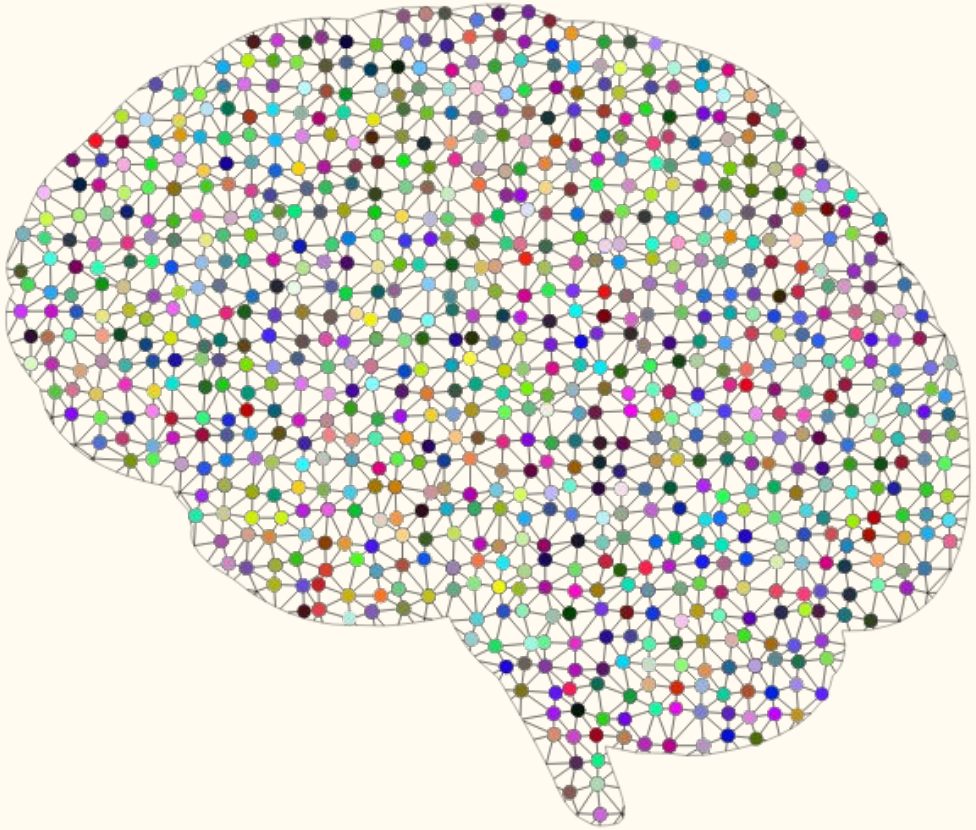
Computational Graph

Nhóm 8



Khái niệm

Computational Graph

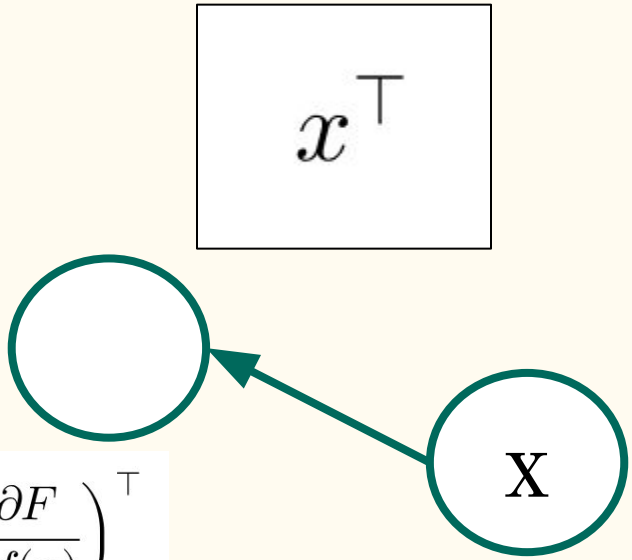
là một loại đồ thị có hướng,

trong đó các nút tương ứng với
các phép toán hoặc biến số.

Mỗi nút là một giá trị
tensor, ma trận, vector hoặc vô hướng (scalar).

x

Mỗi cạnh đại diện cho đối số của hàm.

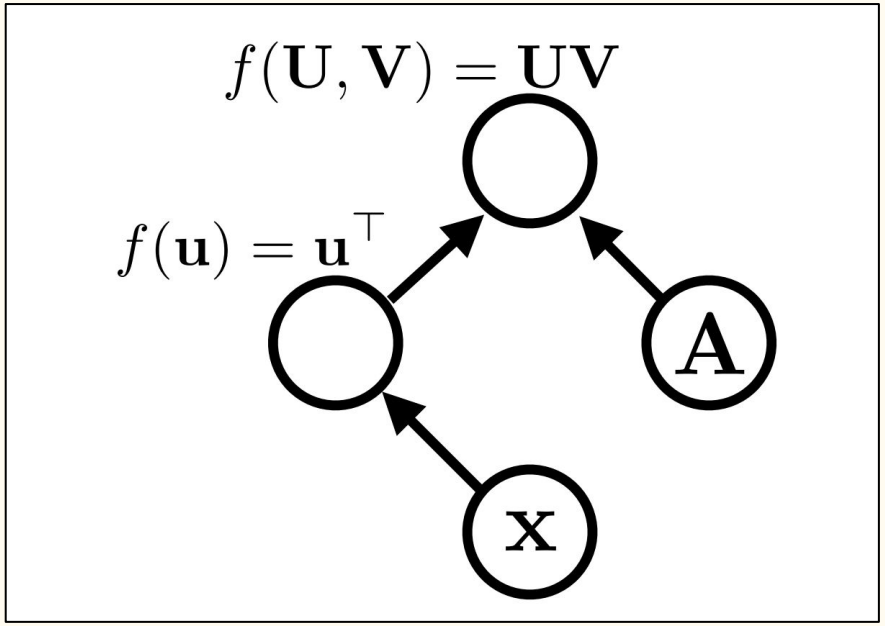


$$f(u) = u^\top$$

$$\frac{\partial f(u)}{\partial u}\frac{\partial F}{\partial f(u)} = \left(\frac{\partial F}{\partial f(u)}\right)^\top$$

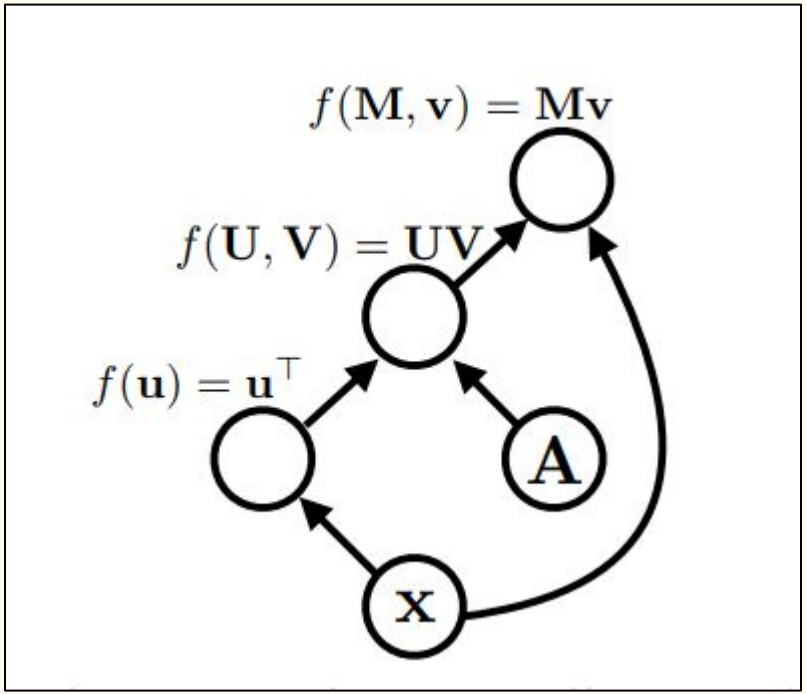
Hàm số thường có một đến hai đối số.

$$\mathbf{x}^\top \mathbf{A}$$



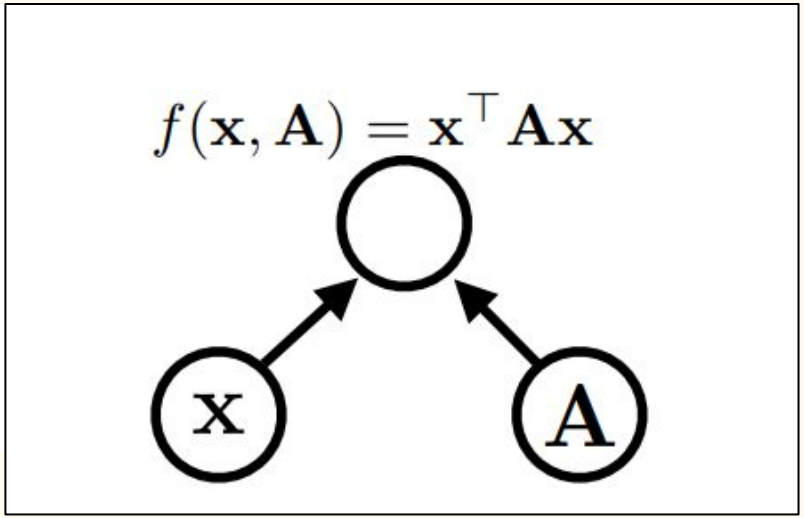
Thường có hướng và không tuần hoàn

$$\mathbf{x}^\top \mathbf{A} \mathbf{x}$$



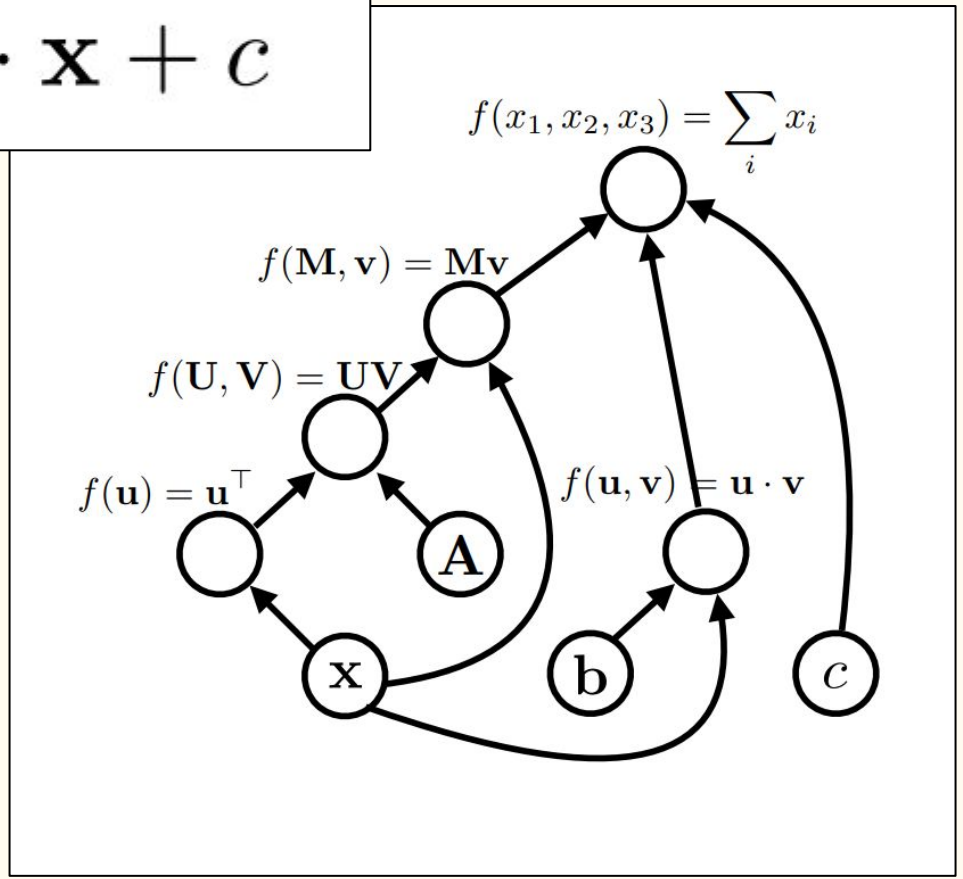
$$\frac{\partial f(\mathbf{x}, \mathbf{A})}{\partial \mathbf{x}} = (\mathbf{A}^\top + \mathbf{A})\mathbf{x}$$

$$\frac{\partial f(\mathbf{x}, \mathbf{A})}{\partial \mathbf{A}} = \mathbf{x}\mathbf{x}^\top$$

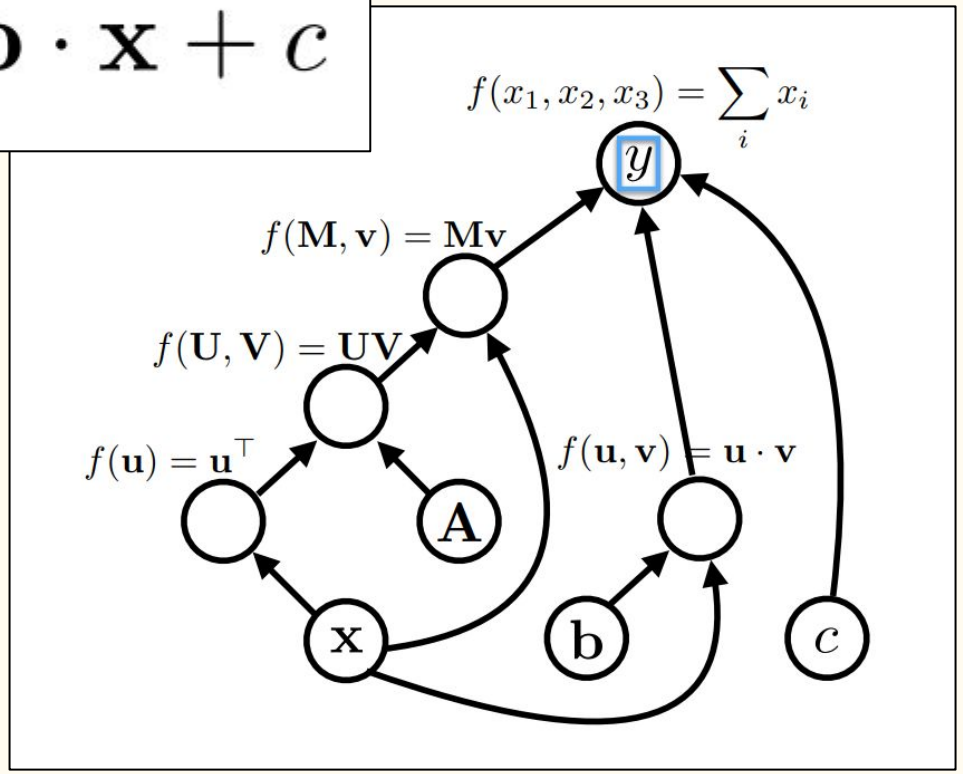


$$\mathbf{x}^\top \mathbf{A}\mathbf{x}$$

$$\mathbf{x}^\top \mathbf{A}\mathbf{x} + \mathbf{b} \cdot \mathbf{x} + c$$



$$y = \mathbf{x}^\top \mathbf{A}\mathbf{x} + \mathbf{b} \cdot \mathbf{x} + c$$



Tên biến ở đây dùng để dán nhãn cho các nút.

Sự cần thiết

Sự cần thiết

- Lập lịch dựa trên sự phụ thuộc
- Tối ưu hóa đồ thị
- Vi phân tự động

Cách tổ chức và vận hành

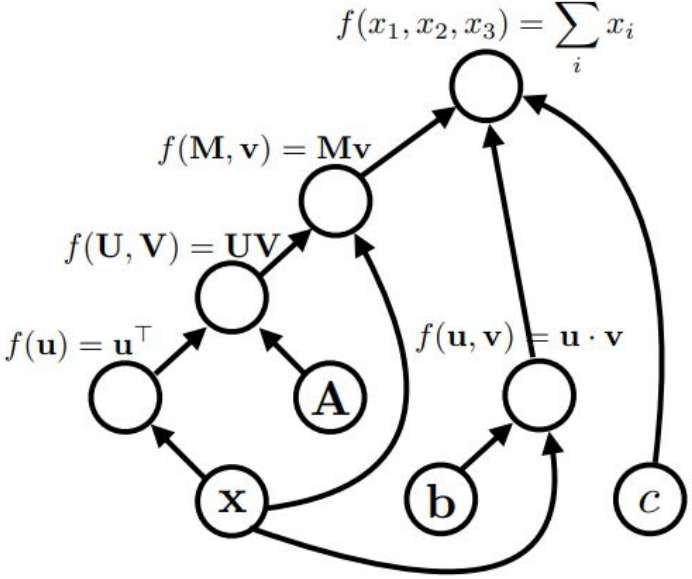
Giải thuật

- Khởi tạo đồ thị
- Lan truyền xuôi (Forward Propagation)
  - Đi qua các nút theo thứ tự tôpô, bắt đầu từ các nút đầu vào.
  - Tính giá trị của mỗi nút theo đầu vào đã cho.
  - Đưa ra kết quả dự đoán hoặc lỗi so với đầu ra mục tiêu.
- Lan truyền ngược (Backward Propagation hay Backpropagation)
  - Chạy qua các nút theo thứ tự tôpô ngược, bắt đầu từ nút đích cuối cùng.
  - Tính đạo hàm của nút đích cuối cùng ứng với nút đuôi của từng cạnh.
  - Tính được đạo hàm của nút đích ứng với từng nút đầu vào.

Thay đổi nhỏ ở đầu vào sẽ ảnh hưởng đến đầu ra thế nào ?

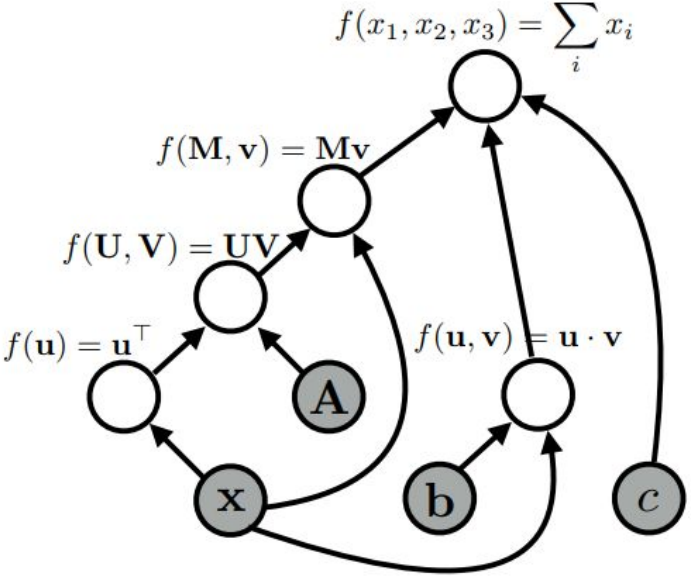
Forward Propagation

graph:



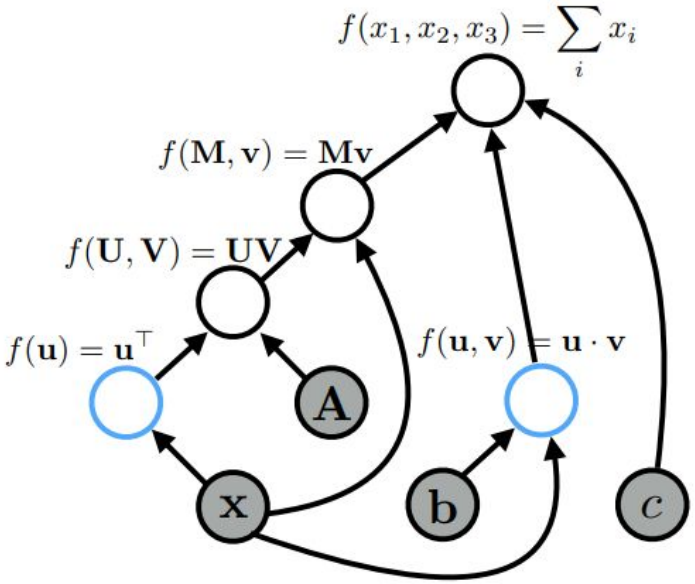
Forward Propagation

graph:



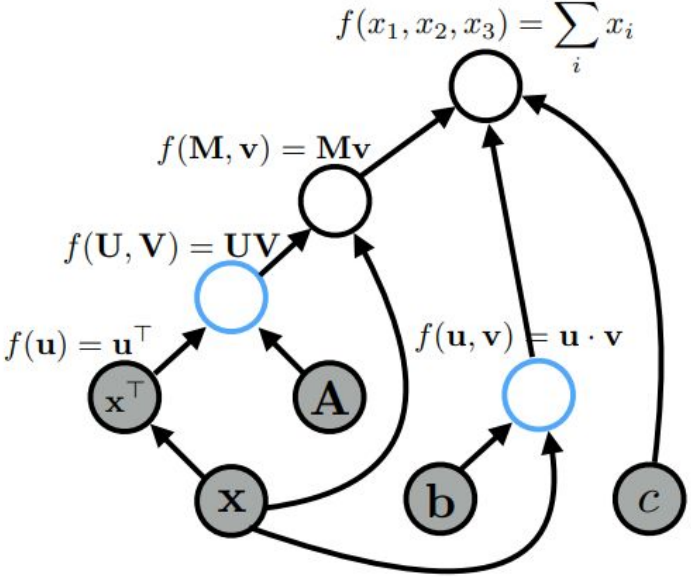
Forward Propagation

graph:



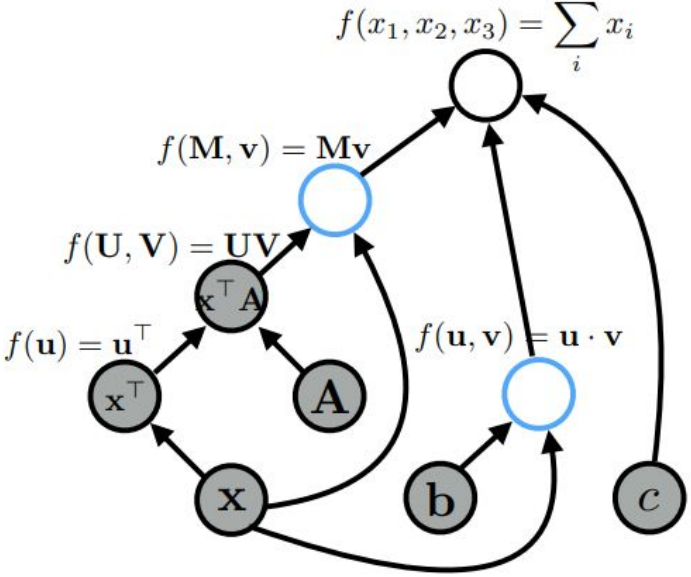
Forward Propagation

graph:



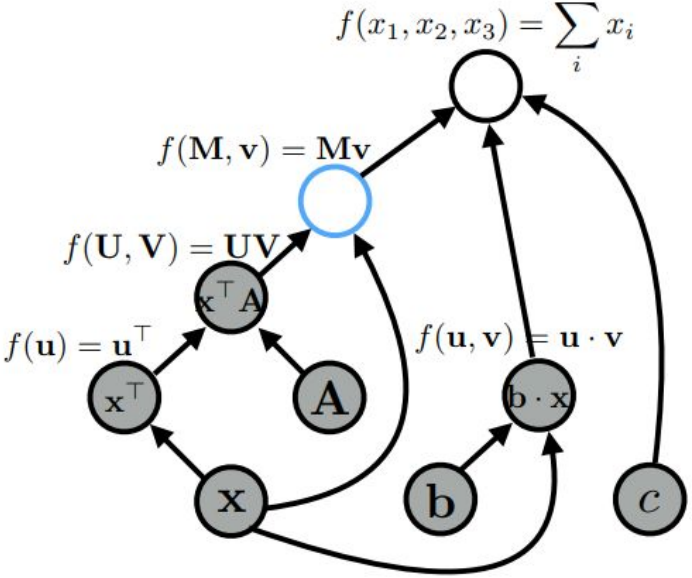
Forward Propagation

graph:



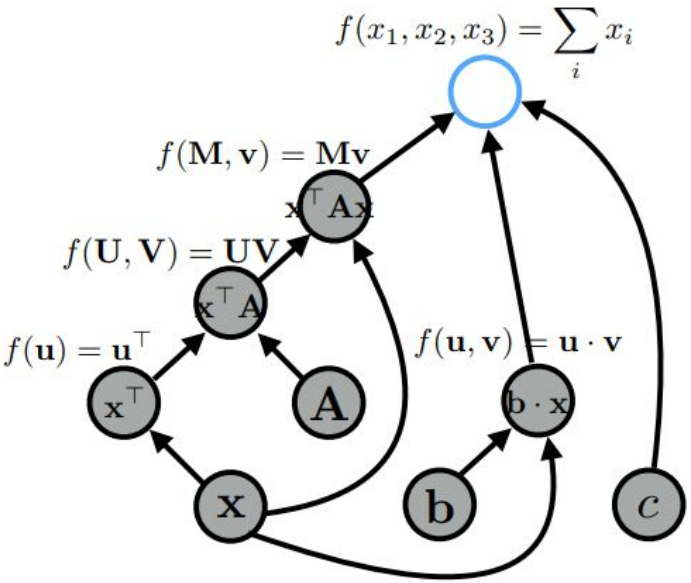
Forward Propagation

graph:



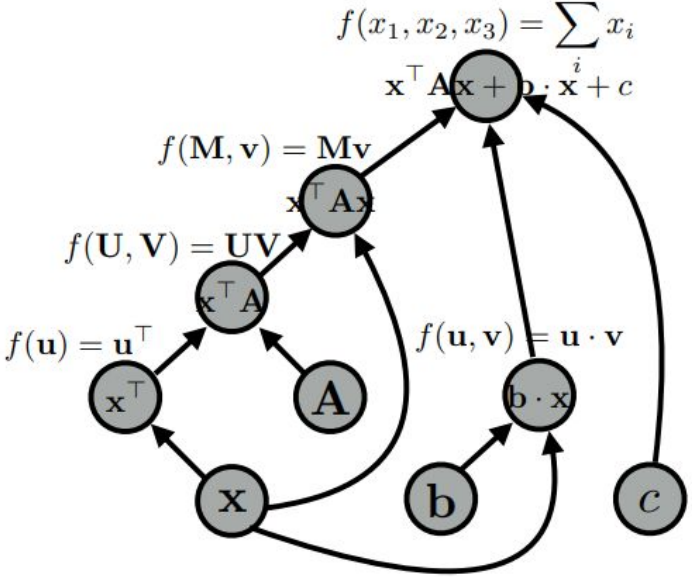
Forward Propagation

graph:



Forward Propagation

graph:



Bài tập: Vẽ đồ thị tính toán

Tính giá trị của biểu thức khi a $= 1$, b $= 2$.

$$(a+b)\,b$$

Hai cách xây dựng đồ thị

Static Computational Graph

- Pha 1. Định nghĩa cấu trúc (có thể bao gồm các điều khiển luồng thô sơ như vòng lặp và điều kiện).
- Pha 2. Cho dữ liệu chạy qua để huấn luyện model và đưa ra dự đoán.

Dynamic Computational Graph

Đồ thị được định nghĩa ngầm (chẳng hạn bằng overload toán tử) khi thực hiện tính toán thuận chiều.

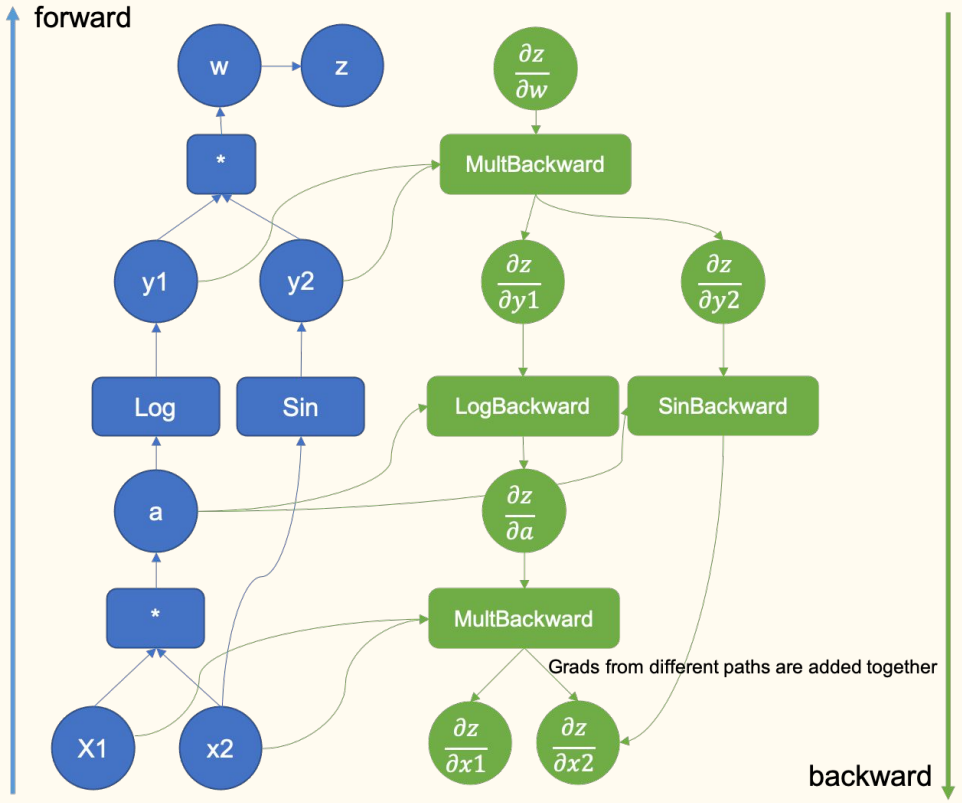
Hai cách xây dựng đồ thị

Static Computational Graph

- PyTorch
- Theano
- TensorFlow

Dynamic Computational Graph

- Chainer
- Thư viện vi phân tự động (hầu hết đều hỗ trợ)
- DyNet



(Đồ thị tính toán được xây dựng như thế nào trong PyTorch)

Cài đặt Computational Graph bằng PyTorch

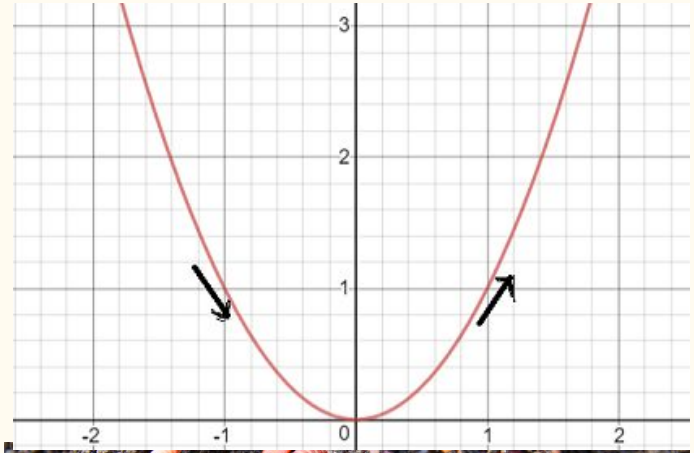
Xem tại computational-graphs-example.ipynb

Sẽ trình bày trong phần
“Ví dụ minh họa”

Ứng dụng

Ứng dụng

- Dùng để tính đạo hàm trong Gradient Descent
- Tăng tốc độ đào tạo các mô hình sâu



Ví dụ minh họa

Ví dụ minh họa

Xem tại computational-graphs-example.ipynb

(Download và mở file
bằng Google Colab)

Cuối cùng

Quiz

Bài tập về nhà

Chương trình demo

Xem tại computational-graphs-demo.ipynb

(Download và mở file
bằng Google Colab)

Goodbye