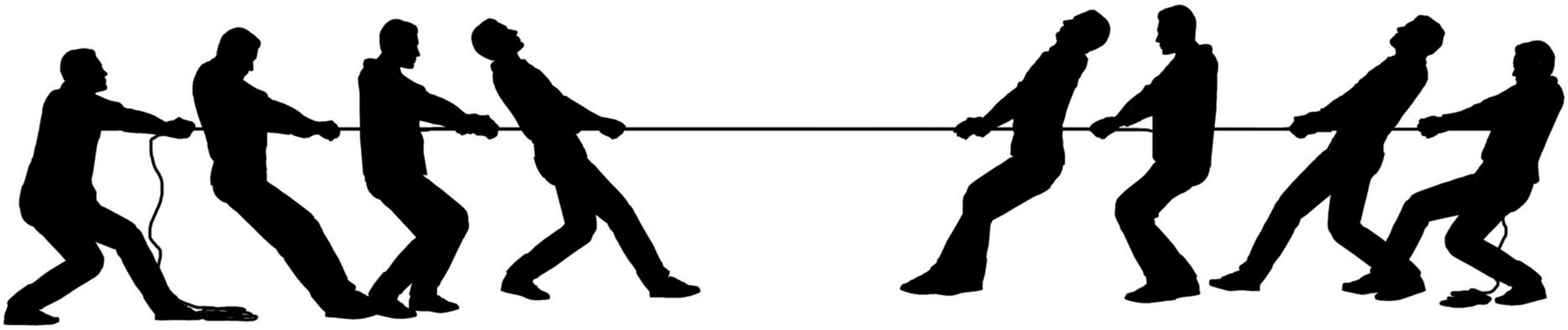
Mental models as probabilistic programs

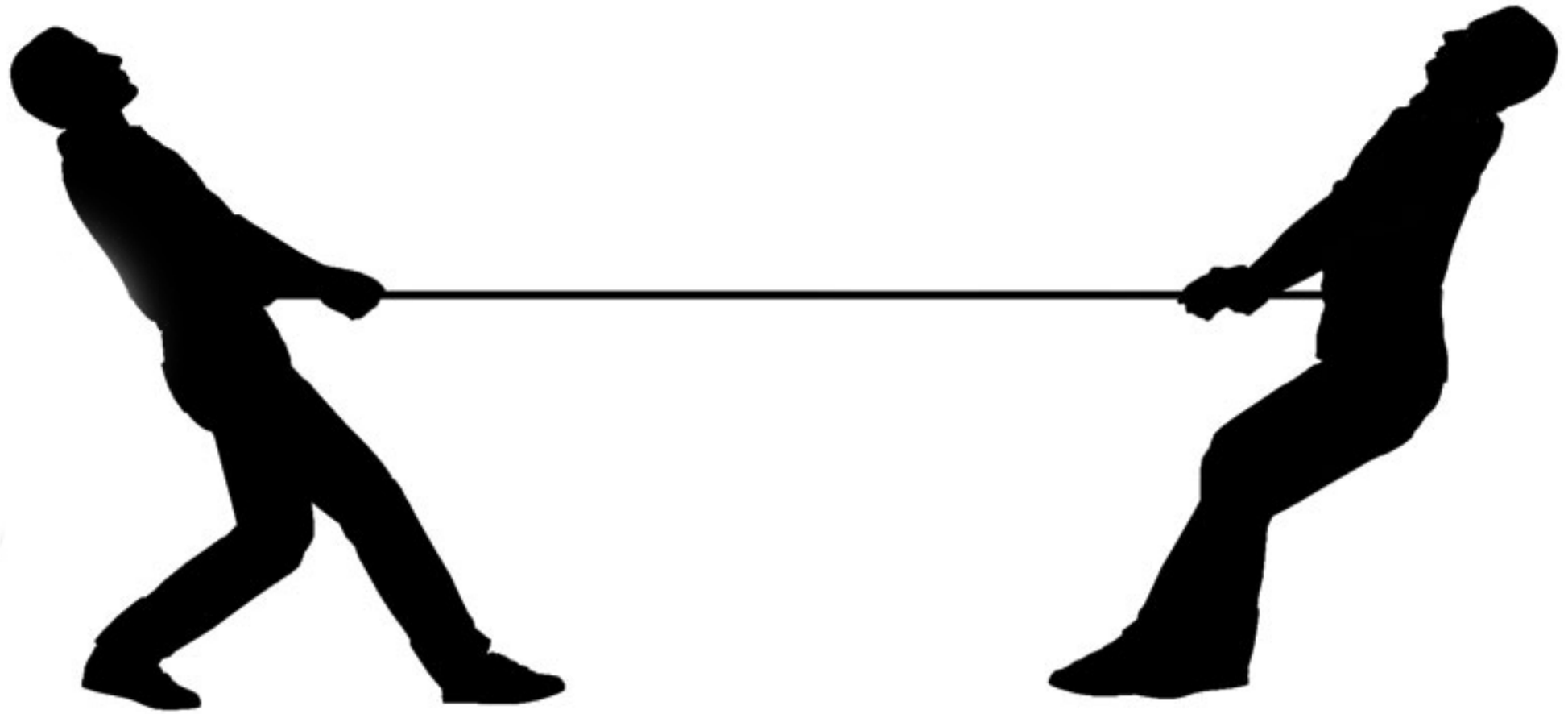
Tobias Gerstenberg
Kevin Smith
Tomer Ullman

Brains, Minds, and Machines Summer School

August 21st, 2017

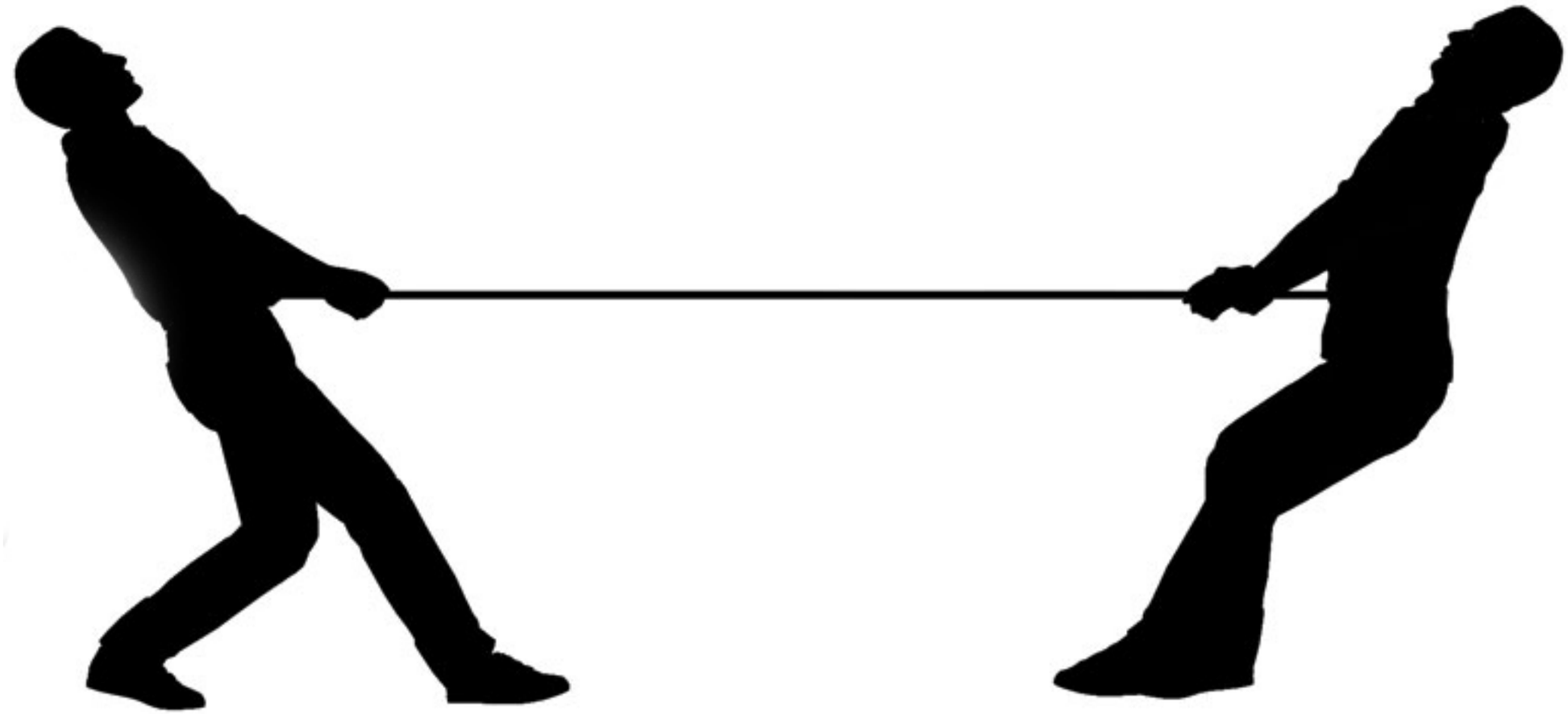
Tug of War





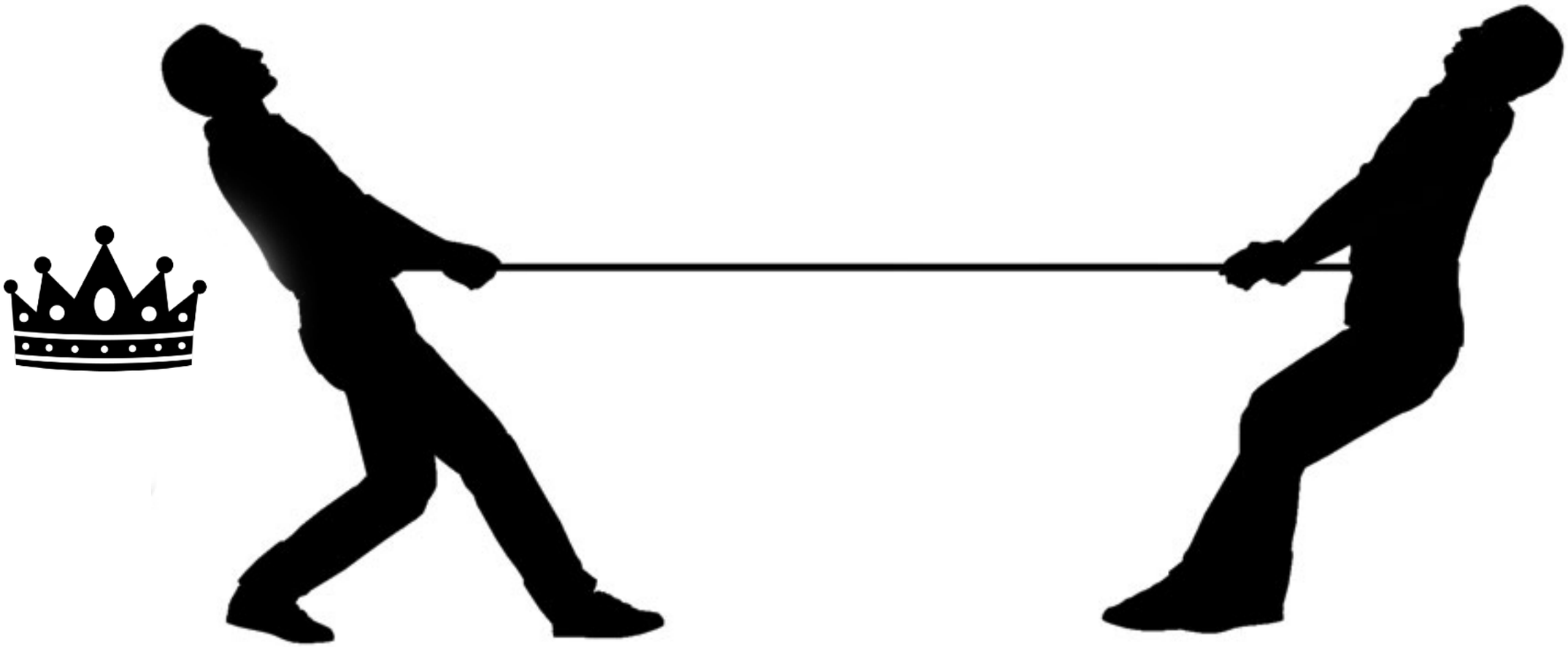
Tom

Steve



Tom

Steve

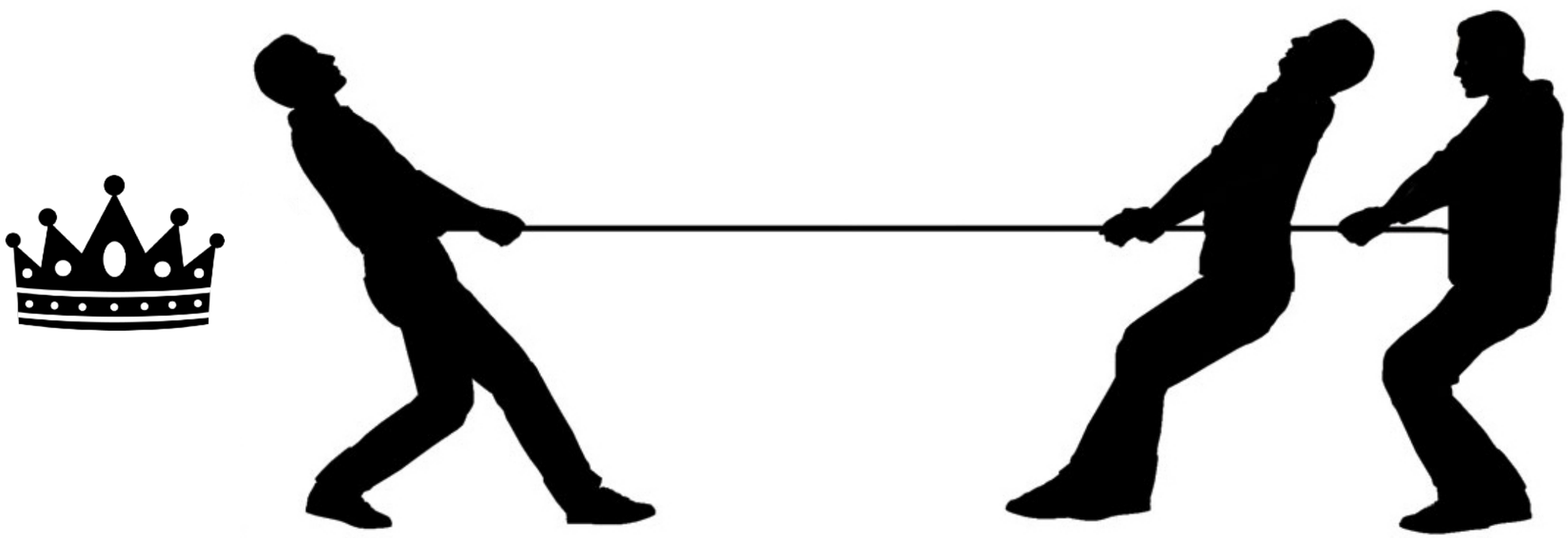


Tom

Steve

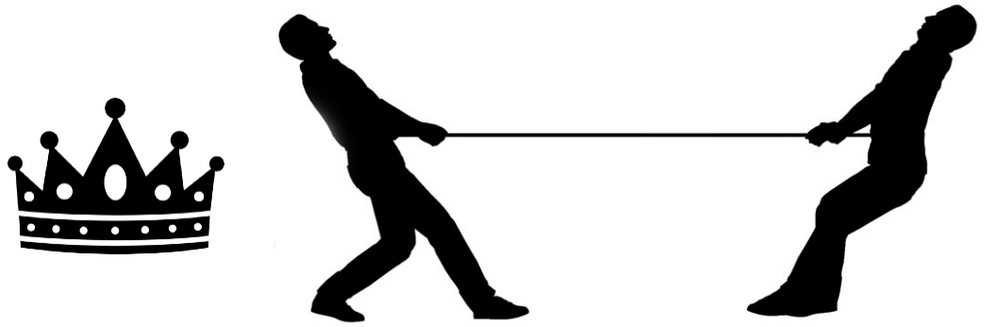
very weak

very strong

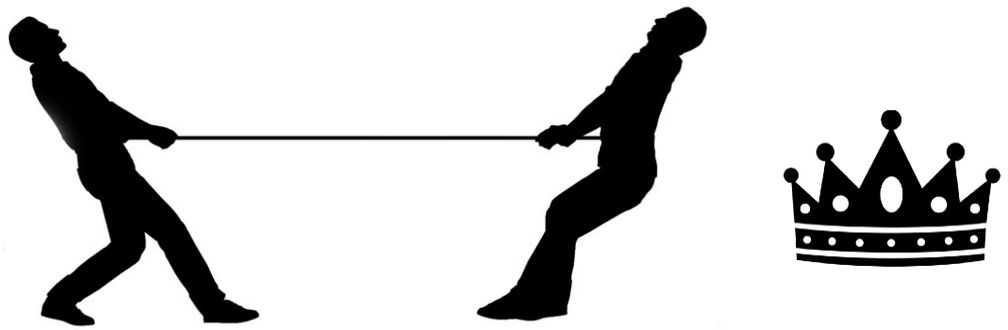


Bill Mark Nick

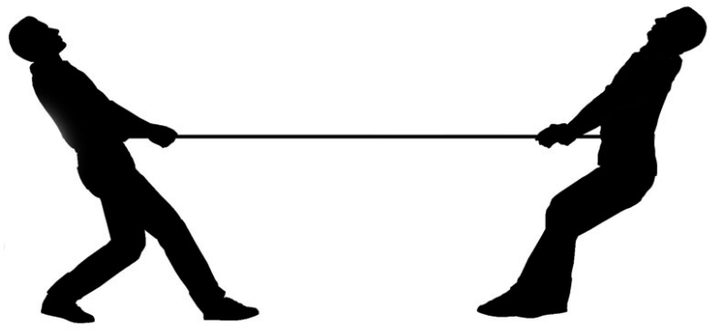
very weak very strong



Tom Steve

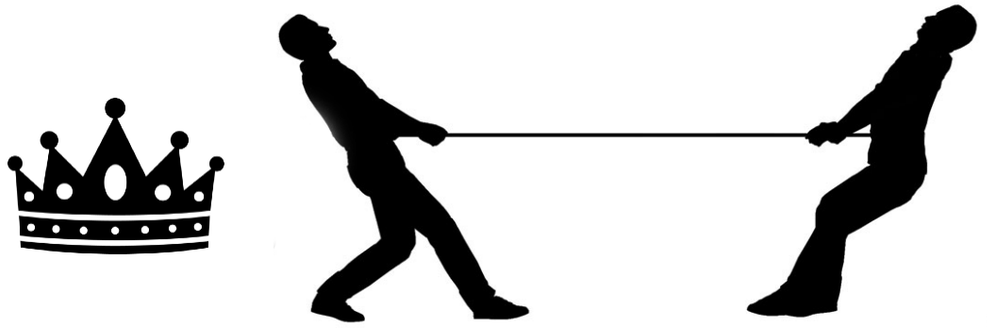


Bill Steve

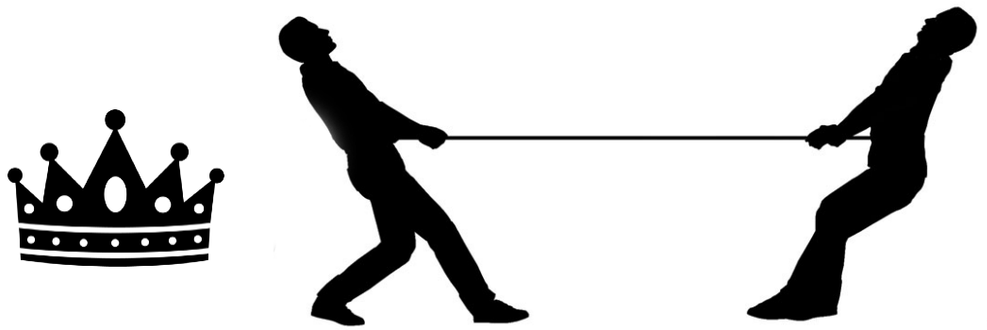


Ryan Steve

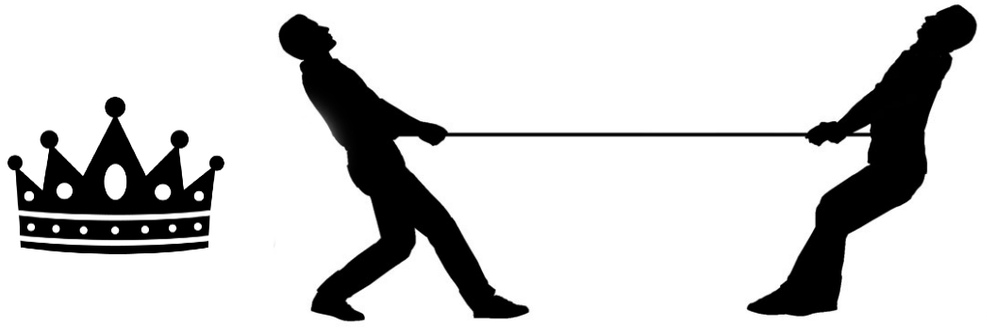
very weak very strong



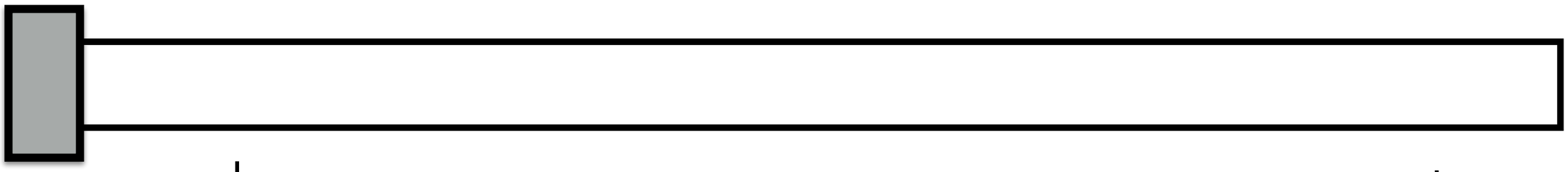
Tom Steve



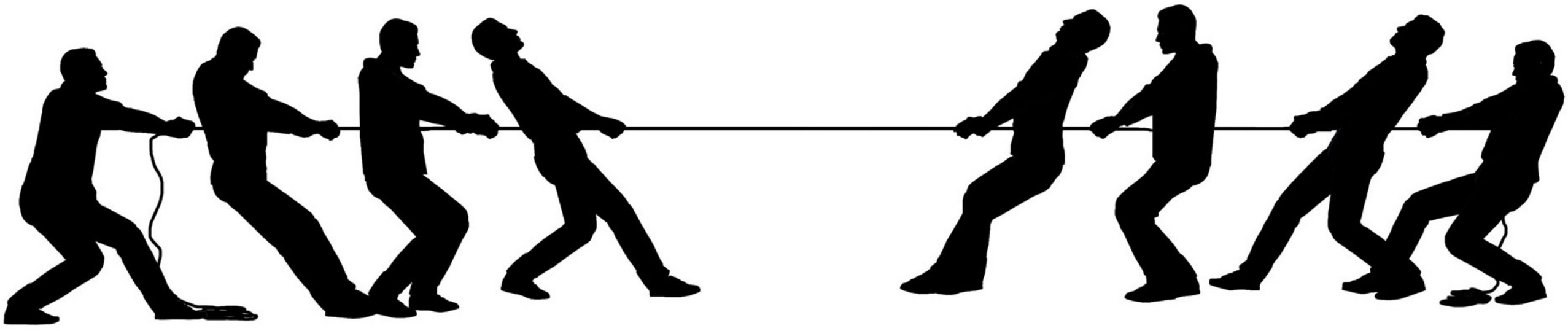
Bill Steve



Ryan Steve



very weak very strong



Concepts:

teams

person

winner

strength

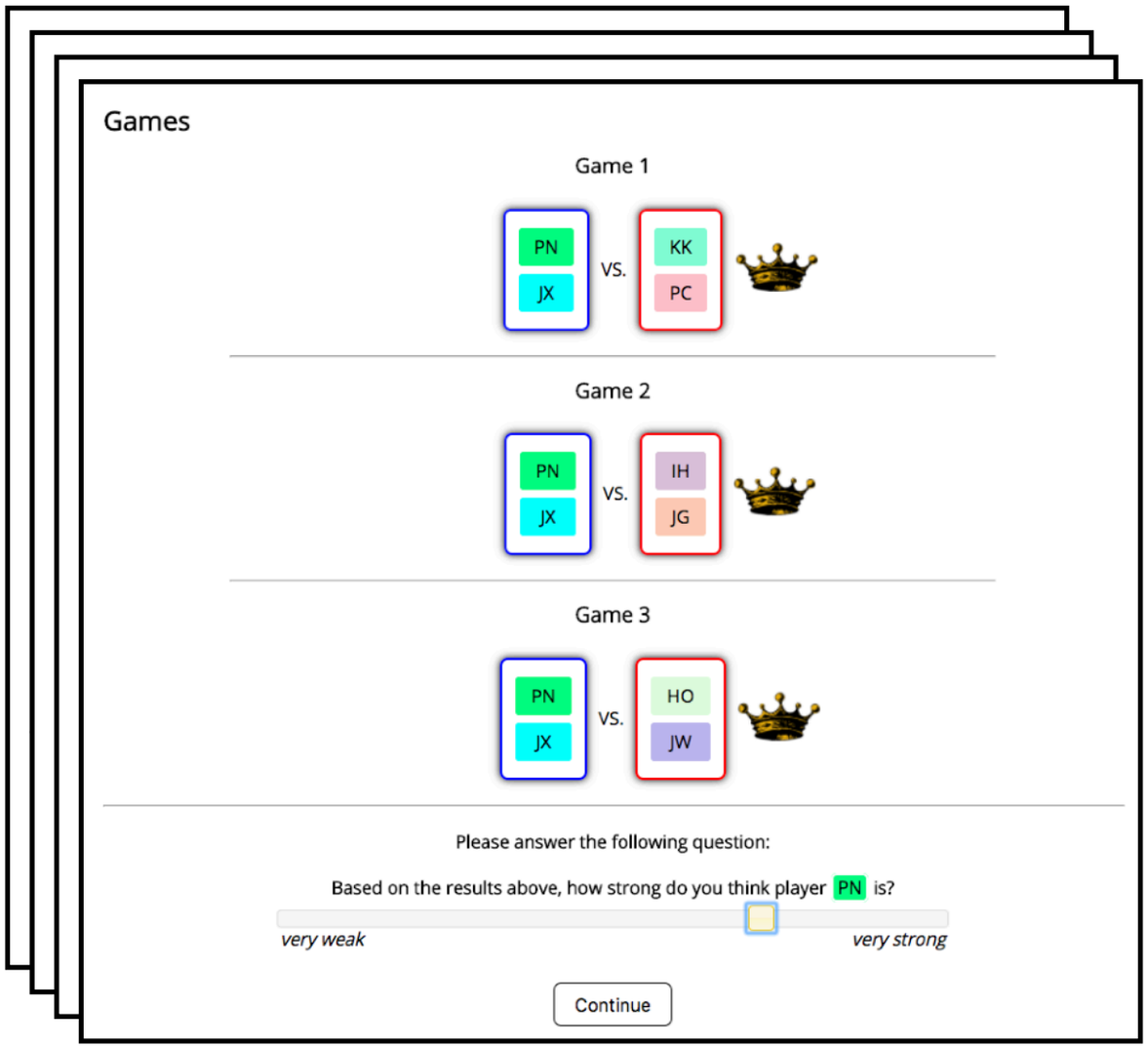
pulling

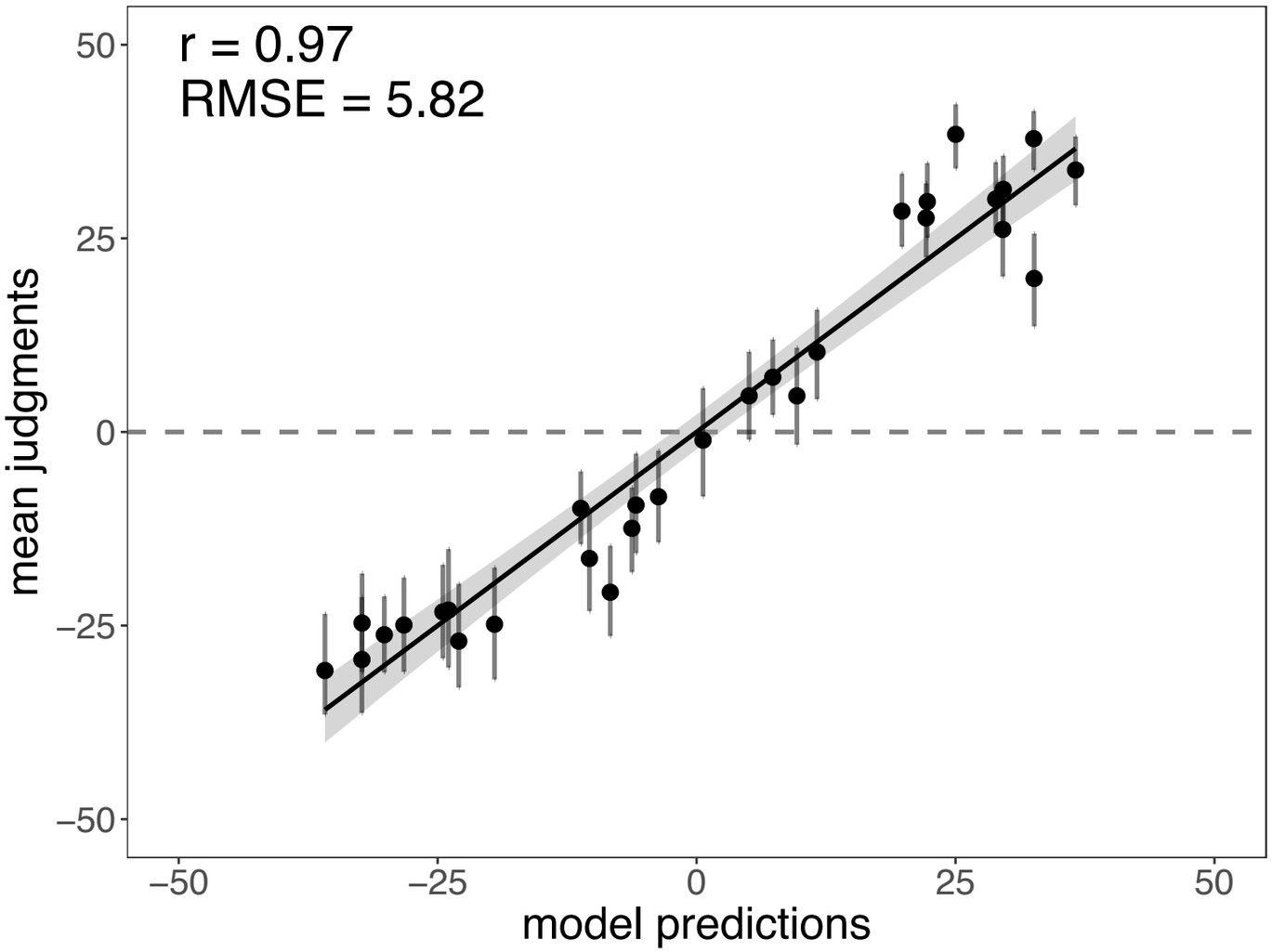
effort

A computational model of tug of war

```
var towModel = function() {
  var strength = mem(function (person) {return gaussian(50, 10)})

  var lazy = function(person) {return flip(0.1) }

  var pulling = function(person) {
    return lazy(person) ? strength(person) / 2 : strength(person) }

  var totalPulling = function (team) {return sum(map(pulling, team))}

  var winner = function (team1, team2) {
    totalPulling(team1) > totalPulling(team2) ? team1 : team2 }

  var beat = function(team1,team2){winner(team1,team2) == team1}

  condition(beat(['bob'], ['tom']))

  return strength('bob')
}
```

Experiment





Gerstenberg & Goodman (2012) Ping Pong in Church: Productive use of concepts in human probabilistic inference. *Cognitive Science Proceedings*

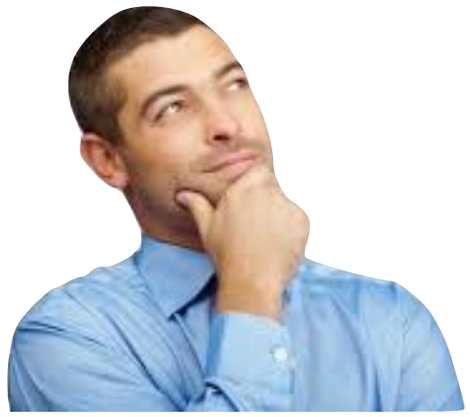
Gerstenberg & Tenenbaum (2017) Intuitive Theories. *Oxford Handbook of Causal Reasoning*

Goodman, Tenenbaum, & Gerstenberg (2015) Concepts in a probabilistic language of thought. *The Conceptual Mind: New Directions in the Study of Concepts*

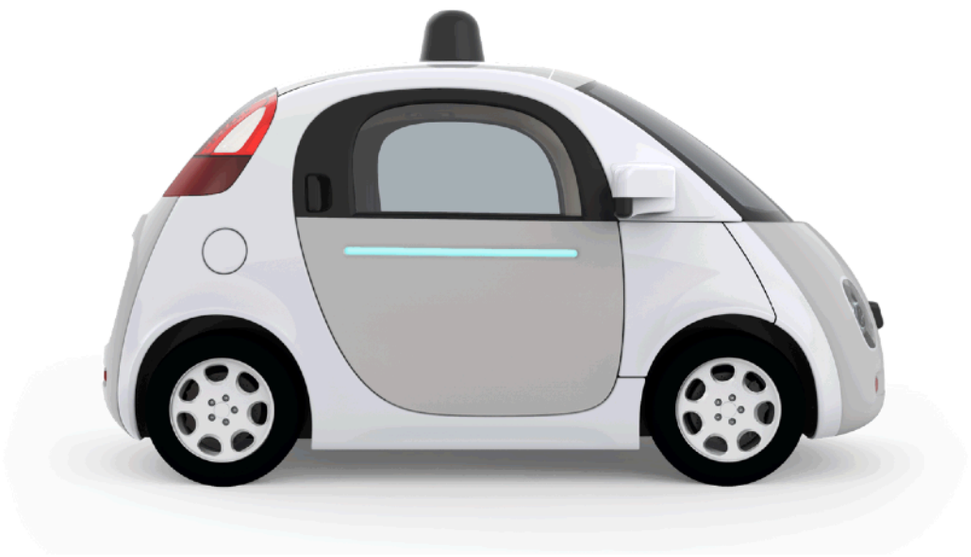
Outline

- What is thought?
- What kind of a program?
- Probabilistic language of thought hypothesis
  - Compositionality
  - Productivity
- **Do it yourself!**
  - WebPPL basics
  - Building generative models
  - Doing inference
- Questions, Resources, some cool examples

What is thought?



How can we describe the intelligent inferences made in everyday human reasoning and learning?



How can we engineer intelligent machines?

What is thought?

Computational theory of mind

mind = computer

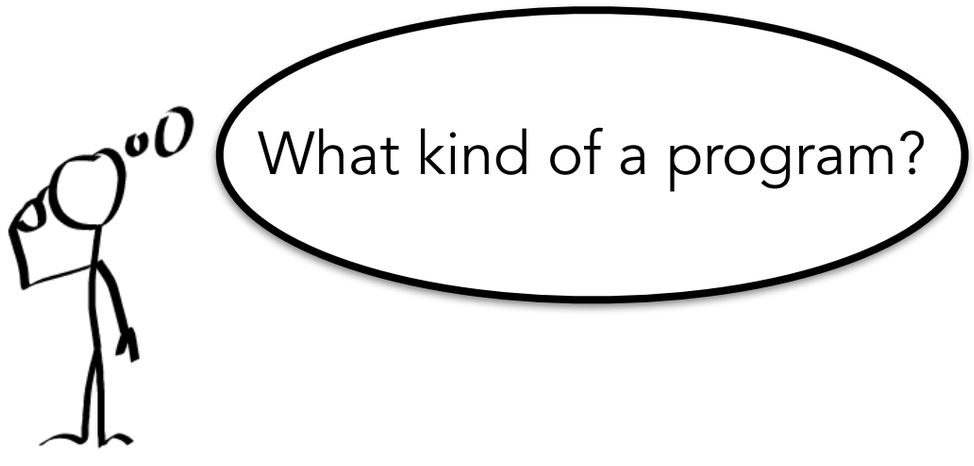
mental representations =

computer programs

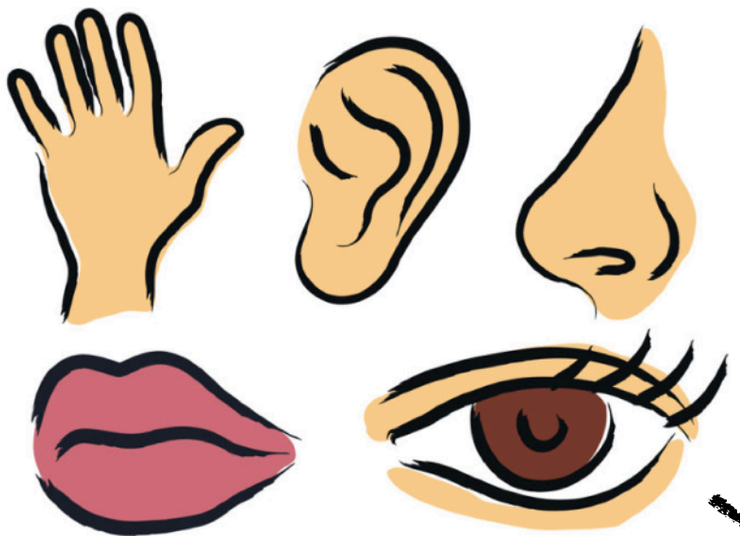
run(program)

thinking =

running a program



What kind of a program?



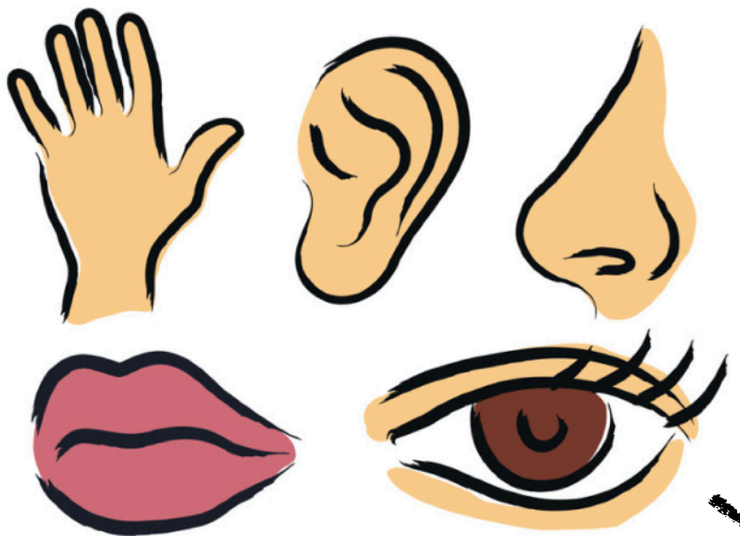
Input

if … then …

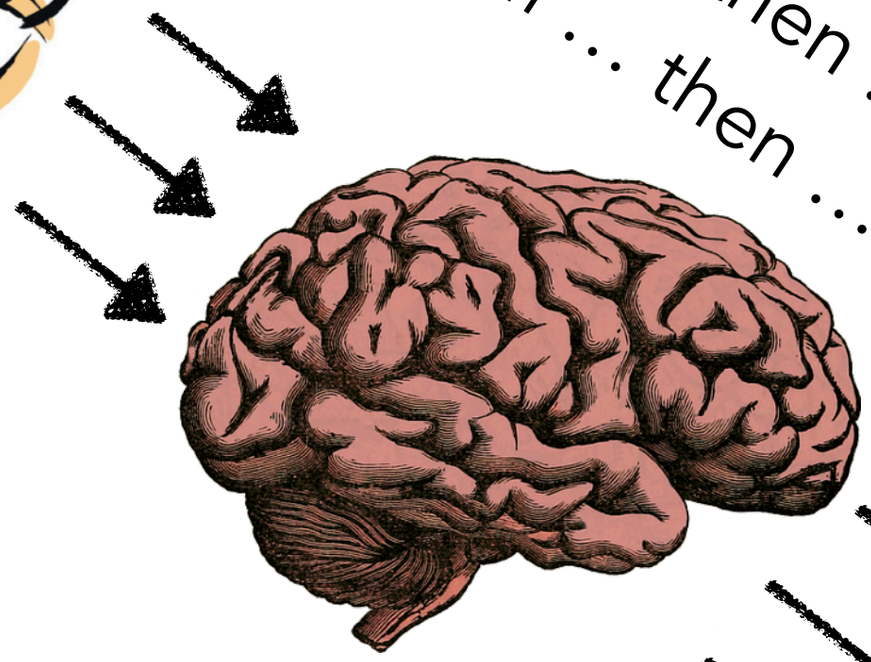
Functional mapping

Output

What kind of a program?



Input

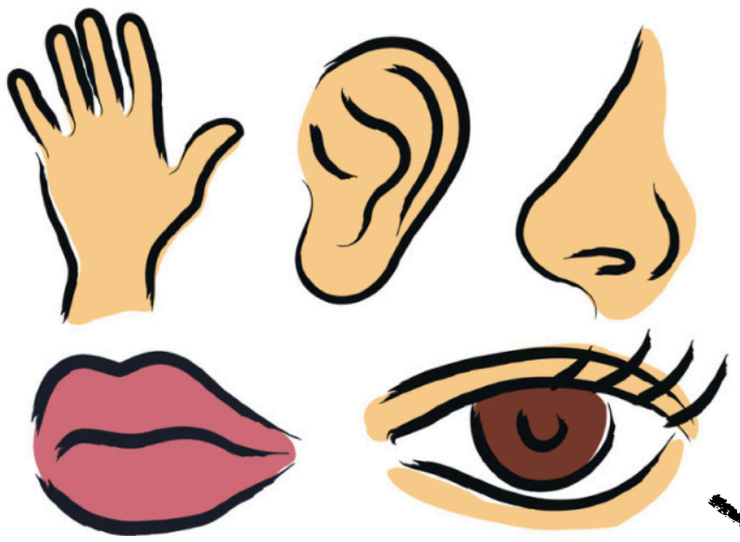


if … then …
if … then …
if … then …

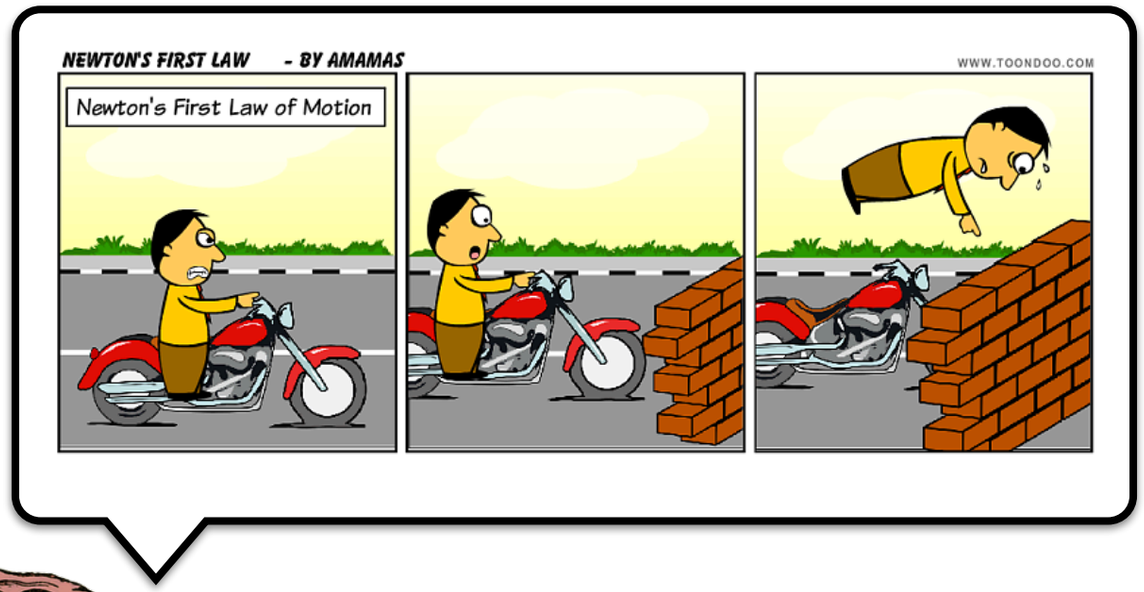
Functional mapping

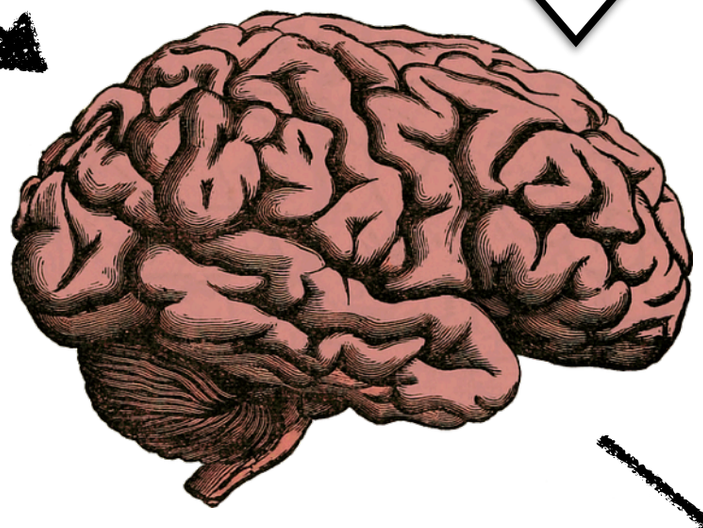
Output

What kind of a program?



Input

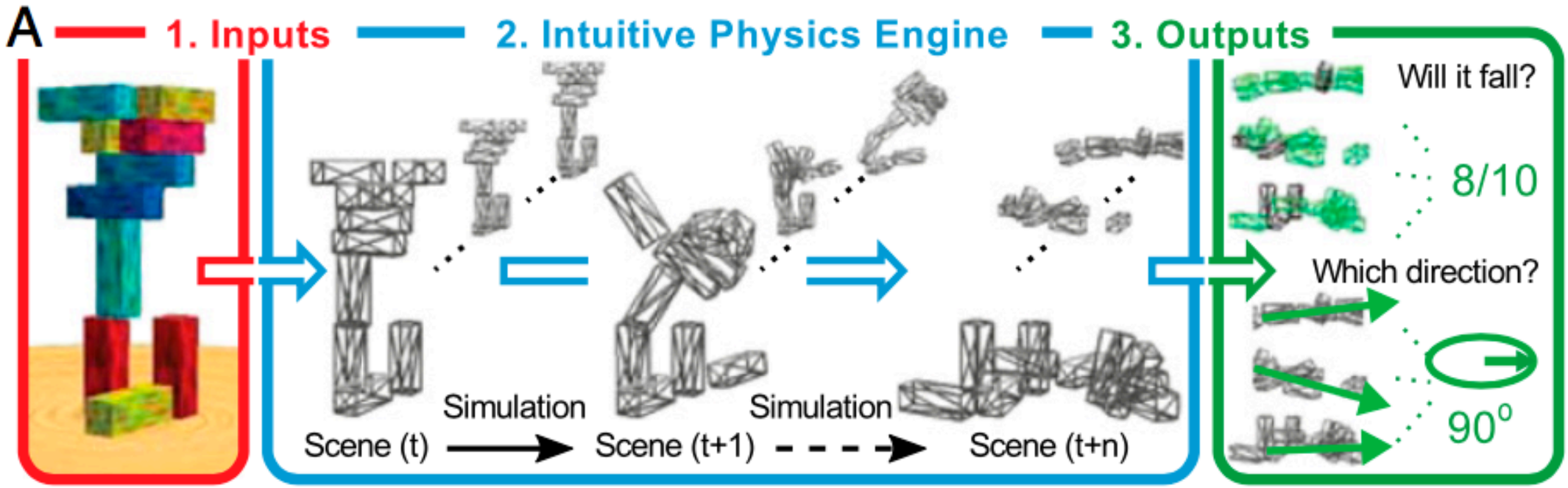


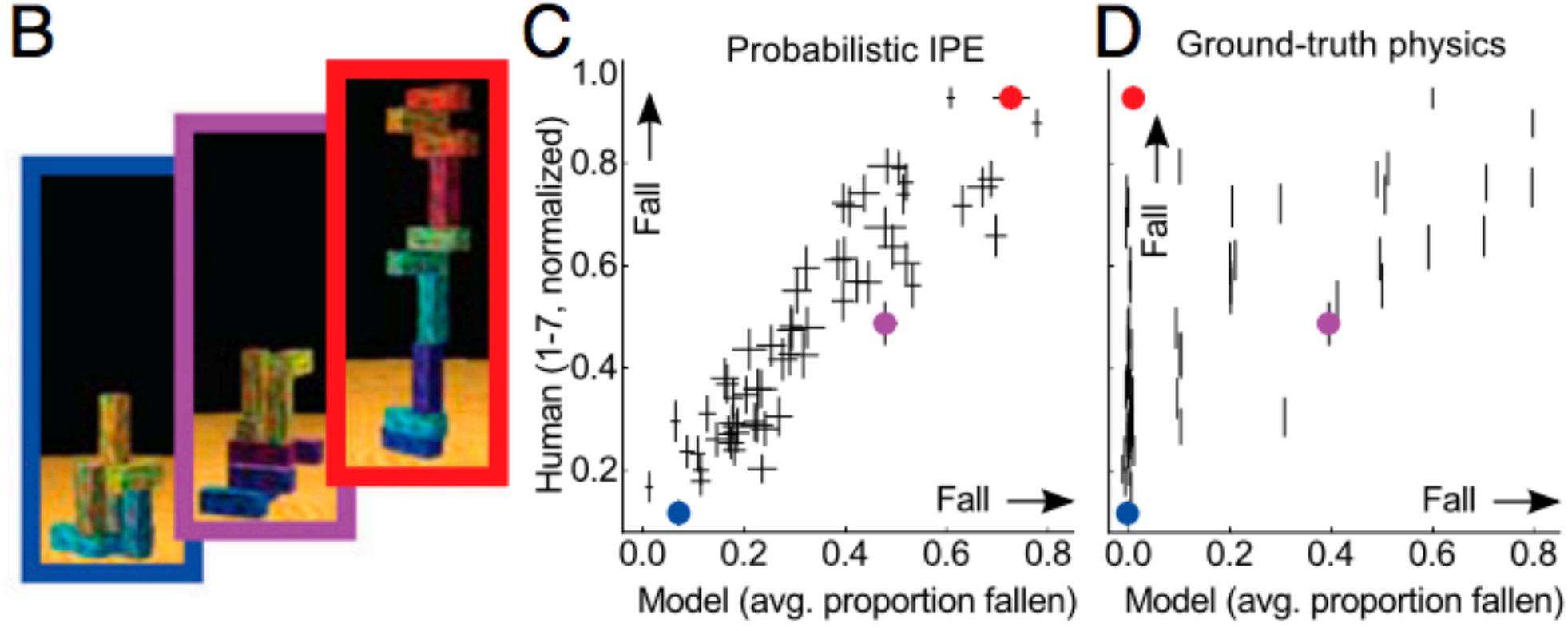


Intuitive theories

Output

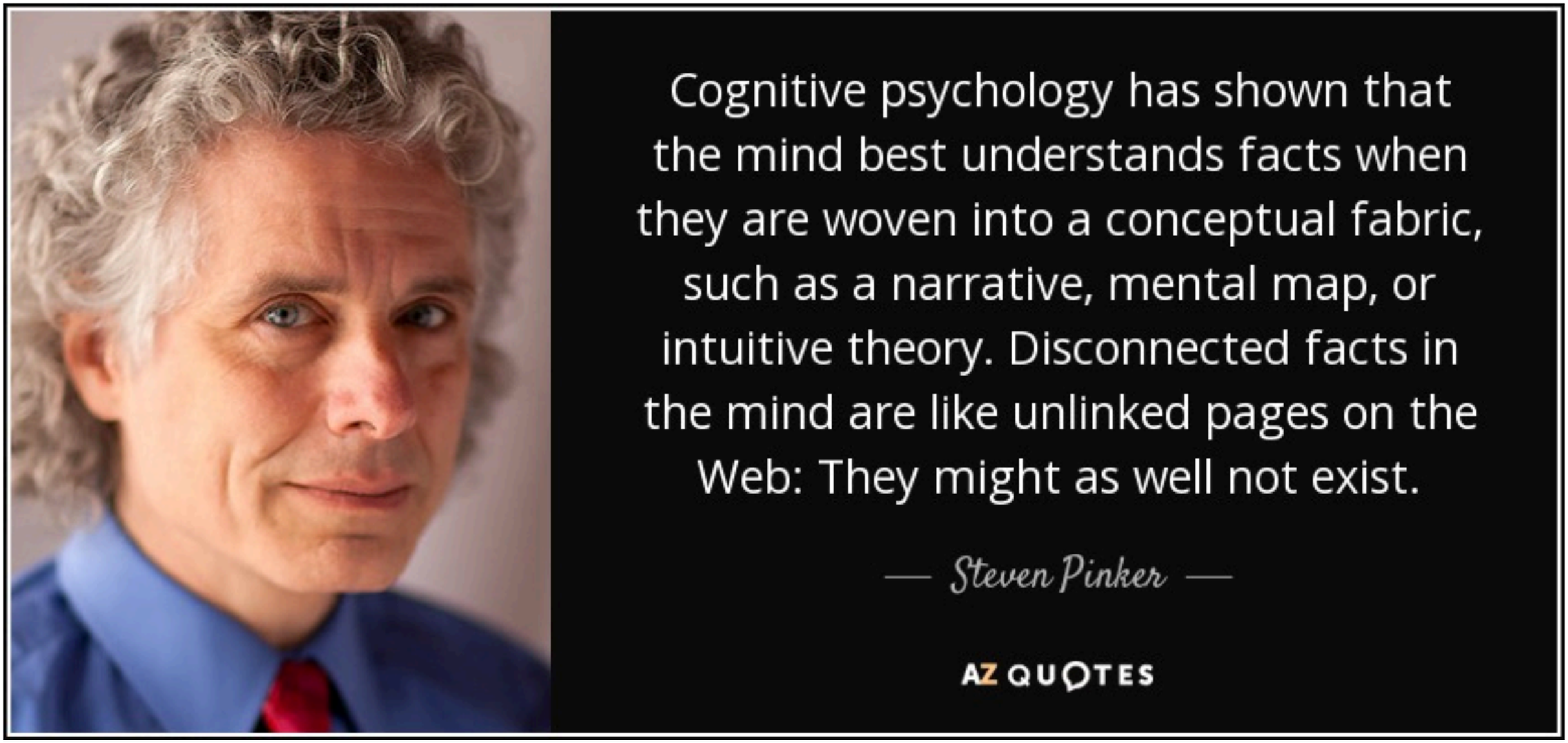
What kind of a program?





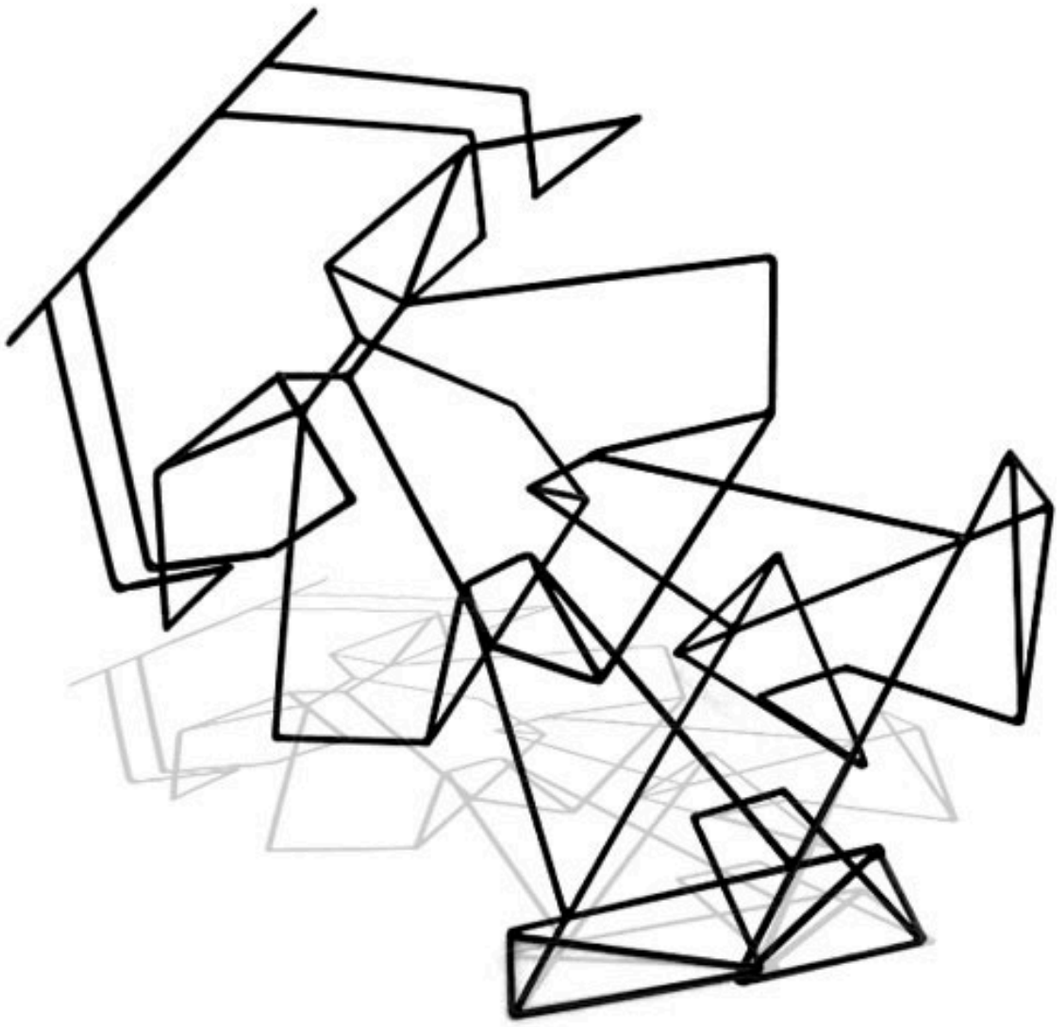
Battaglia, Hamrick, & Tenenbaum (2013) Simulation as an engine of physical scene understanding. Proceedings of the National Academy of Sciences

What kind of a program?



What kind of a program?

Structure



Probability

Knowledge

Uncertainty

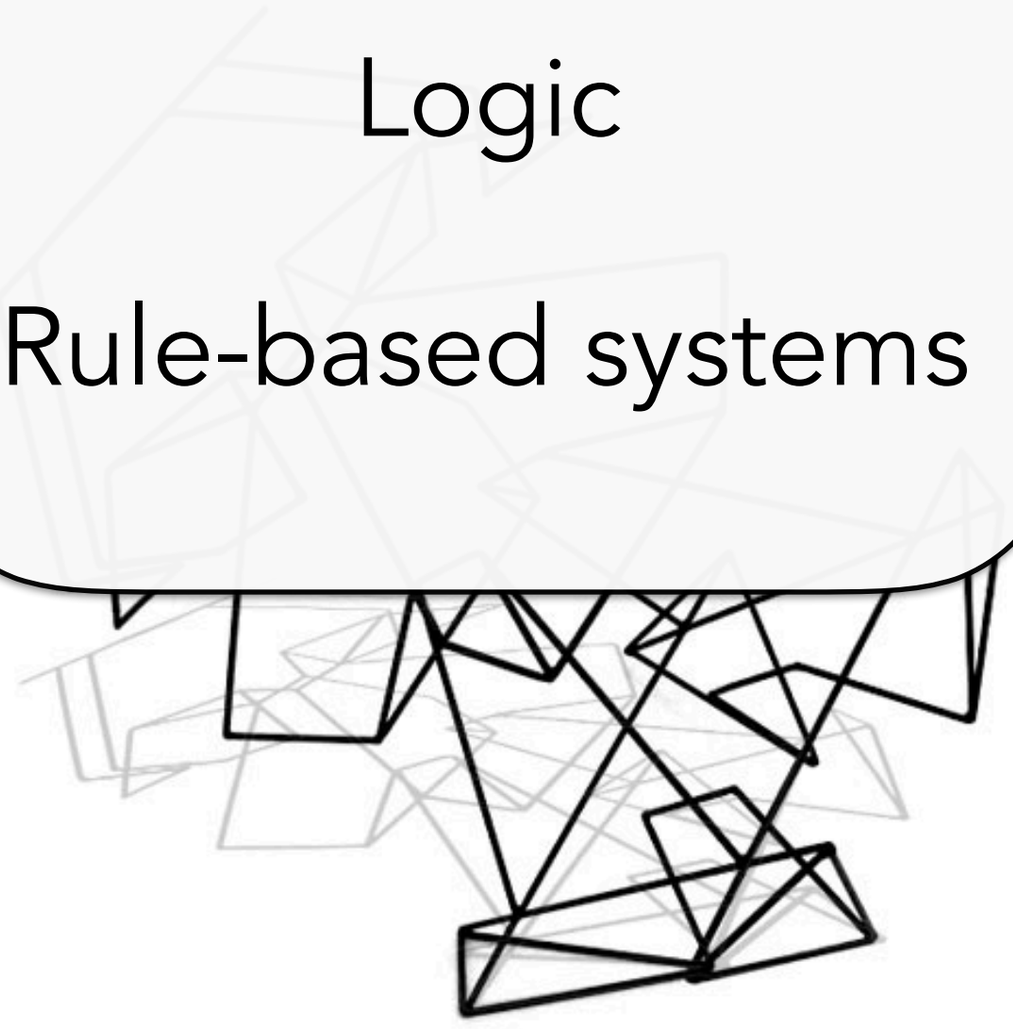
Tenenbaum, Kemp, Griffiths, & Goodman (2011) How to grow a mind: Statistics, structure, and abstraction. *Science*

What kind of a program?

Structure

Logic

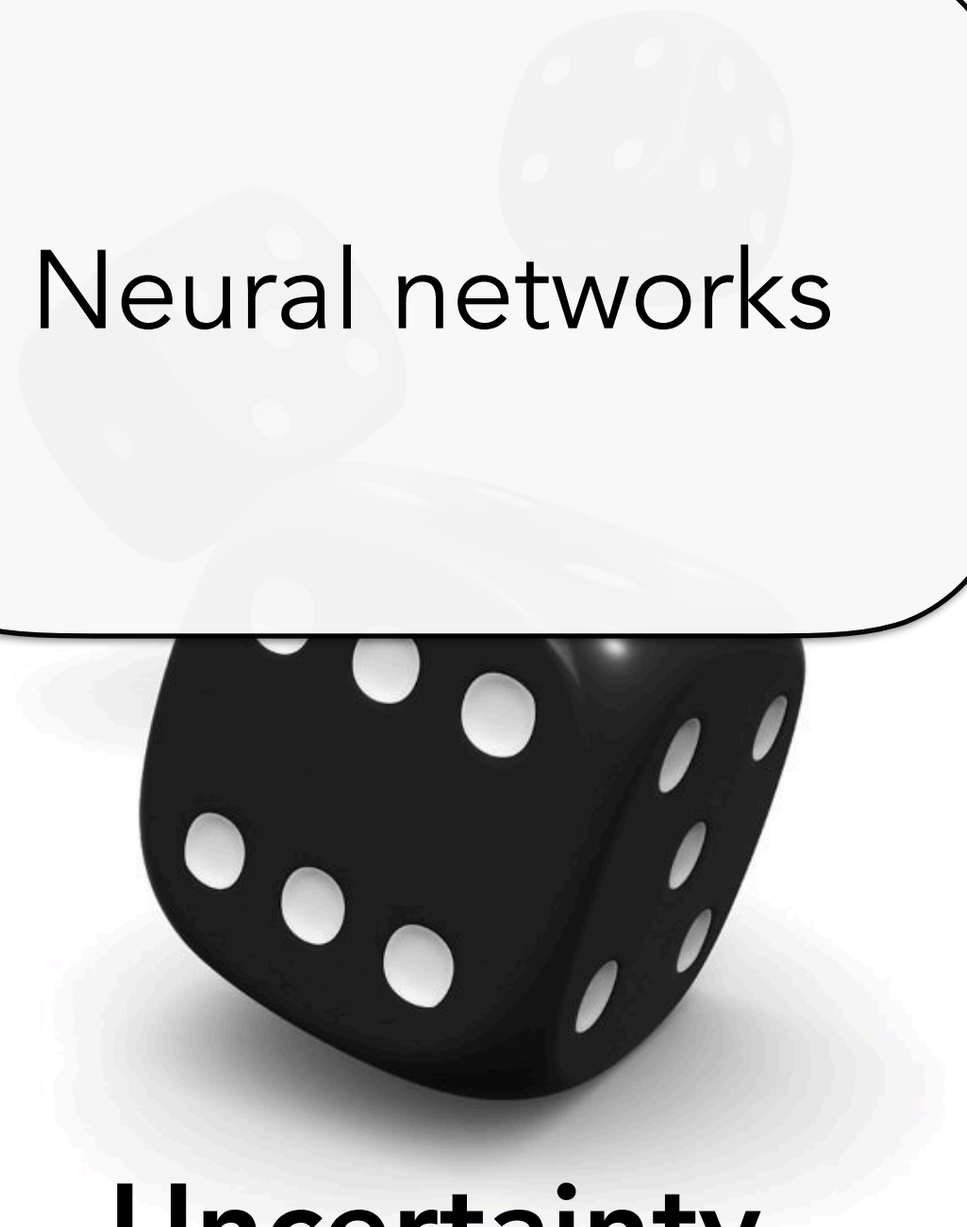
Rule-based systems



Knowledge

Probability

Neural networks



Uncertainty

Tenenbaum, Kemp, Griffiths, & Goodman (2011) How to grow a mind: Statistics, structure, and abstraction. *Science*

What kind of a program?

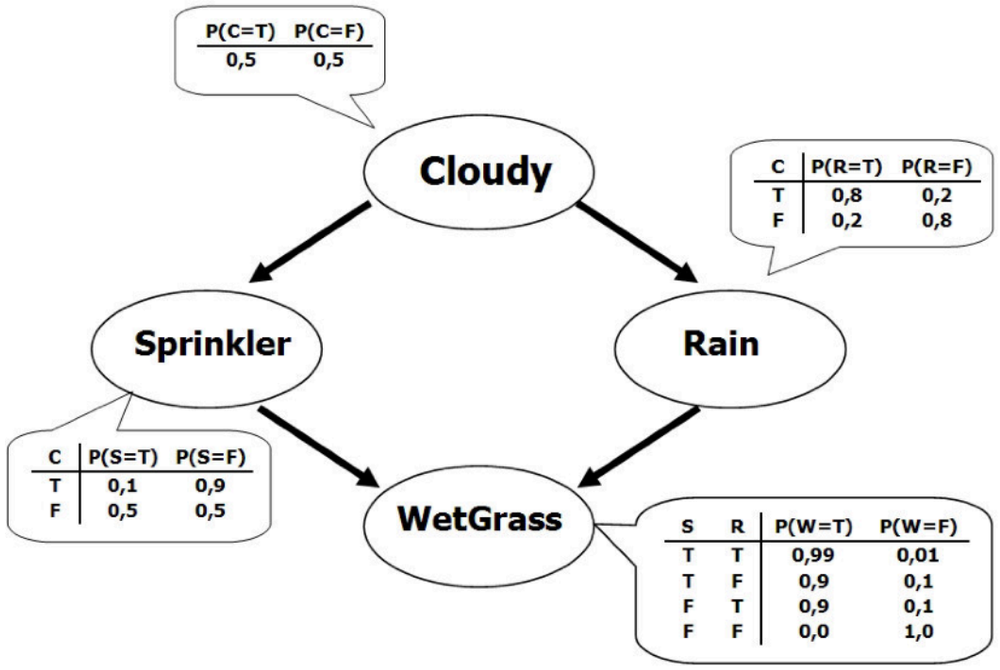
Structure	Probability
Logic Rule-based systems	Neural networks

Probabilistic graphical models
(Bayesian Networks)

Probabilistic programs

Tenenbaum, Kemp, Griffiths, & Goodman (2011) How to grow a mind: Statistics, structure, and abstraction. *Science*

What kind of a program?



(Causal) Bayesian networks

- represent probabilistic/causal dependencies
- have "thick arrows"
- limited expressiveness
- relationships between events / propositions

```
var towModel = function() {
  var strength = mem(function (person) {return gaussian(50, 10)})

  var lazy = function(person) {return flip(0.1) }

  var pulling = function(person) {
    return lazy(person) ? strength(person) / 2 : strength(person) }

  var totalPulling = function (team) {return sum(map(pulling, team))}

  var winner = function (team1, team2) {
    totalPulling(team1) > totalPulling(team2) ? team1 : team2 }

  var beat = function(team1,team2){winner(team1,team2) == team1}

  condition(beat(['bob'], ['tom']))

  return strength('bob')
}
```

Probabilistic programs

- have rich compositionality
- provide a complete specification of the domain (nothing hidden behind arrows)
- support productive inferences
- support reasoning about events, properties, agents, objects, …

Probabilistic language of thought hypothesis

Informal version:

"Concepts have a language-like compositionality and encode probabilistic knowledge. These features allow them to be extended productively to new situations and support flexible reasoning and learning by probabilistic inference."

Fodor (1975). The language of thought

Goodman, Tenenbaum, & Gerstenberg (2015) Concepts in a probabilistic language of thought. *The Conceptual Mind: New Directions in the Study of Concepts*

Probabilistic language of thought hypothesis

Informal version:

"Concepts have a language-like **compositionality** and encode probabilistic knowledge. These features allow them to be extended **productively** to new situations and support flexible reasoning and learning by **probabilistic inference**."

Productivity

"*infinite use of finite means*"
(Wilhelm von Humboldt)

Language

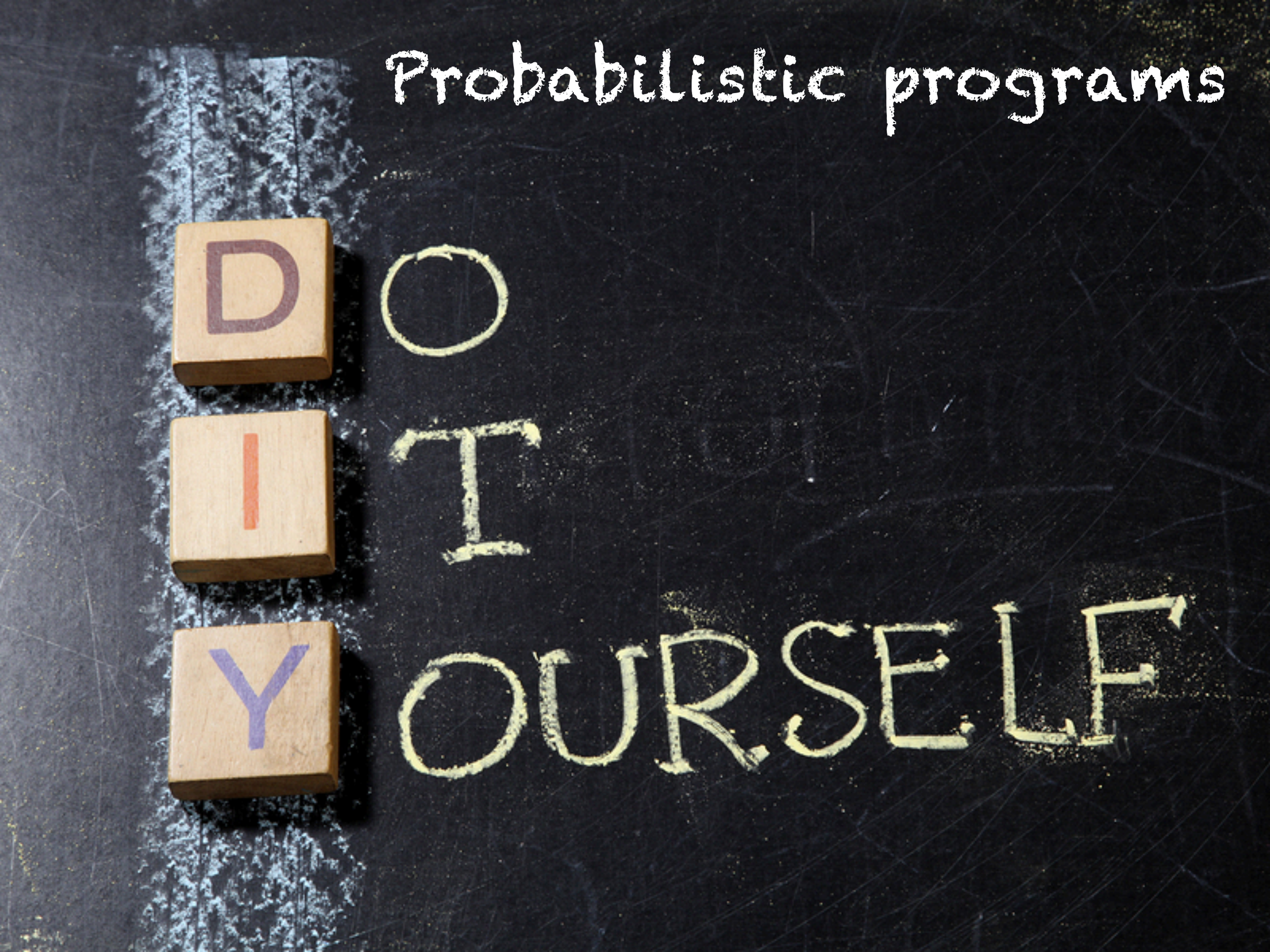
limited number of letters

fixed set of grammatical rules

can express an infinite number of thoughts

Goodman, Tenenbaum, & Gerstenberg (2015) Concepts in a probabilistic language of thought. *The Conceptual Mind: New Directions in the Study of Concepts*

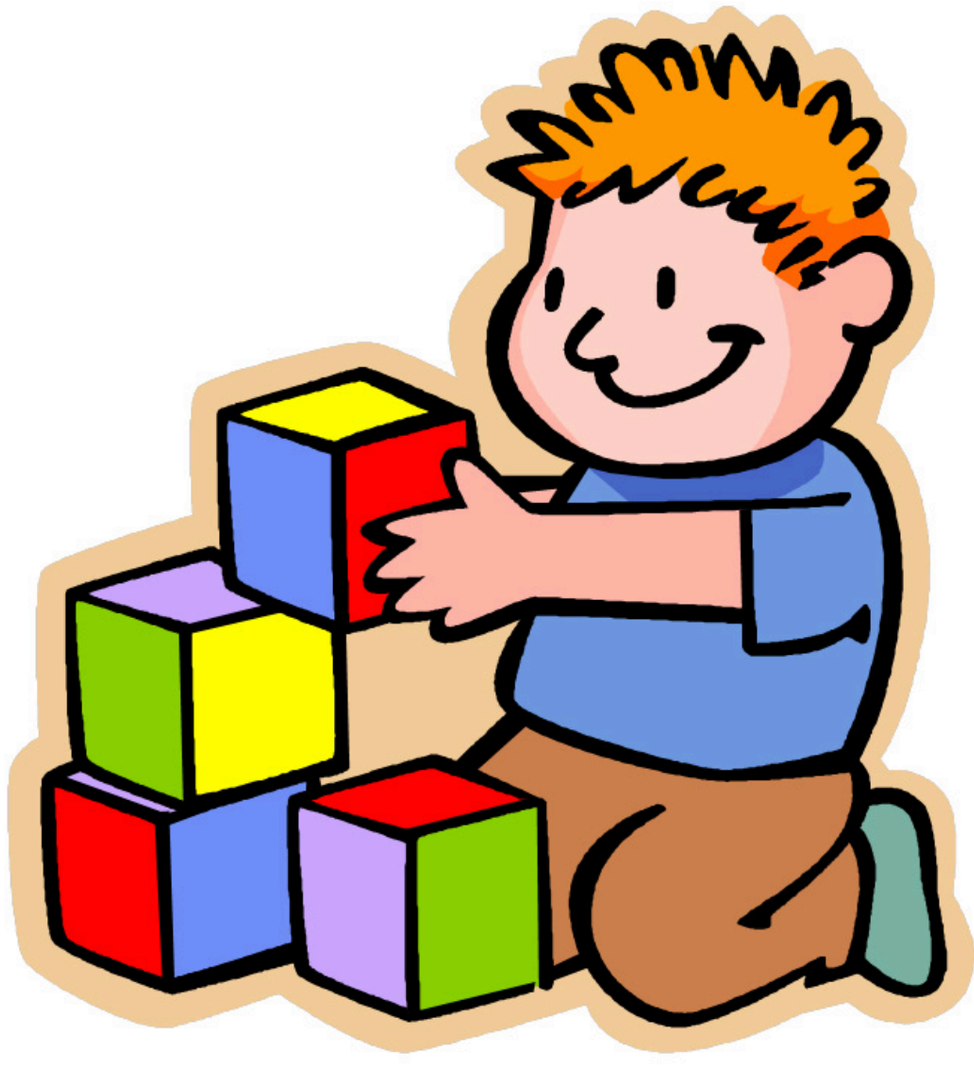
Probabilistic programs



1. WebPPL basics
2. Building generative models
3. Doing inference

Think

Play



http://bit.do/webppl

1. WebPPL basics

WebPPL basics

- declare variables
- data formats: numbers, strings, logicals, dictionaries, arrays
- if-then-statements
- defining functions
- higher order functions
  - `map`
  - `repeat`

WebPPL basics

- WebPPL = purely functional programming language
  - **can't** write `for` loops or `while` statements
  - **can** create higher-order functions and recursive functions
    - for example: `map` = apply a function to each element of a list (like a `for` loop)

2. Building generative models

Building generative models

- forward sampling and random primitives
- building simple models
- sample from probability distributions
- memoization: `mem`
- recursive functions

Building generative models

- WebPPL is a language to formally describe how the world works
- random choices capture our uncertainty or ignorance
- the language is **universal**: it can express any computable process
- causal dependence is important: the program describes what influences what

Stuhlmüller, Tenenbaum, & Goodman (2010) Learning Structured Generative Concepts. Cognitive Science Proceedings

Stuhlmüller & Goodman (2014) Reasoning about reasoning by nested conditioning: Modeling theory of mind with probabilistic programs. Cognitive Systems Research

Building generative models

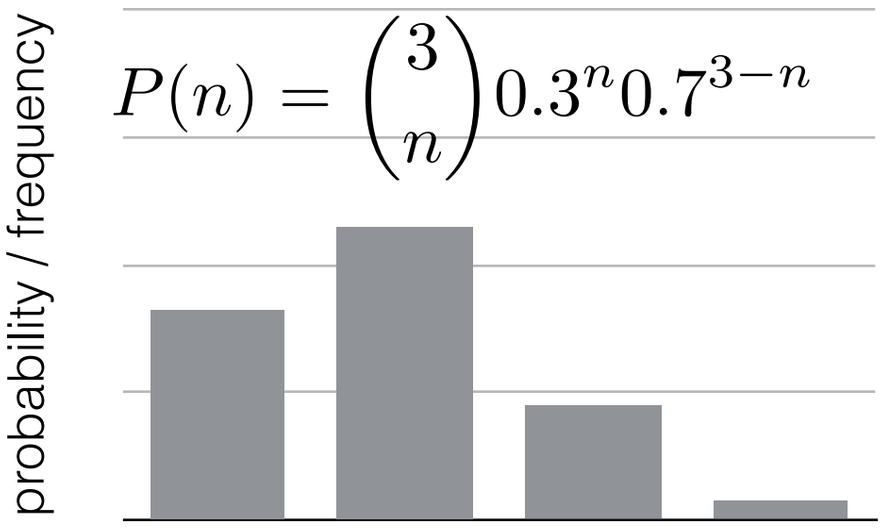
relationship between sampling and probability distributions

Random primitives:

```
var a = flip(0.3)
var b = flip(0.3)
var c = flip(0.3)
return a + b + c
```

```
=> 1 0 0
=> 0 0 0
=> 1 0 1
=> 2 0 1
```

...



Sampling $\cong$ Distributions

any computable distribution can be represented as the distribution induced by sampling from a probabilistic program

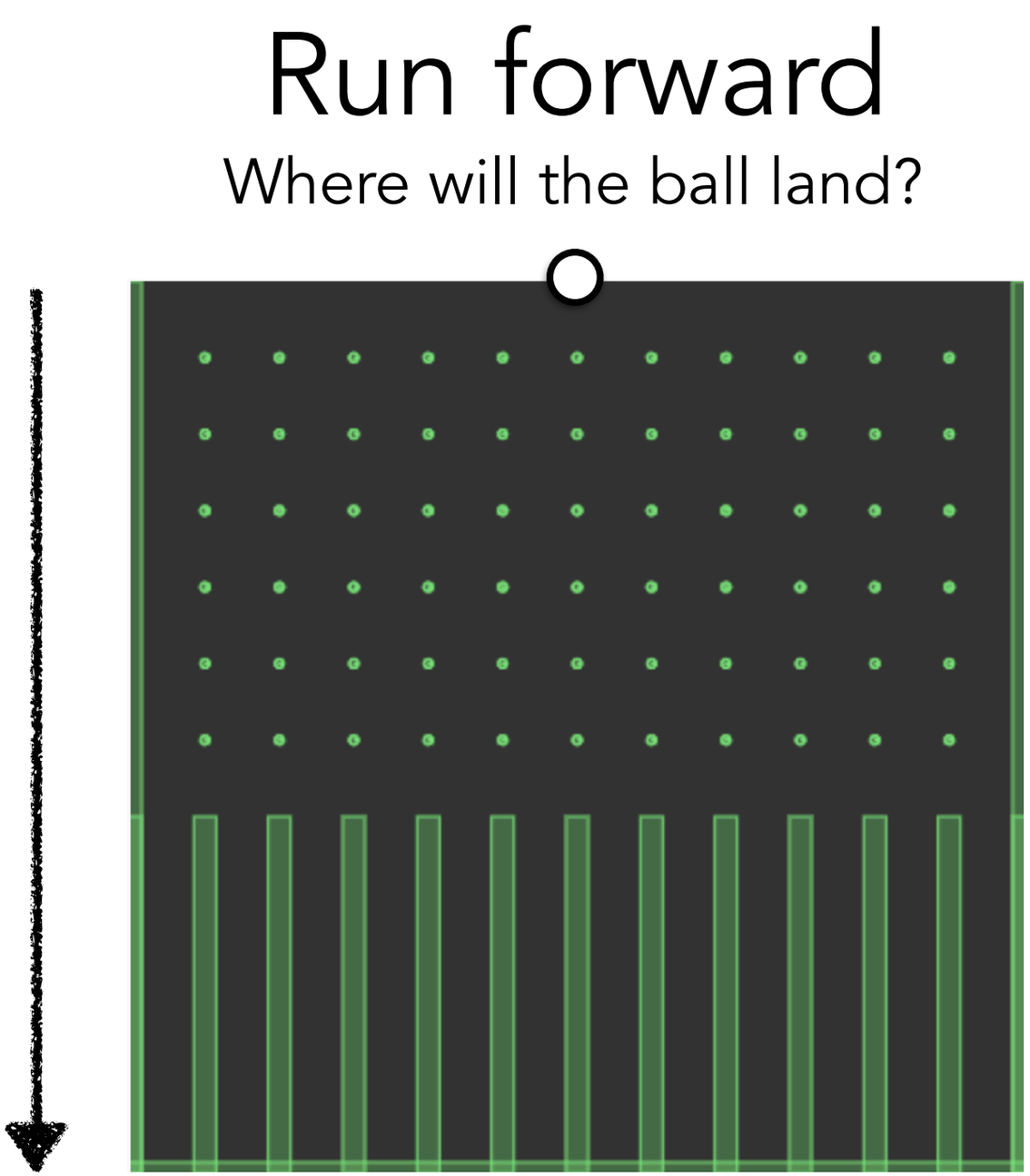
Goodman, Mansinghka, Roy, Bonawitz, & Tenenbaum (2008) Church: A language for generative models. Uncertainty in Artificial Intelligence

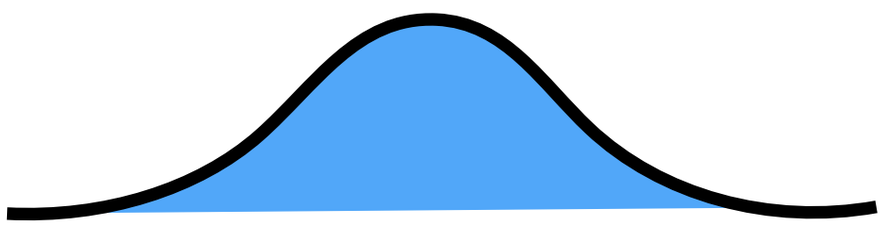
3. Doing inference

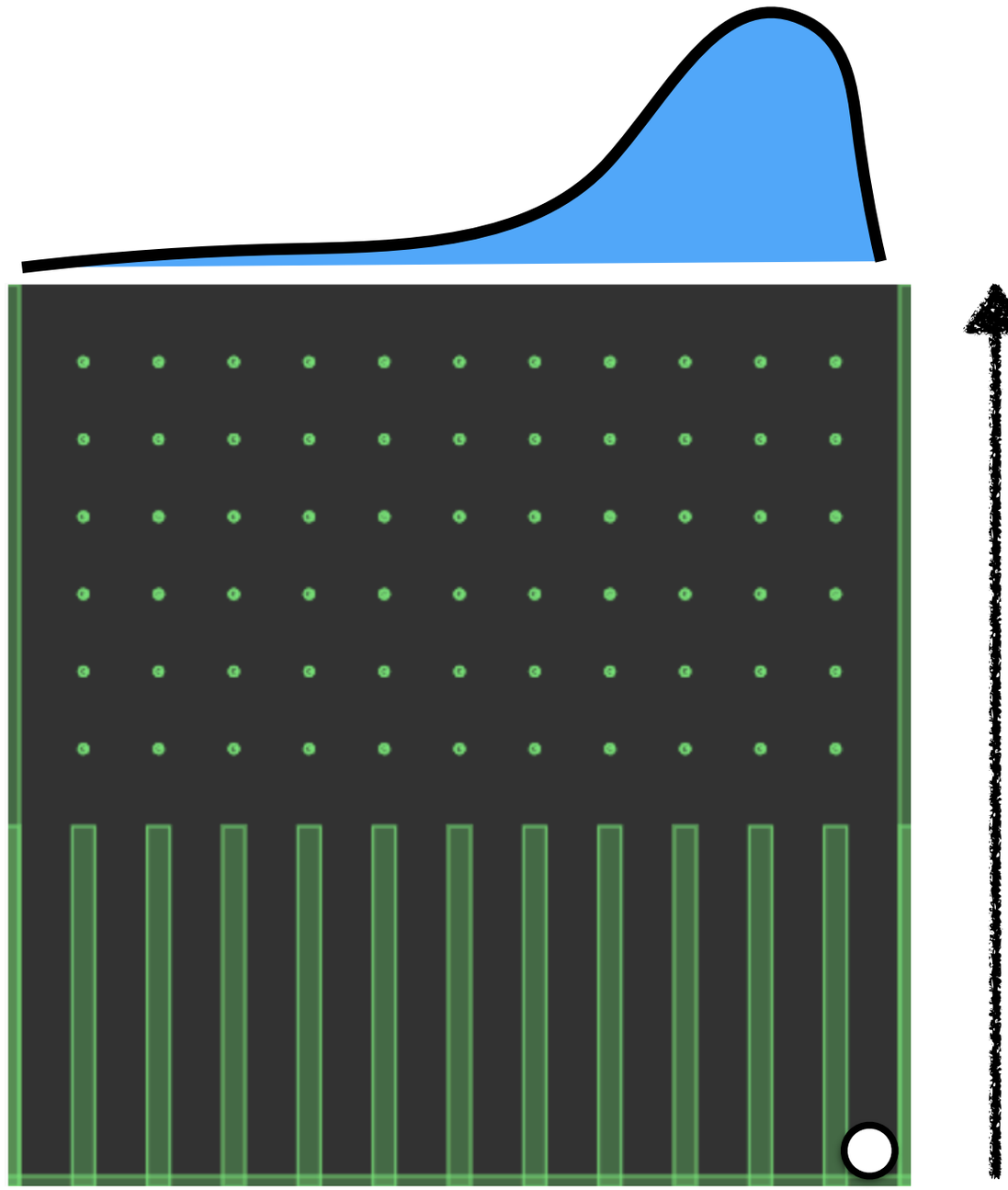
Doing inference

Run forward

Where will the ball land?







Reason backward

Where did the ball come from?

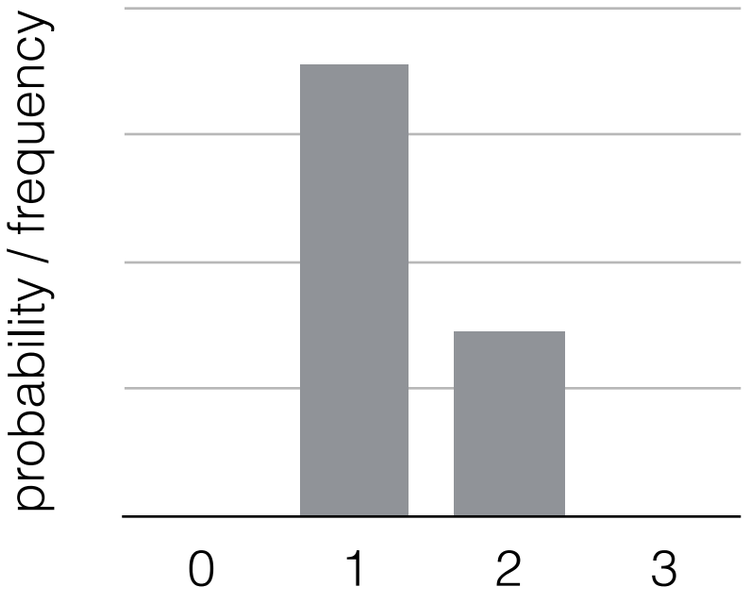
Doing inference

Conditional inference:

```
Infer(
 function(){
  var a = flip(0.3)
  var b = flip(0.3)
  var c = flip(0.3)
  condition(a + b == 1)
  return a + b + c})
```

```
=> 1 0 0 1
=> 0 0 0 0
=> 1 0 1 0
=> T F F T
=> 2 0 1 1
```

Posterior distribution



"It is an old maxim of mine that when you have excluded the impossible, whatever remains, however improbable, must be the truth."

Doing inference

- conditioning on variables
  - rejection sampling
  - WebPPL's inference procedures
- conditioning on arbitrary expressions
- other inference procedures

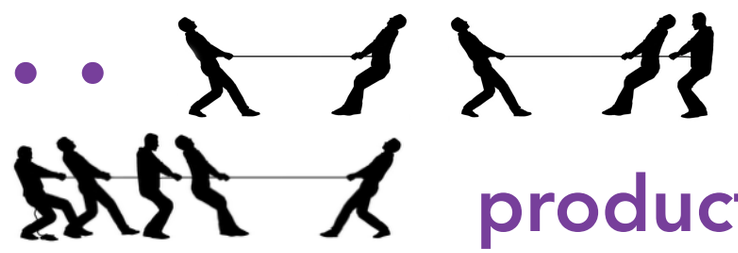
- `forward`, `rejection`, `enumerate`, `MH`, ...

- you don't have to worry about inference. **very nice!**
- WebPPL allows us to parsimoniously describe rich generative model structures and explore the inference patterns that emerge from the interaction of model and data

"Concepts have a language-like **compositionality** and encode probabilistic knowledge. These features allow them to be extended **productively** to new situations and support flexible reasoning and learning by **probabilistic inference**."

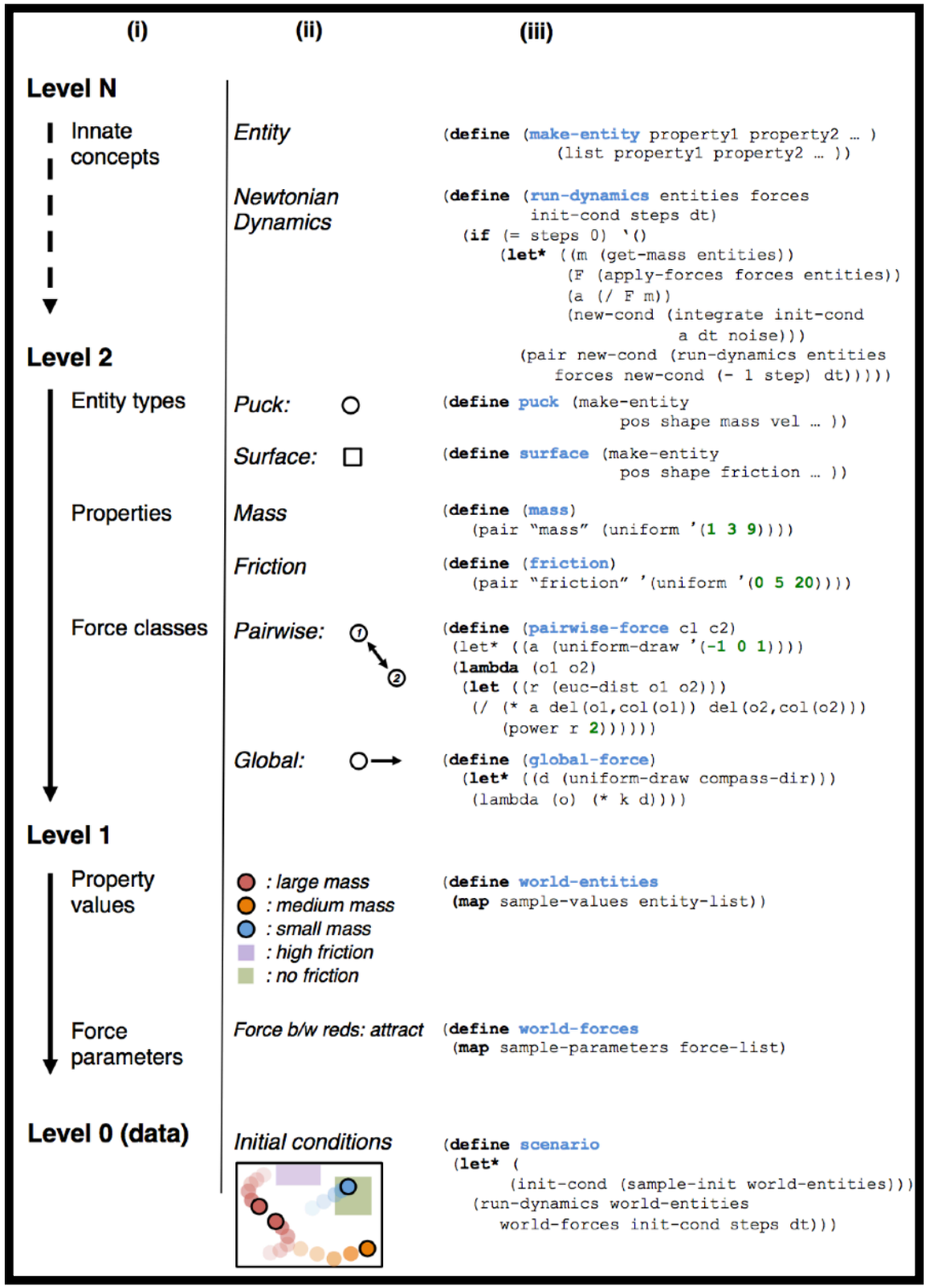
```
var towModel = function() {
  var strength = mem(function (person) {return gaussian(50, 10)})

  var lazy = function(person) {return flip(0.1) }

  var pulling = function(person) {
    return lazy(person) ? strength(person) / 2 : strength(person) }

  var totalPulling = function (team) {return sum(map(pulling, team))}

  var winner = function (team1, team2) {
    totalPulling(team1) > totalPulling(team2) ? team1 : team2 }

  var beat = function(team1,team2){winner(team1,team2) == team1}

  condition(beat(['bob'], ['tom']))

  return strength('bob')
}
```

compositionality

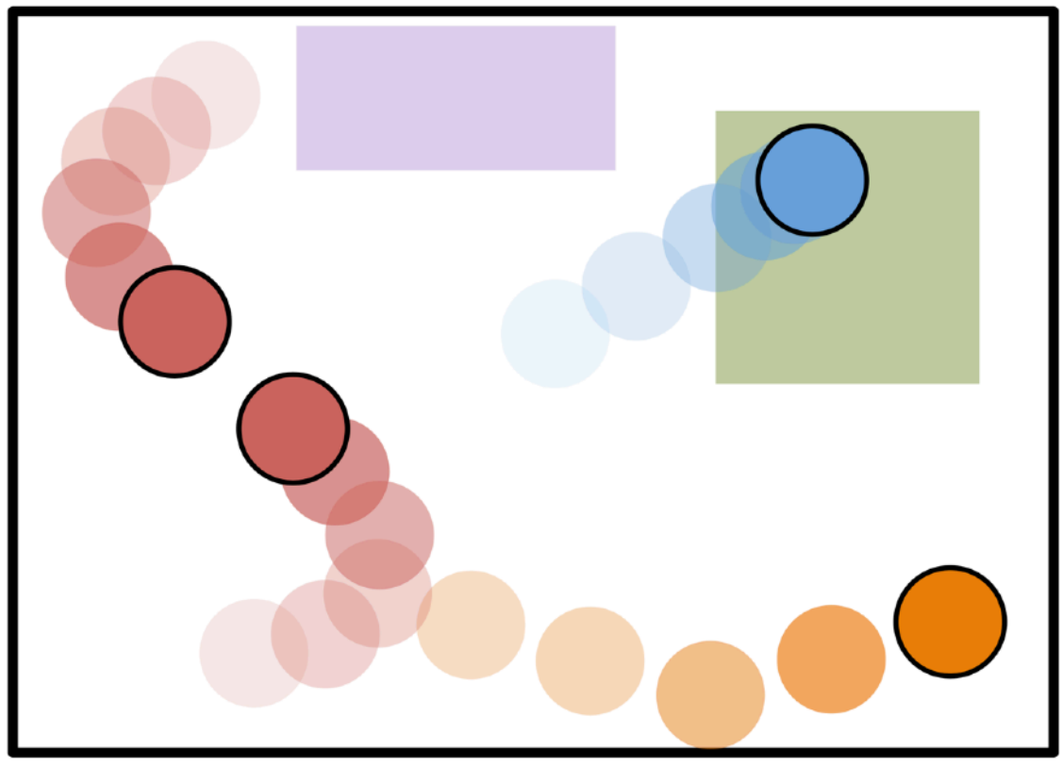


productivity

Some more cool examples: Program induction

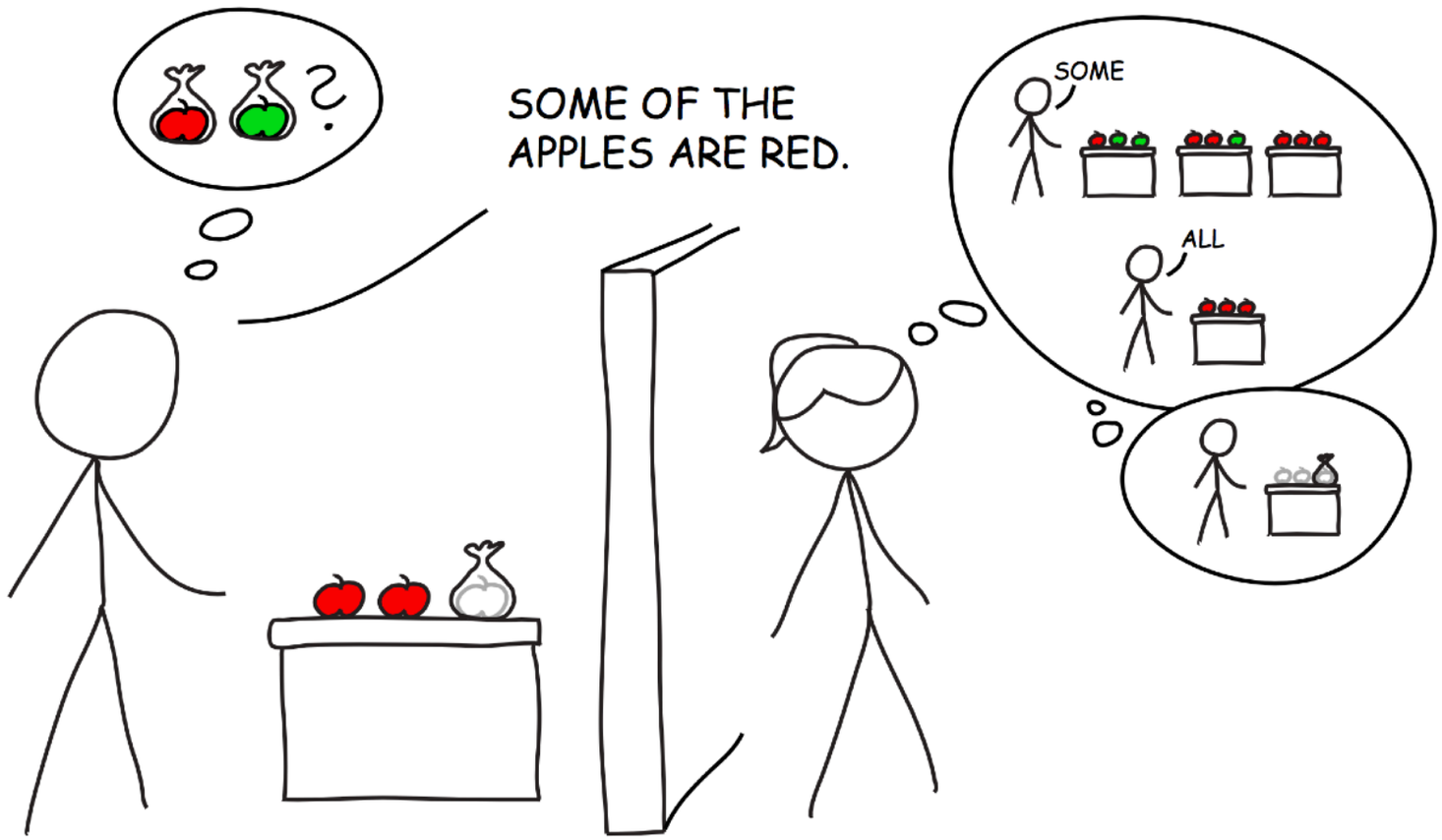


Learning at theory of physics



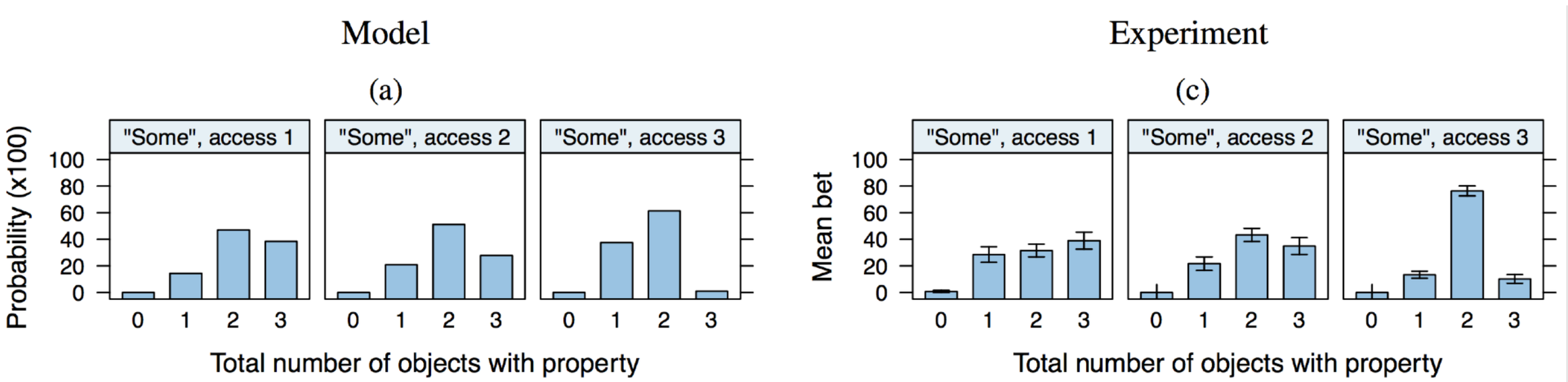
Ullman, Stuhlmüller, Goodman, & Tenenbaum (2014) Learning physics from dynamical scenes. Cognitive Science Proceedings

Some more cool examples: Pragmatic inference



Scalar implicature

("some" but not "all")



Goodman & Stuhlmüller (2013) Knowledge and implicature: Modeling language understanding as social cognition. Topics in Cognitive Science

Resources: theory

- Goodman, Mansinghka, Roy, Bonawitz, & Tenenbaum (2008) Church: A language for generative models. Uncertainty in Artificial Intelligence
  - ➡ https://stanford.edu/~ngoodman/papers/churchUAI08_rev2.pdf
- Goodman, Tenenbaum, & Gerstenberg (2015) Concepts in a probabilistic language of thought. The Conceptual Mind: New Directions in the Study of Concepts
  - ➡ web.mit.edu/tger/www/papers/Concepts in a probabilistic language of thought (Goodman, Tenenbaum, Gerstenberg, 2014).pdf
- Freer, Roy, & Tenenbaum (2012) Towards common-sense reasoning via conditional simulation: legacies of Turing in Artificial Intelligence. arXiv preprint arXiv:1212.4799
  - ➡ https://arxiv.org/pdf/1212.4799.pdf
- Lake, Ullman, Tenenbaum, & Gershman (2016) Building machines that learn and think like people. arXiv preprint arXiv:1604.00289
  - ➡ https://arxiv.org/pdf/1604.00289.pdf
- Gerstenberg & Tenenbaum (2017) Intuitive Theories. Oxford Handbook of Causal Reasoning
  - ➡ web.mit.edu/tger/www/papers/Intuitive Theories, Gerstenberg, Tenenbaum, 2017.pdf
- Chater & Oaksford (2013) Programs as causal models: Speculations on mental programs and mental representation. Cognitive Science
  - ➡ http://onlinelibrary.wiley.com/doi/10.1111/cogs.12062/abstract

Resources: practice

- https://probmods.org/
  - many more cool chapters to play around with
- http://webppl.org/
  - Editor to play around with code
- http://dippl.org/
  - Details about WebPPL
- https://github.com/probmods/webppl
  - github repository with latest developments
- http://webppl.readthedocs.io/en/master/
  - function reference for the webppl language
- http://agentmodels.org/
  - Great web book that focuses on how to model agents and inferences about agents
- http://probabilistic-programming.org/wiki/Home
  - homepage comparing different probabilistic programming languages

Thanks!

CENTER FOR
Brains
Minds+
Machines