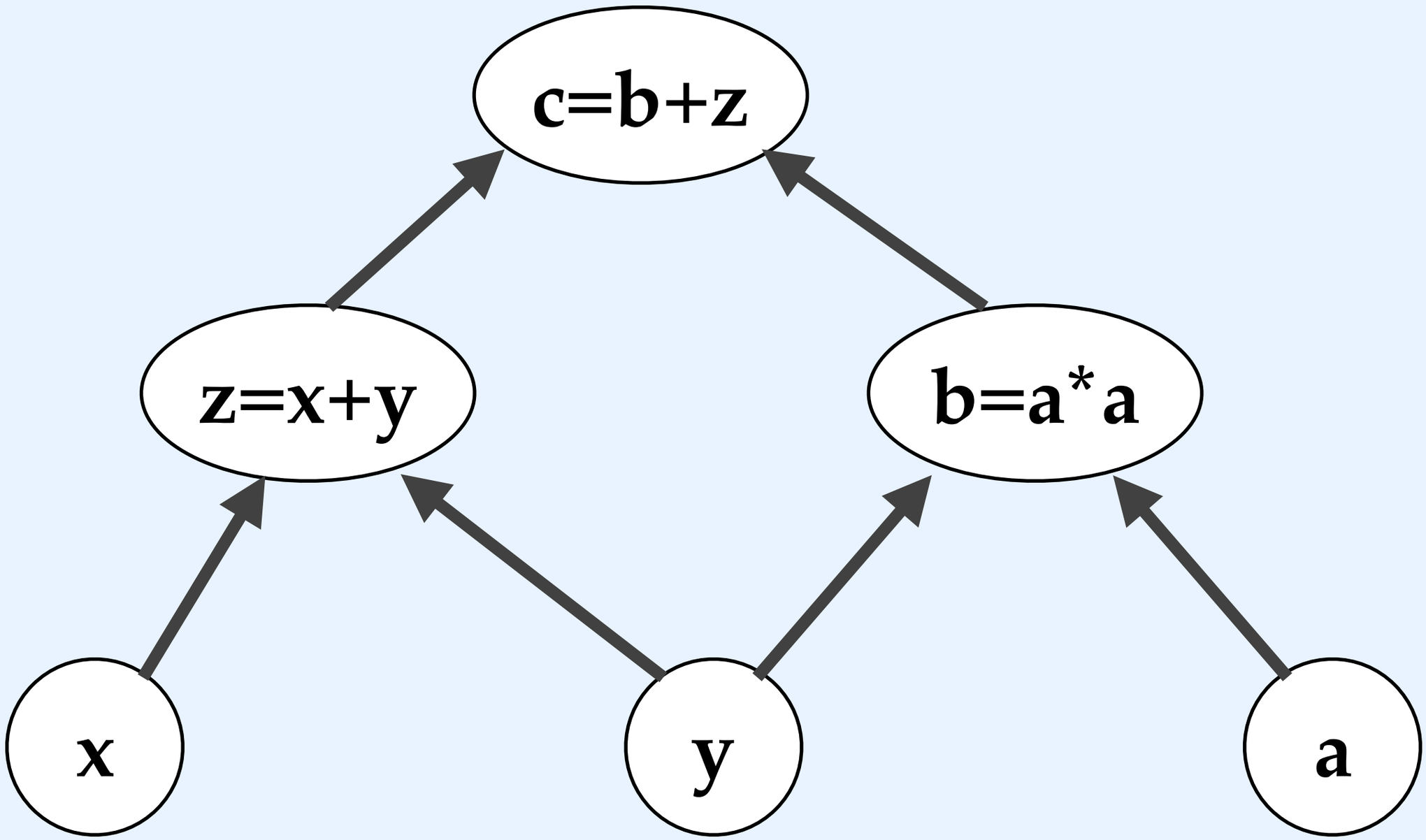
Computation Graph Languages

December 4, 2018

Computation Graph

A computation graph is a directed graph where nodes represent operations and variables and edges define the order of computation.

Computation Graph



Probabilistic Programming Languages

Probabilistic programming languages are useful for problems that require reasoning under uncertainty.

Key concept: the programs are probability models.

Probability Models

A probability model is a formal representation of a problem that involves non-determinism (randomness).

Three key parts: sample space, events, and probabilities for the events.

Lucky Charms cereal

Imagine that we draw a single piece of Lucky Charms cereal.

Source: www.village-bakery.com

Source: https://www.walmart.com

Lucky Charms cereal

Imagine that we draw two pieces of Lucky Charms cereal out of a bowl.

<u>Events:</u>

Picking a horseshoe + a rainbow

Picking a rainbow + a letter

Picking two letters

…

Source: www.village-bakery.com

Lucky Charms cereal

Sample space (set of all outcomes):

horseshoe + rainbow

rainbow + a letter

two letters

two rainbows

two horseshoes

horseshoe + letter

….

Lucky Charms cereal

Probabilities:

p(horseshoe) = 0.1

p(rainbow) = 0.2

p(letter) = 0.7

p(shooting star) = 0.0

p(balloon) = 0.0

…

The probability of the sample space always sums to 1.

Probabilistic Programming Languages

Key concept: the programs are probability models.

PPLs have stochastic elements whose values are sampled on every run of the program. The meaning of the program is the probability of every possible execution of the program.

Probabilistic Programming Languages

Two main modes: prediction and inference. Prediction uses observed causes to guess unseen results; inference uses observed results to try to understand unseen causes.

Probabilistic Programs as Computation Graphs

From a computation graph point-of-view, prediction is the forward propagation of data through the graph, while inference is the backwards propagation of data.

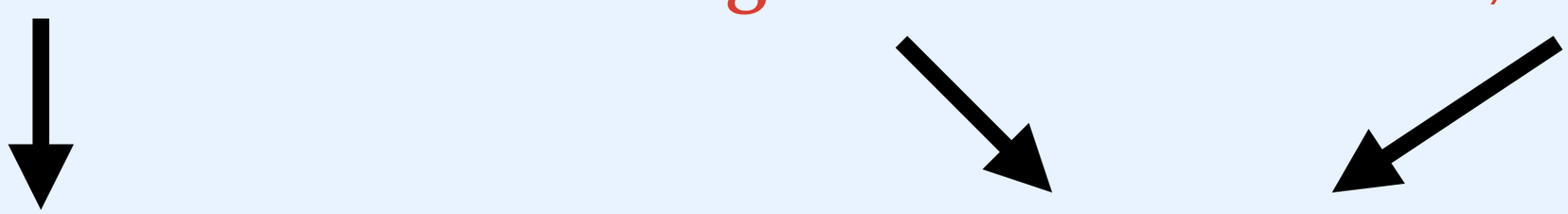
Figaro

Figaro is a probabilistic programming language that uses Scala syntax.

This is another example of a domain-specific language embedded in a general-purpose language.

Elements

Scala variable **Figaro element** **True with 20% chance**

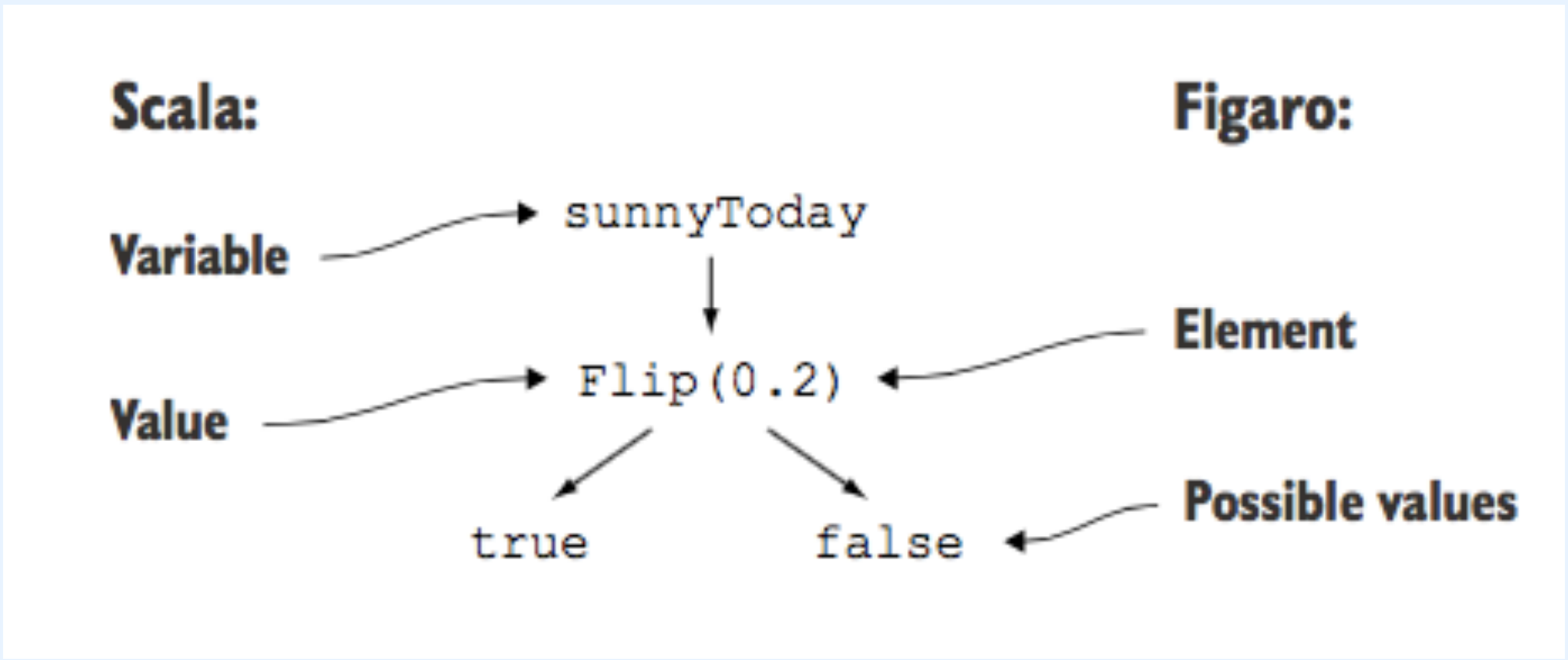


```
val sunnyToday = Flip(0.2)
```

`Flip(0.2)` is an instance of `Element[Boolean]`.

Elements

```
val sunnyToday = Flip(0.2)
```



Morning greeting application

Every morning, I wake up, lean out my window, and shout a greeting.

When the weather is good, I usually say, “Hello world!” or “Howdy, universe!”.

When the weather is bad, I’m grumpier. Sometimes I say, “Hello world!”, but sometimes I say, “Oh no, not again.”

Today's weather		
Sunny	0.2	
Not sunny	0.8	
Today's greeting		
If today's weather is sunny	"Hello, world!"	0.6
	"Howdy, universe!"	0.4
If today's weather isn't sunny	"Hello, world!"	0.2
	"Oh no, not again"	0.8
Tomorrow's weather		
If today's weather is sunny	Sunny	0.8
	Not sunny	0.2
If today's weather isn't sunny	Sunny	0.05
	Not sunny	0.95
Tomorrow's greeting		
If tomorrow's weather is sunny	"Hello, world!"	0.6
	"Howdy, universe!"	0.4
If tomorrow's weather isn't sunny	"Hello, world!"	0.2
	"Oh no, not again"	0.8

Morning greeting application

Let's see how we would model two days of my morning routine.

There are three tasks that we want our model to be able to do:

1. Predict the greeting today
2. Given an observation of the greeting, infer the weather
3. Learn from an observation of today's greeting in order to predict tomorrow's greeting.

Elements

An element is a language construct that represents a process that probabilistically produces a value.

Like Variables in Tensorflow, the value of an element isn't known until the computation graph is run.

Unlike Tensorflow Variables, however, Figaro elements let you specify the probabilistic process used to sample a value more explicitly.

Probabilistic Programming Languages

Figaro is one of the newest, most powerful probabilistic programming languages, partly because of its strong interface with Scala.

Older statistical modeling languages:

STAN, BayesiaLab

Others include:

Church, WebPPL

Applications

Probabilistic programming languages are relatively new (~2000), and we're still figuring out useful applications for them.

Example applications:

- evaluating online game players (Microsoft)
- identifying nuclear test treaty violations (Stuart Russell)
- identifying malware (Charles River Analytics)
- modeling language learning and conversation dynamics (various folks at Stanford and MIT)