Guided Tour of Machine Learning in Finance

Week 2-Lesson 1-part 1: DataFlow and TensorFlow

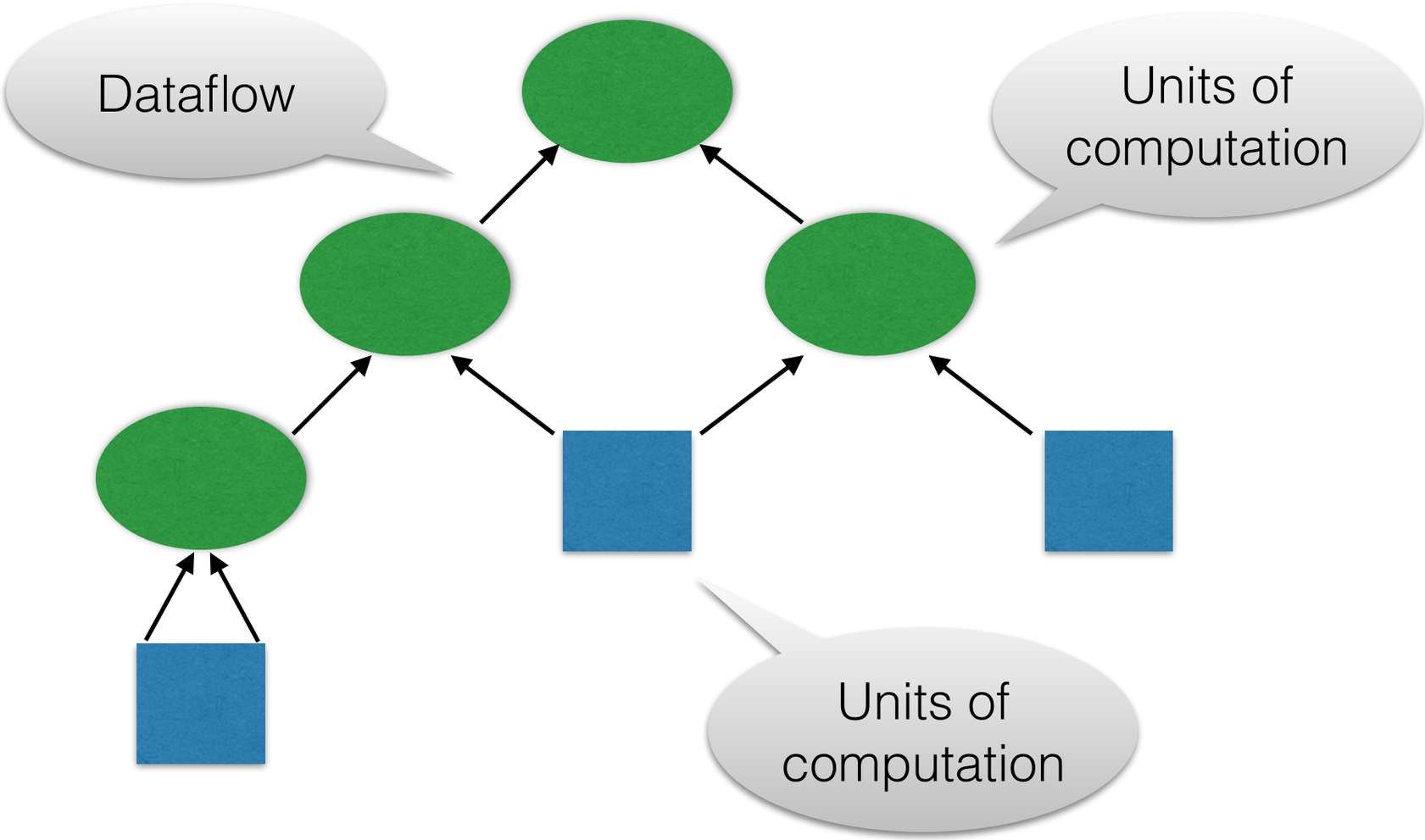
Igor Halperin

NYU Tandon School of Engineering, 2017

Dataflow graph

Dataflow programming model for parallel computing:

- Nodes represent units of computation
- Edges represent data consumed or produced by a node

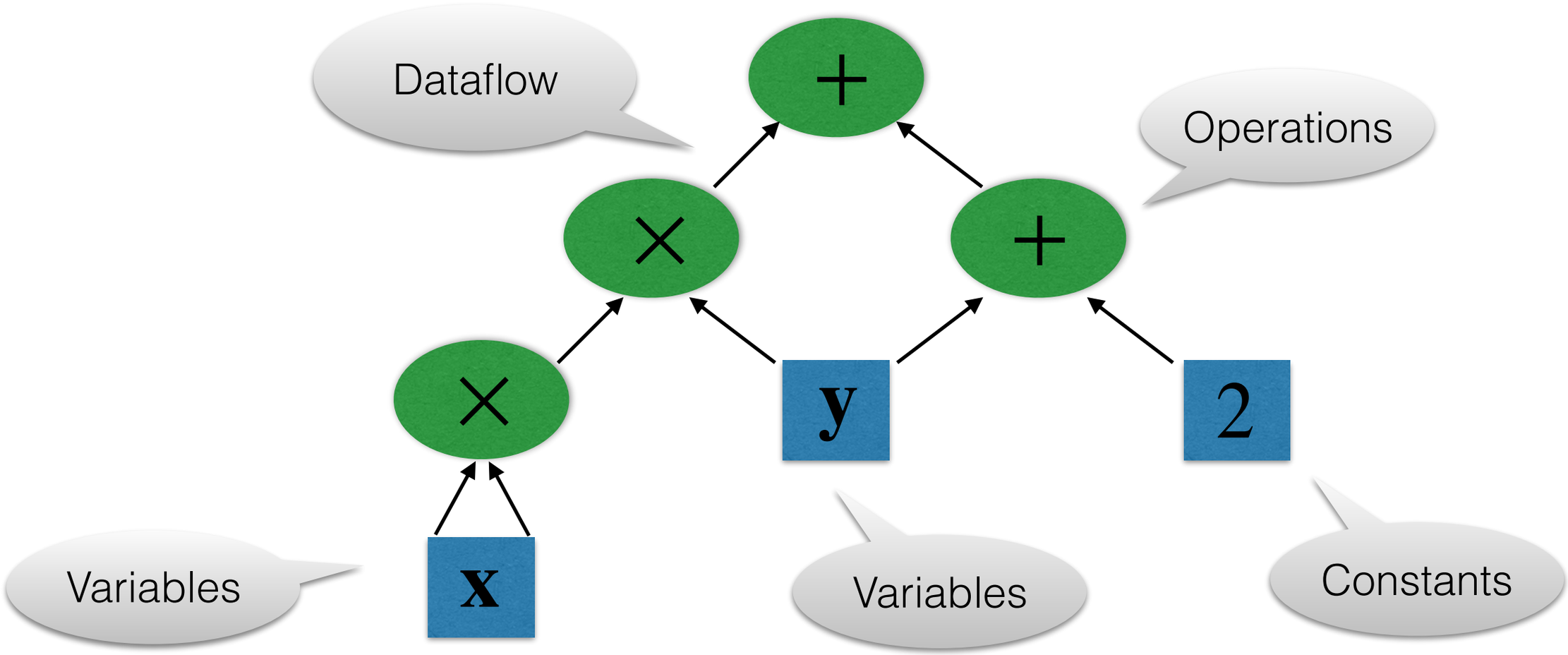


Dataflow graph

Dataflow graph nodes:

- Variables
- Operations
- Constants

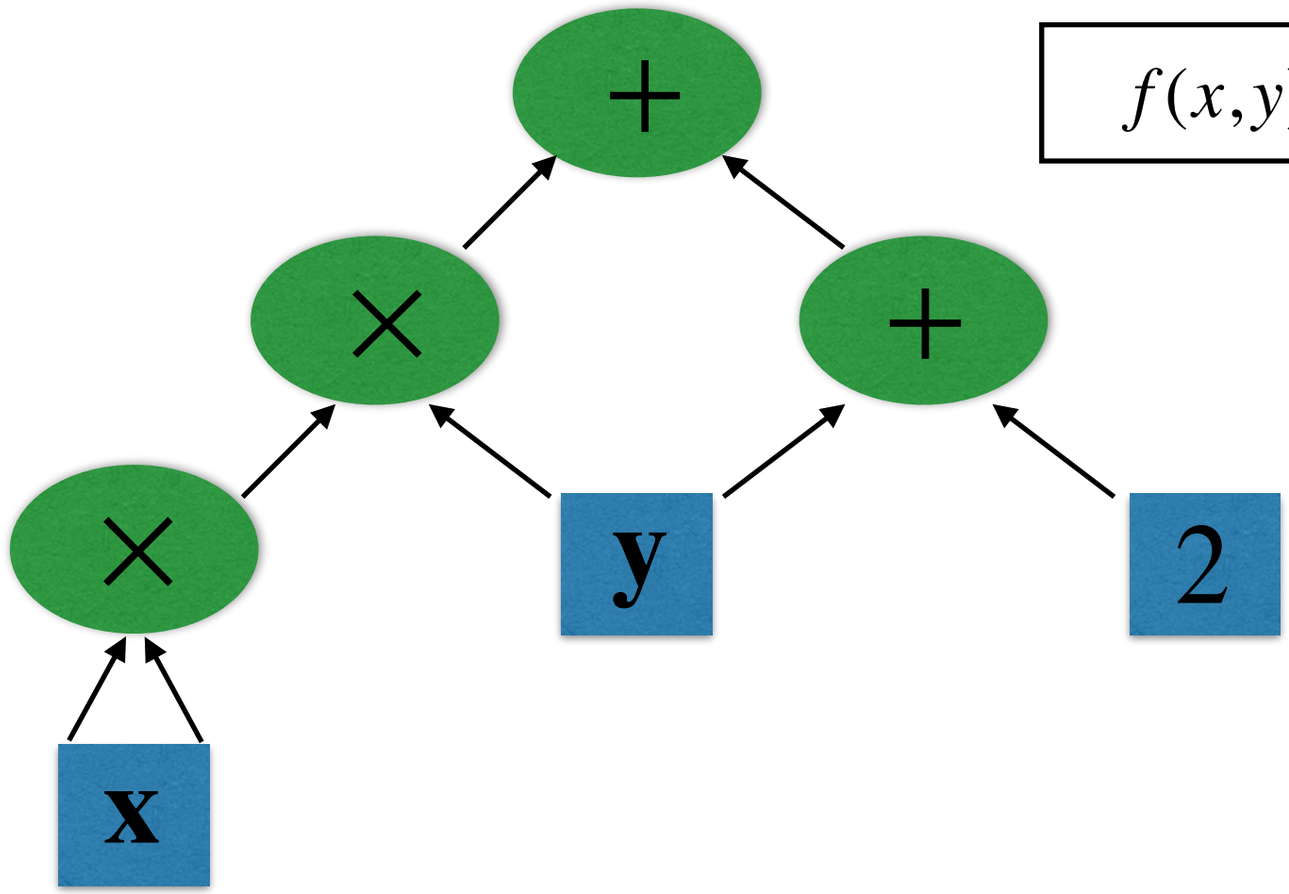
$$f(x,y) = x^2 y + y + 2$$



Dataflow graph with tensors

Tensors:
Rank n= 0: a number
Rank n=1: a list of numbers
Rank n=2: a list of lists
Rank n=3: a list of lists of lists…

Implemented as a Nd-array in numpy



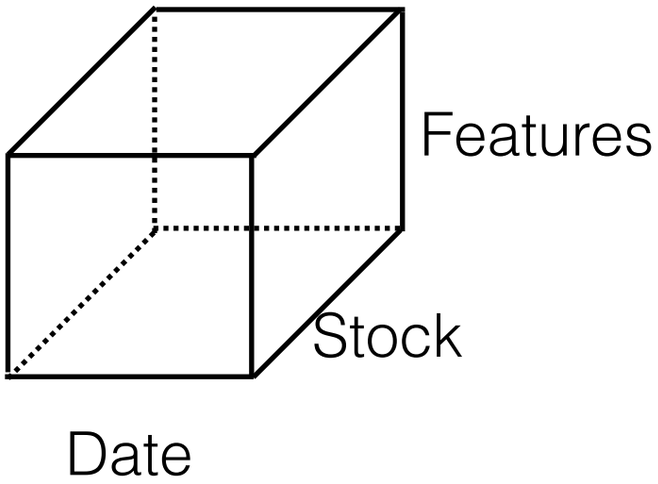
$$f(x,y) = x^2 y + y + 2$$

Dataflow graph with tensors

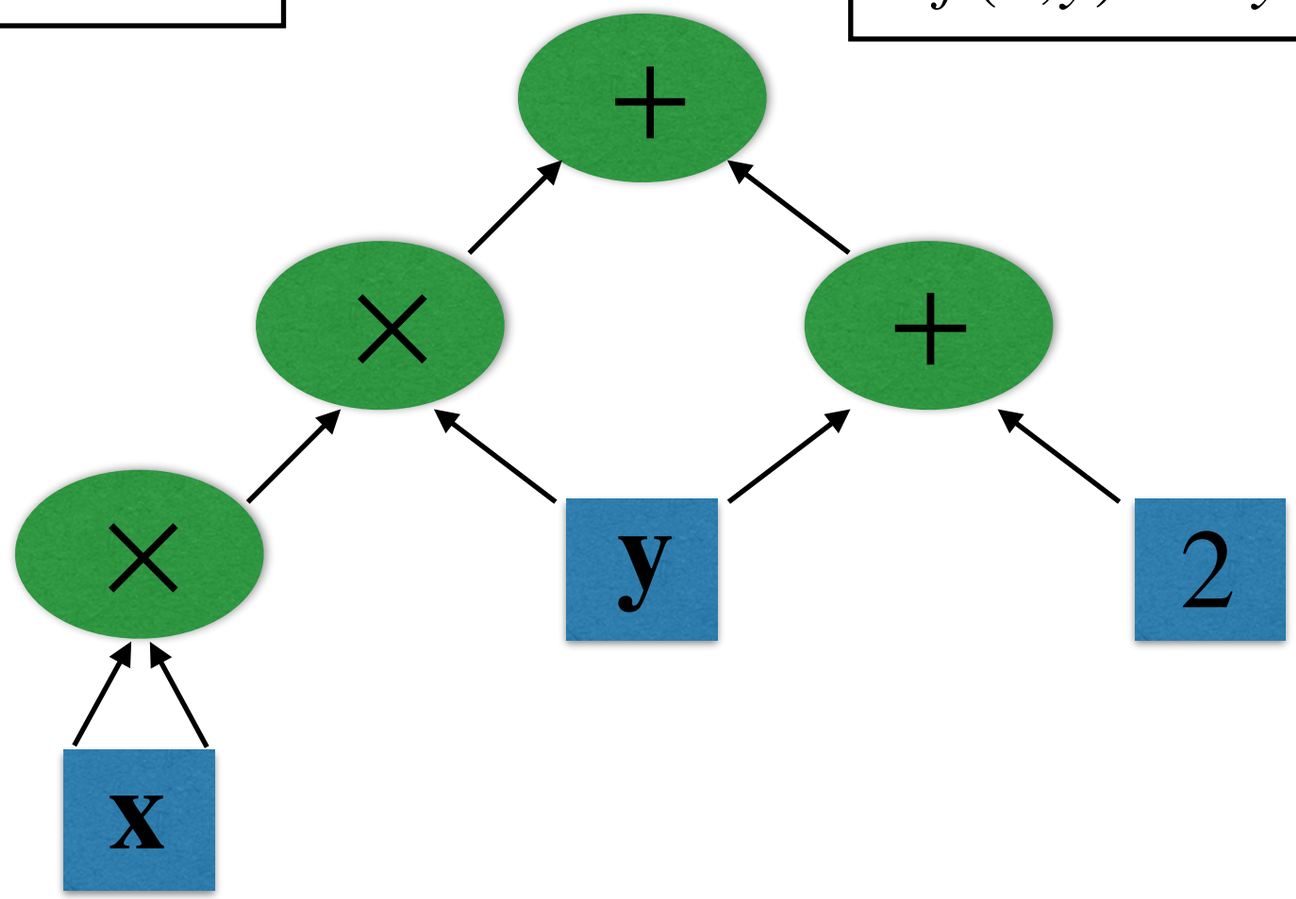
Tensors:
Rank n= 0: a number
Rank n=1: a list of numbers
Rank n=2: a list of lists
Rank n=3: a list of lists of lists…

Implemented as a Nd-array in numpy

Stock data as a rank-3 tensor:



$$f(x,y) = x^2 y + y + 2$$

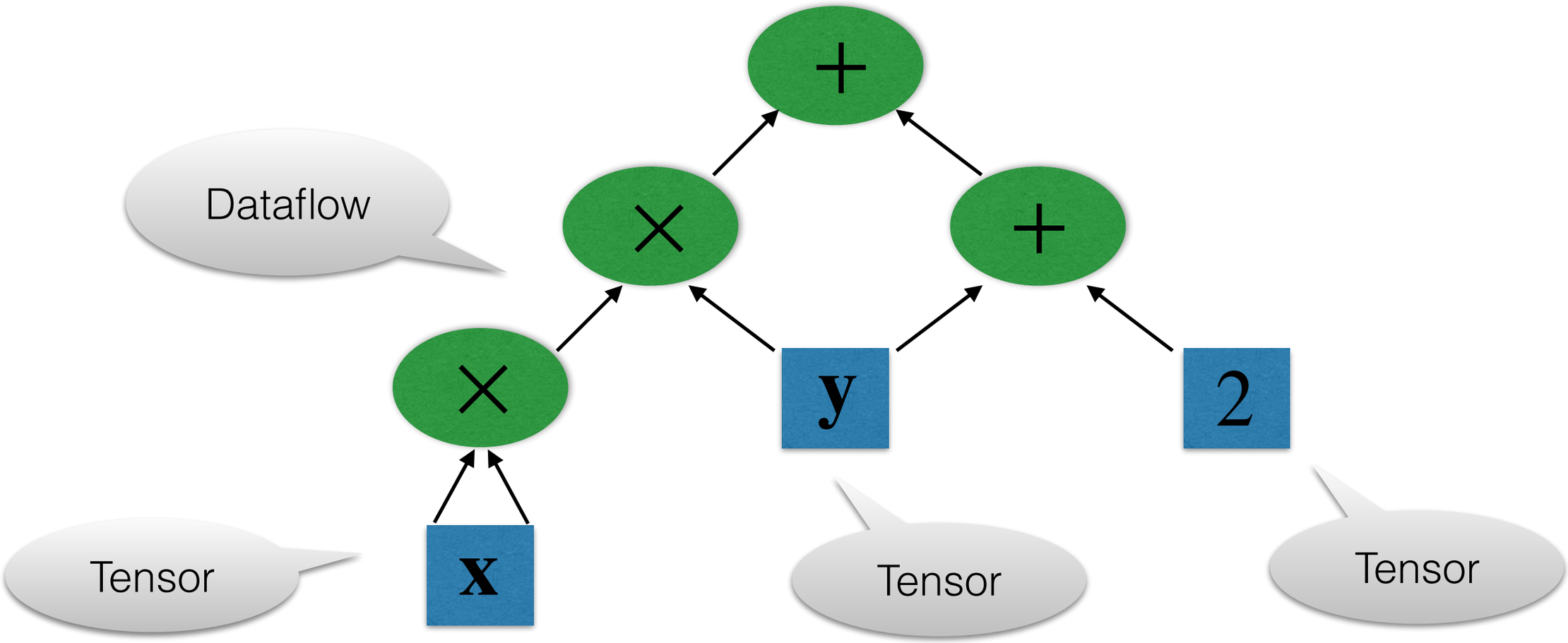


TensorFlow

TensorFlow:

- Construct a a dataflow graph (in Python)
- Run the graph using optimized C++ code
- Open-sourced by Google in 2015

Tensors + Dataflow = TensorFlow!

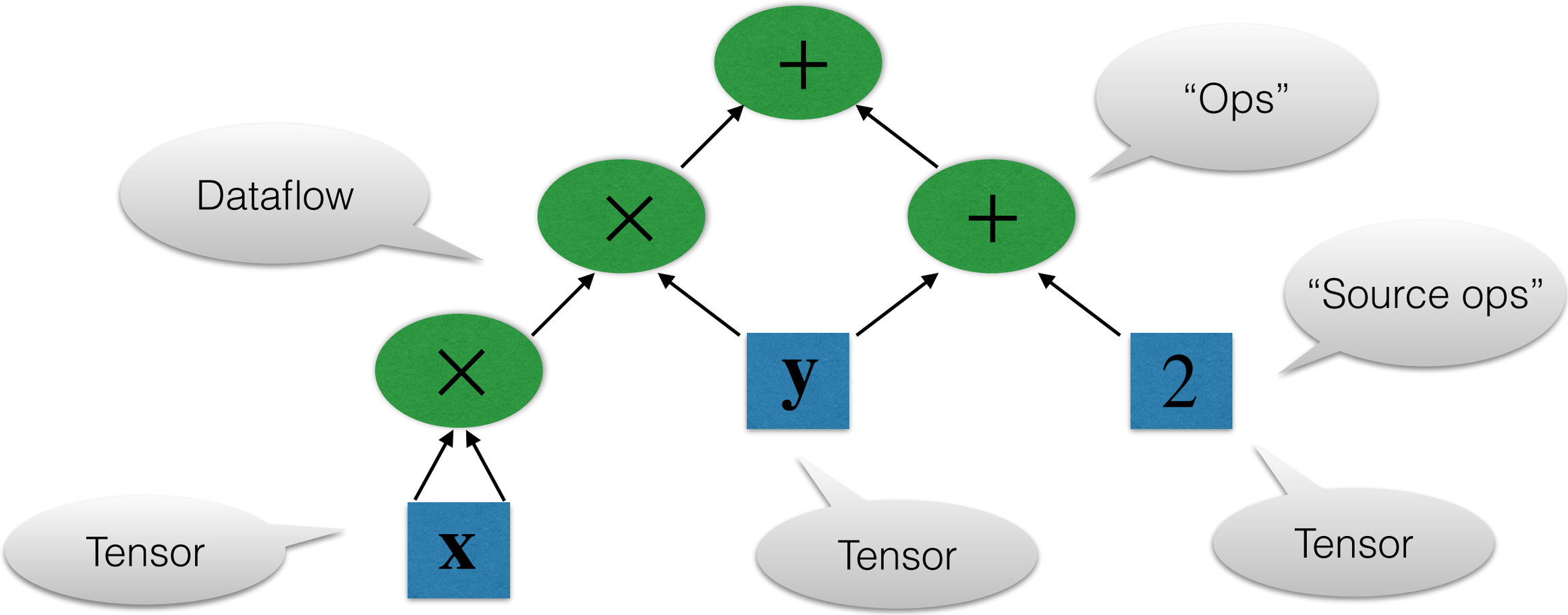


TensorFlow: lazy evaluation

TensorFlow:

- Construct a a dataflow graph (in Python) (**no evaluation yet**!)
- Run the graph using optimized C++ code (**execute to evaluate** the graph)

Tensors + Dataflow = TensorFlow!



TensorFlow: Reverse-Mode Autodiff

Reverse-mode autodiff implements automatic derivatives of any functions**:**

- **Forward pass** (from inputs to outputs)
- **Backward pass** (from outputs to inputs)

TensorFlow: Reverse-Mode Autodiff

Reverse-mode autodiff implements automatic derivatives of any functions**:**

- **Forward pass** (from inputs to outputs)
- **Backward pass** (from outputs to inputs)
- Use the chain rule for a composite function $f = f\left(n_i(x)\right)$

$$\boxed{\frac{\partial f}{\partial x} = \frac{\partial f}{\partial n_i} \cdot \frac{\partial n_i}{\partial x}}$$

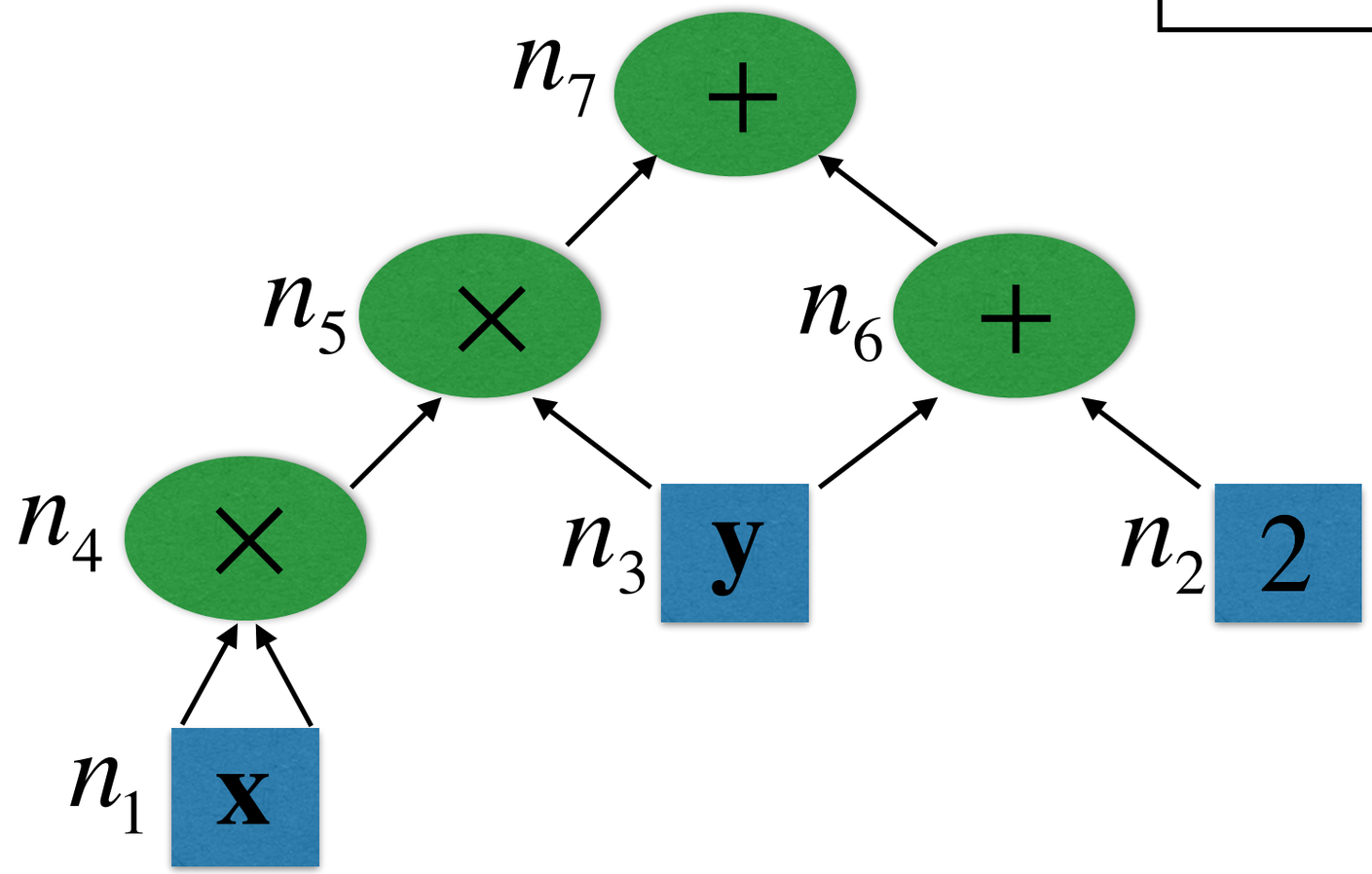
TensorFlow: Reverse-Mode Autodiff

Reverse-mode autodiff implements automatic derivatives of any functions**:**

- **Forward pass** (from inputs to outputs)
- **Backward pass** (from outputs to inputs)
- Use the chain rule for a composite function $f = f\left(n_i(x)\right)$

$$\frac{\partial f}{\partial x} = \frac{\partial f}{\partial n_i} \cdot \frac{\partial n_i}{\partial x}$$

$$f(x,y) = x^2 y + y + 2$$



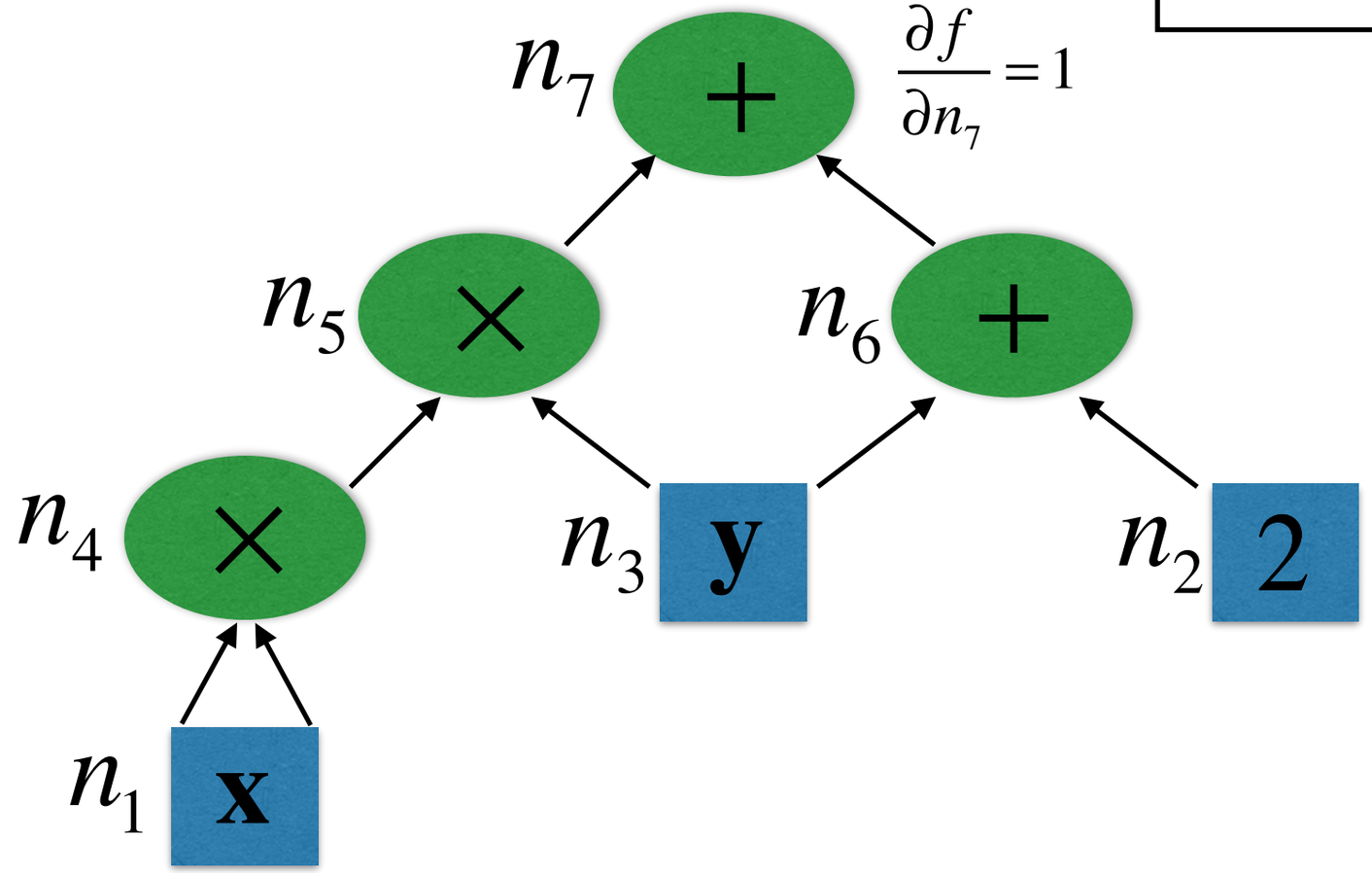
TensorFlow: Reverse-Mode Autodiff

Reverse-mode autodiff implements automatic derivatives of any functions:

- **Forward pass** (from inputs to outputs)
- **Backward pass** (from outputs to inputs)
- Use the chain rule for a composite function $f = f\left(n_i(x)\right)$

$$\frac{\partial f}{\partial x} = \frac{\partial f}{\partial n_i} \cdot \frac{\partial n_i}{\partial x}$$

$$f(x,y) = x^2 y + y + 2$$



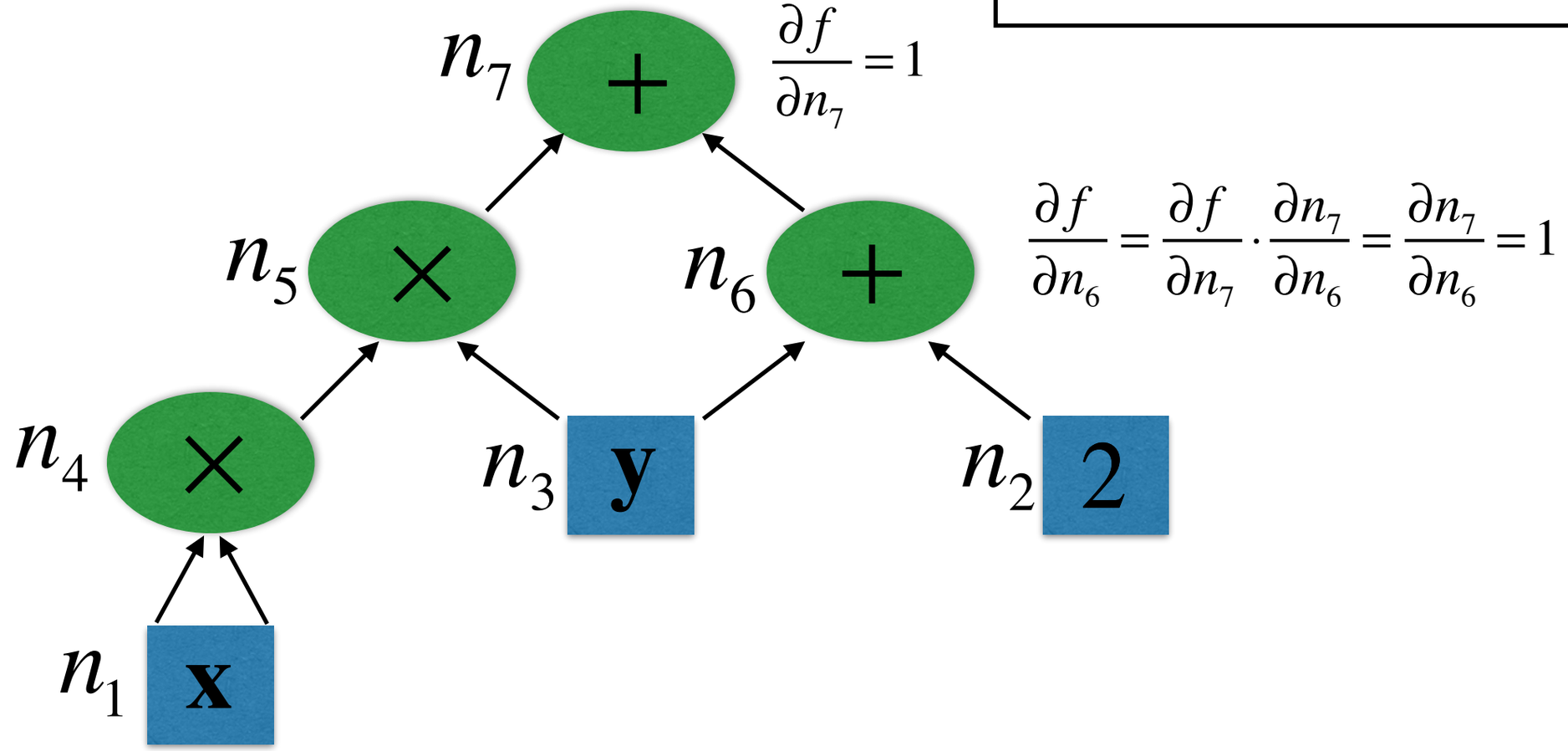
TensorFlow: Reverse-Mode Autodiff

Reverse-mode autodiff implements automatic derivatives of any functions:

- **Forward pass** (from inputs to outputs)
- **Backward pass** (from outputs to inputs)
- Use the chain rule for a composite function $f = f\left(n_i(x)\right)$

$$\frac{\partial f}{\partial x} = \frac{\partial f}{\partial n_i} \cdot \frac{\partial n_i}{\partial x}$$

$$f(x,y) = x^2 y + y + 2$$



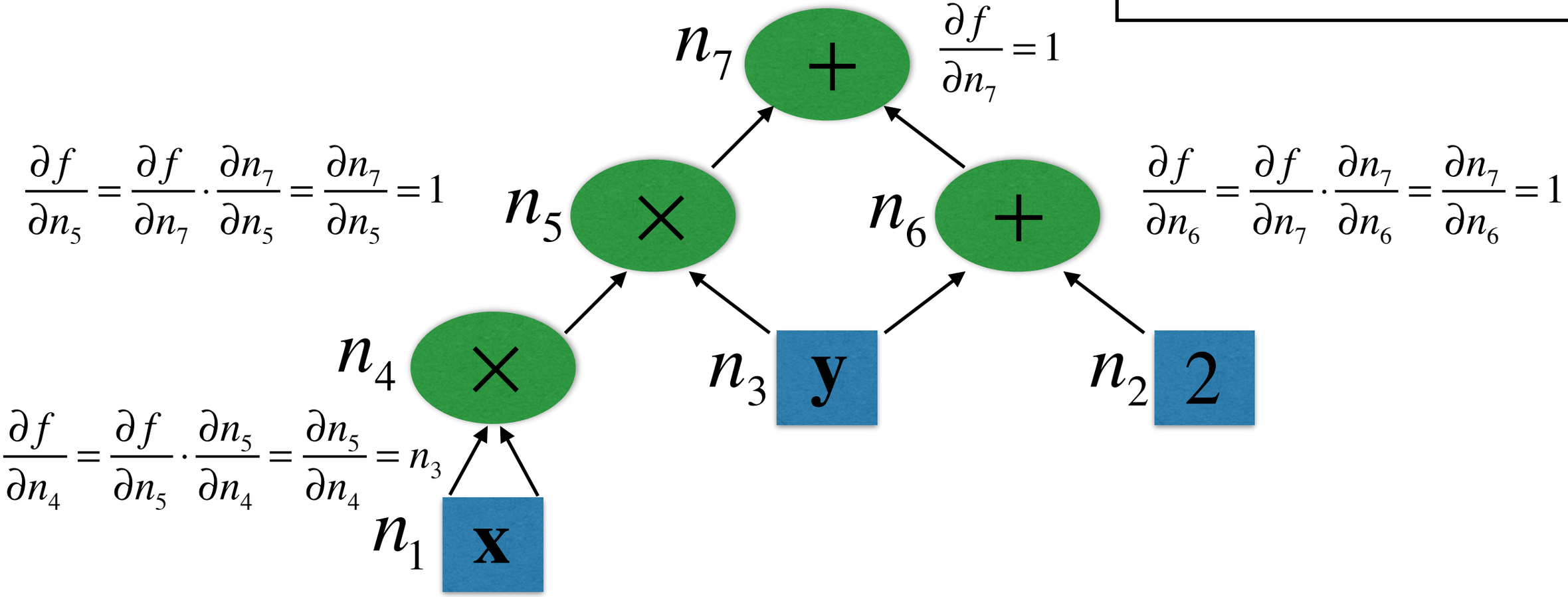
TensorFlow: Reverse-Mode Autodiff

Reverse-mode autodiff implements automatic derivatives of any functions**:**

- **Forward pass** (from inputs to outputs)
- **Backward pass** (from outputs to inputs)
- Use the chain rule for a composite function $f = f\left(n_i(x)\right)$

$$\frac{\partial f}{\partial x} = \frac{\partial f}{\partial n_i} \cdot \frac{\partial n_i}{\partial x}$$

$$f(x,y) = x^2 y + y + 2$$



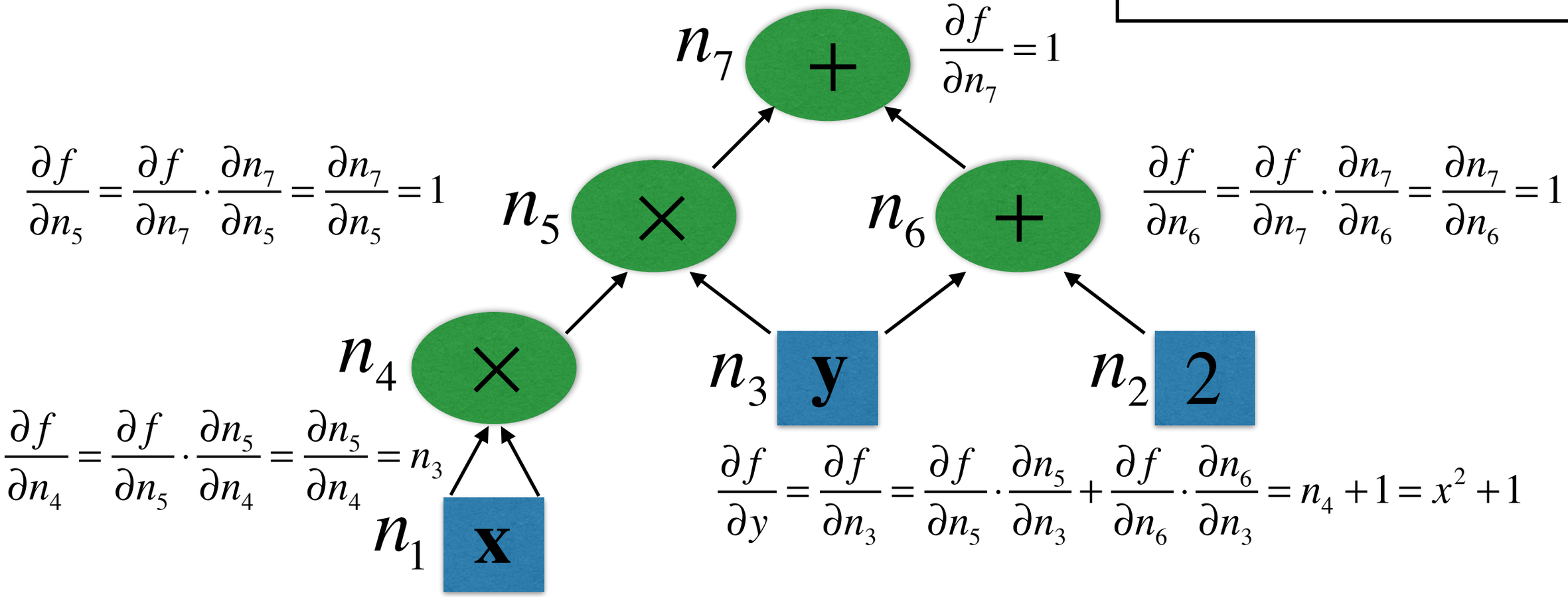
TensorFlow: Reverse-Mode Autodiff

Reverse-mode autodiff implements automatic derivatives of any functions:

- **Forward pass** (from inputs to outputs)
- **Backward pass** (from outputs to inputs)
- Use the chain rule for a composite function $f = f\left(n_i(x)\right)$

$$\frac{\partial f}{\partial x} = \frac{\partial f}{\partial n_i} \cdot \frac{\partial n_i}{\partial x}$$

$$f(x,y) = x^2 y + y + 2$$



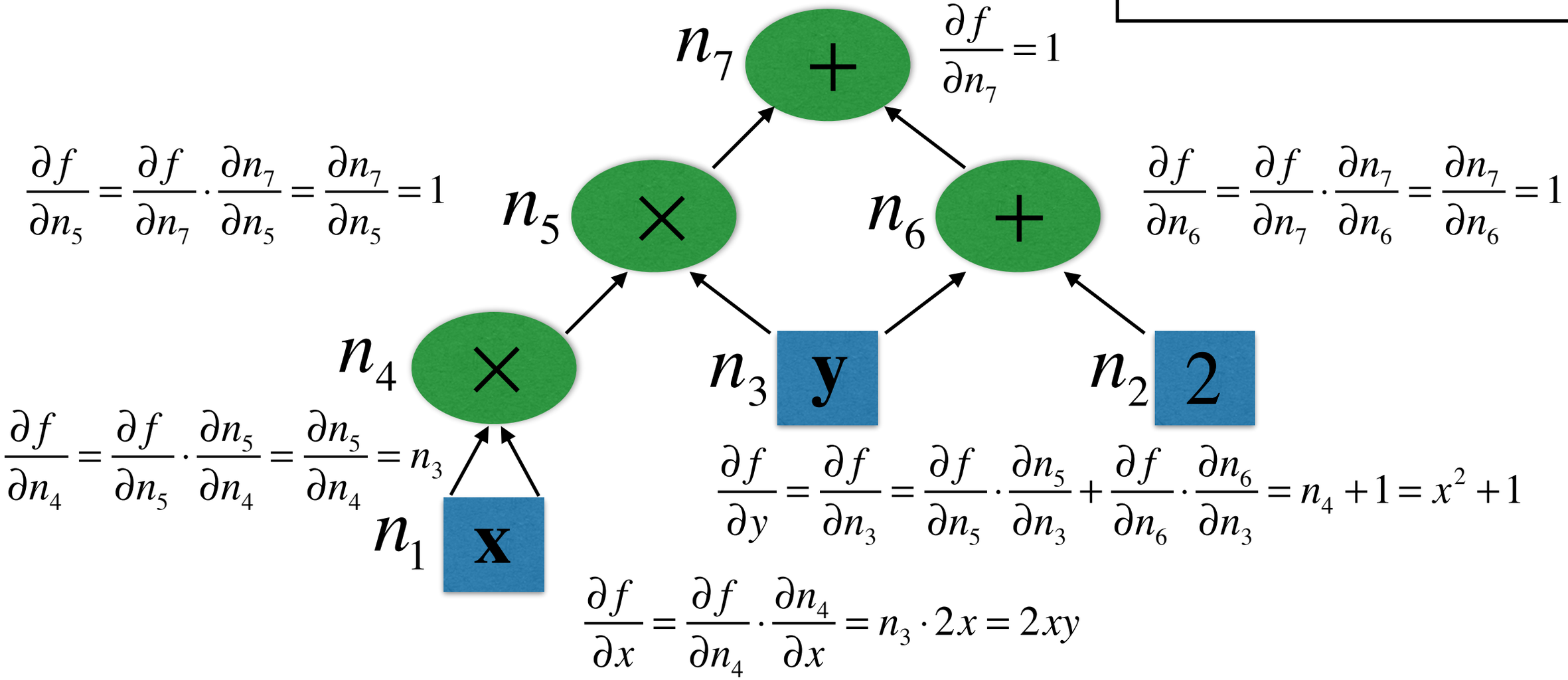
TensorFlow: Reverse-Mode Autodiff

Reverse-mode autodiff implements automatic derivatives of any functions**:**

- **Forward pass** (from inputs to outputs)
- **Backward pass** (from outputs to inputs)
- Use the chain rule for a composite function $f = f\left(n_i(x)\right)$

$$\frac{\partial f}{\partial x} = \frac{\partial f}{\partial n_i} \cdot \frac{\partial n_i}{\partial x}$$

$$f(x,y) = x^2 y + y + 2$$



Control question

Q: In the last relation $\frac{\partial f}{\partial x} = \frac{\partial f}{\partial n_4} \cdot \frac{\partial n_4}{\partial x} = n_3 \cdot 2x = 2xy$, what is the origin of the factor 2?
Select all correct answers:

1. The factor 2 arises because there are two variables in the problem, x and y.
2. It arises because our graph contains exactly two hierarchical levels not counting the last output level. By convention, all derivatives in TensorFlow are multiplied by the depth of the tree, so that this parameter would be easy to extract from the final result.
3. It arises because node n_4 multiplies x by itself. Differentiation with respect to the first x gives the second x, but we can also differentiate the second x, which will produce the first x. Therefore we will have $\frac{\partial n_4}{\partial x} = 2x$
4. This is because $\frac{d}{dx} x^2 = 2x$

Correct answer: 3 and 4.

Why TensorFlow?

- Flexibility and clean design
- Automatic differentiation
- Great visualization via TensorBoard
- Good documentation
- Supported by Google and a wide community of developers
- Scalable and production-ready
- Portable to other operating systems and devices (e.g. on mobile devices)
- Many high-level APIs (TF.Learn, Keras, etc.)
- We will be using TF primarily for neural network models, but will start low, with simple demos and implementation of simple algorithms such as linear regression.