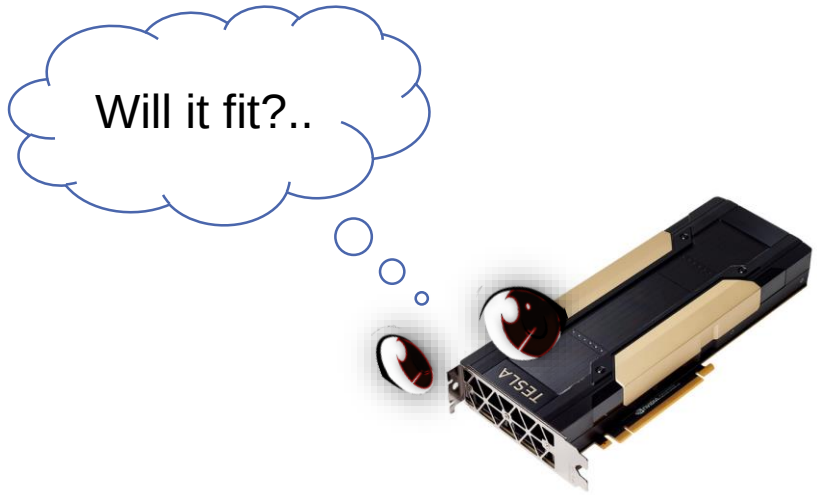
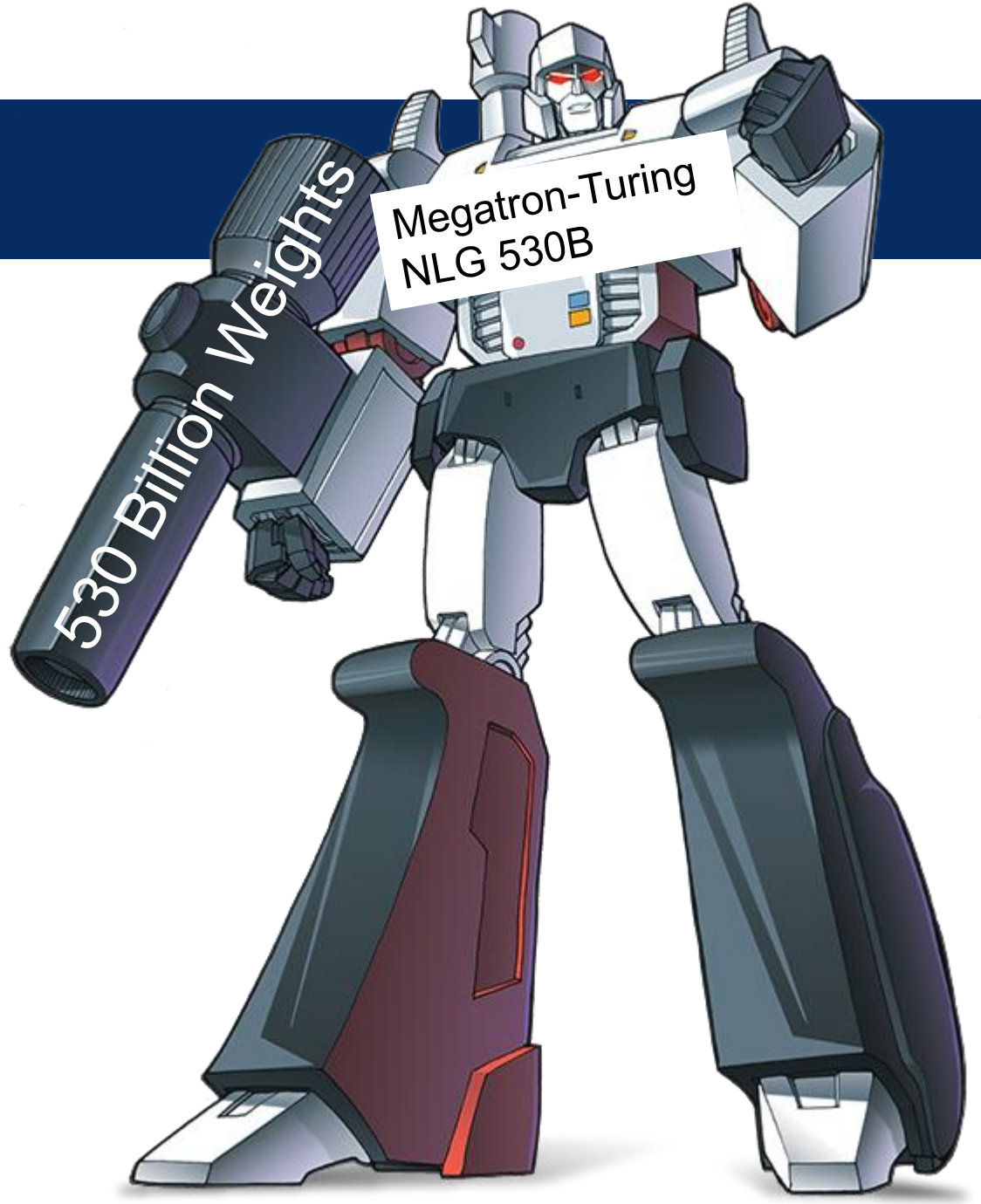
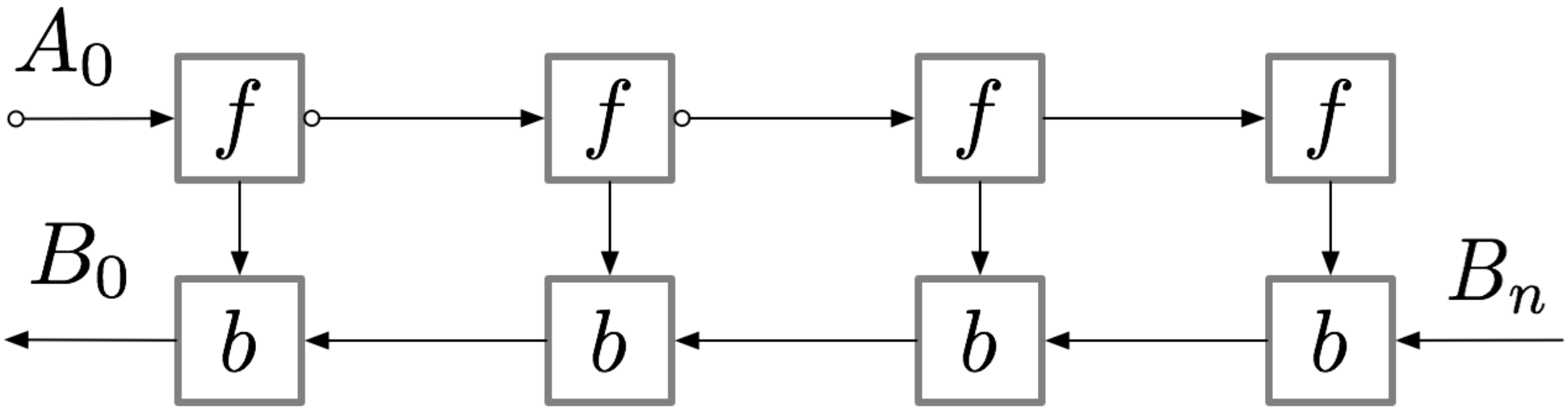
Model Parallelism

А в чём проблема?

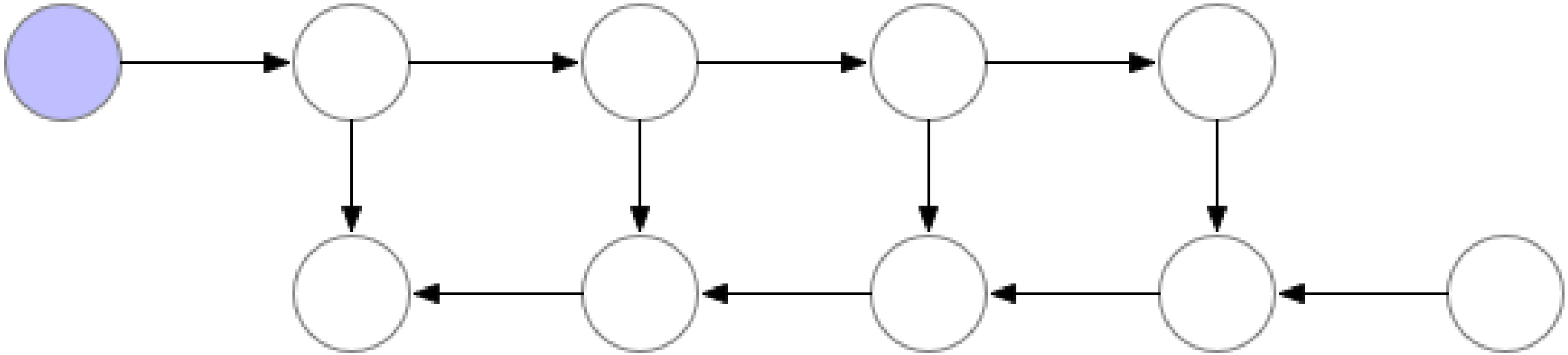




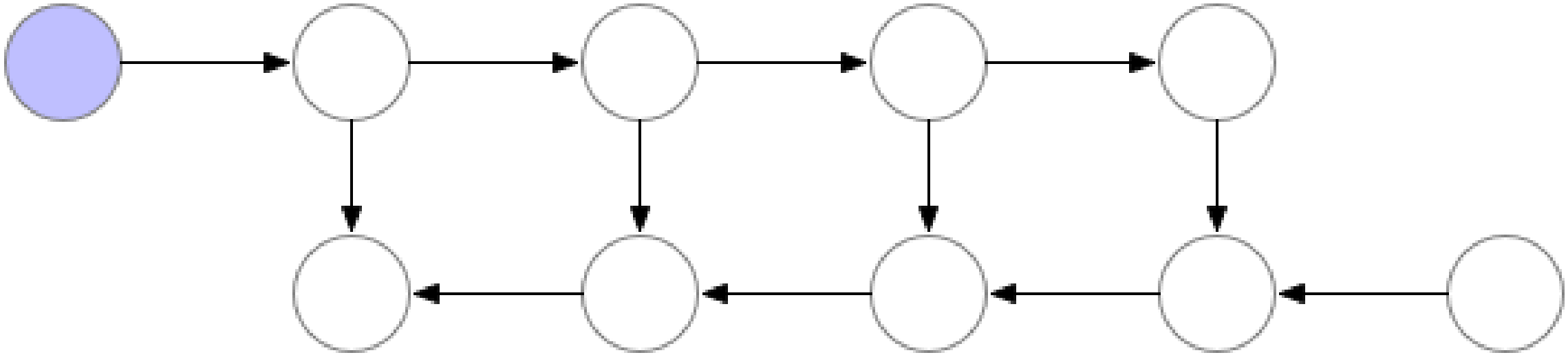
Улучшаем работу с памятью



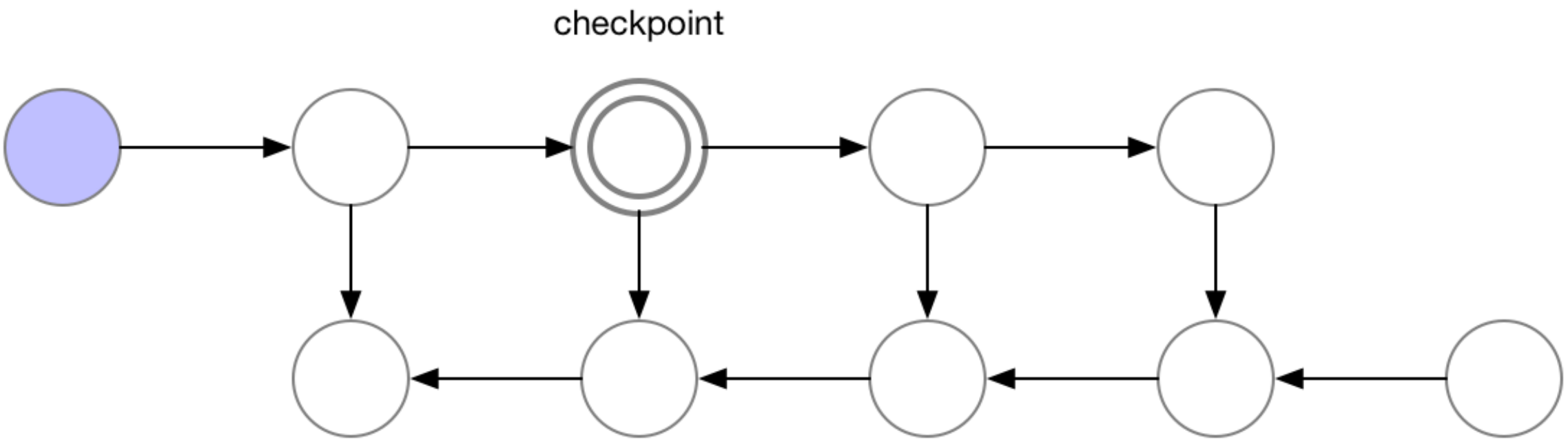
Улучшаем работу с памятью



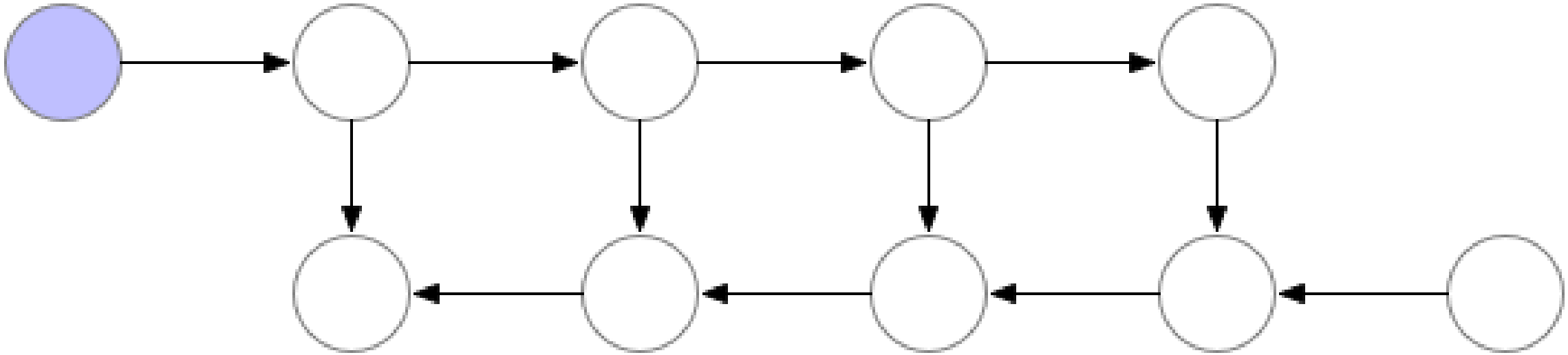
Улучшаем работу с памятью



Улучшаем работу с памятью



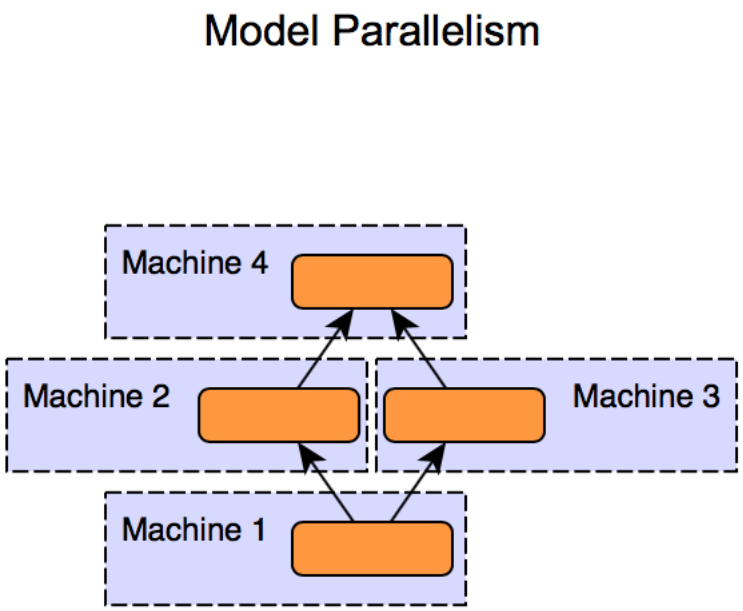
Улучшаем работу с памятью

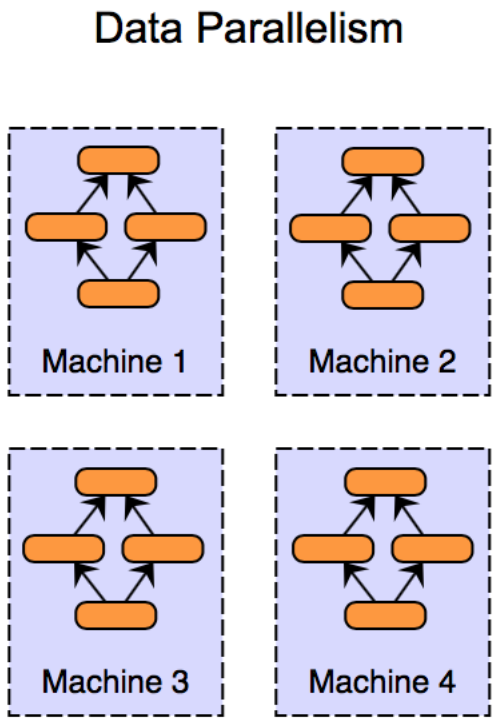


Улучшаем работу с памятью

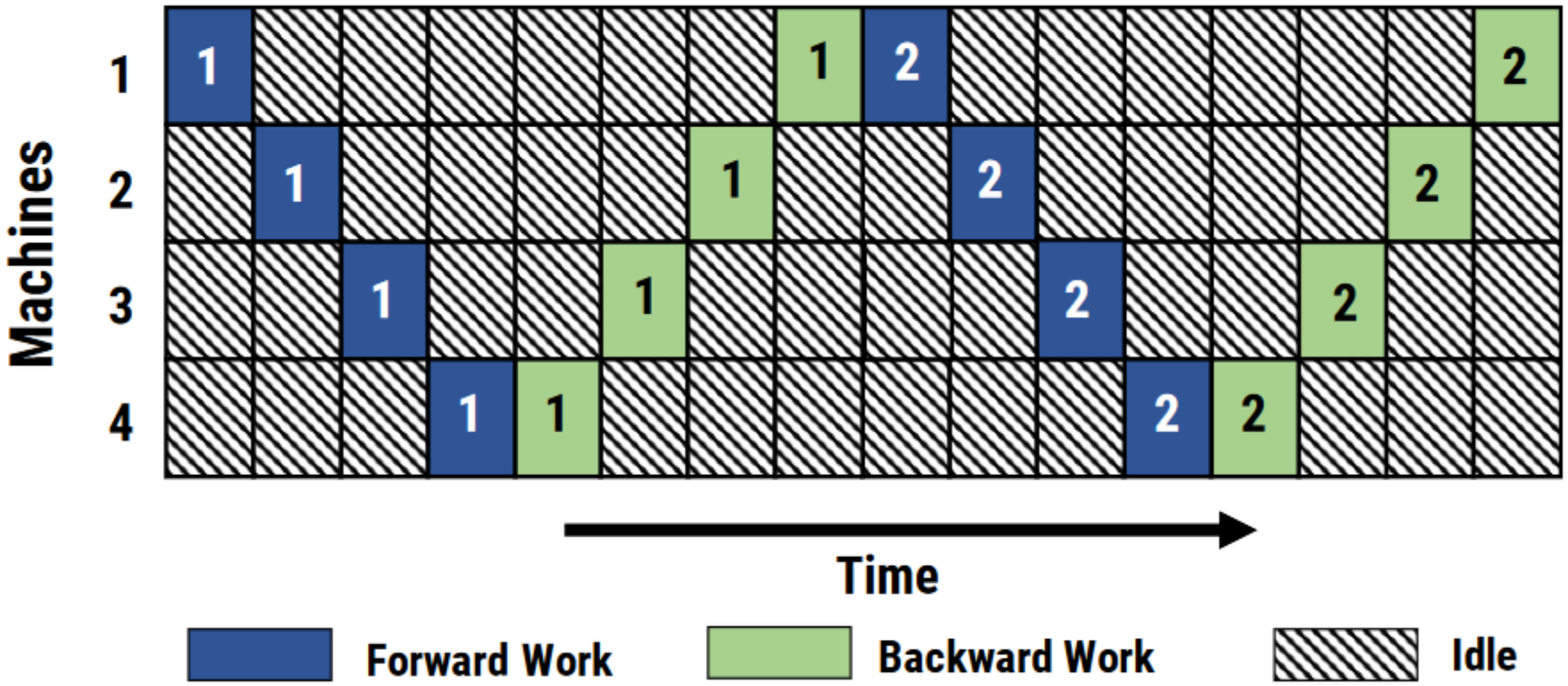
- $M = O\left(\frac{n}{k}\right) + O(k)$
- $k = \sqrt{n}$
- $T = O\left(\frac{n}{\sqrt{n}}\right) + O(\sqrt{n}) = O(2\sqrt{n}) = O(\sqrt{n})$

Всё равно не лезет?

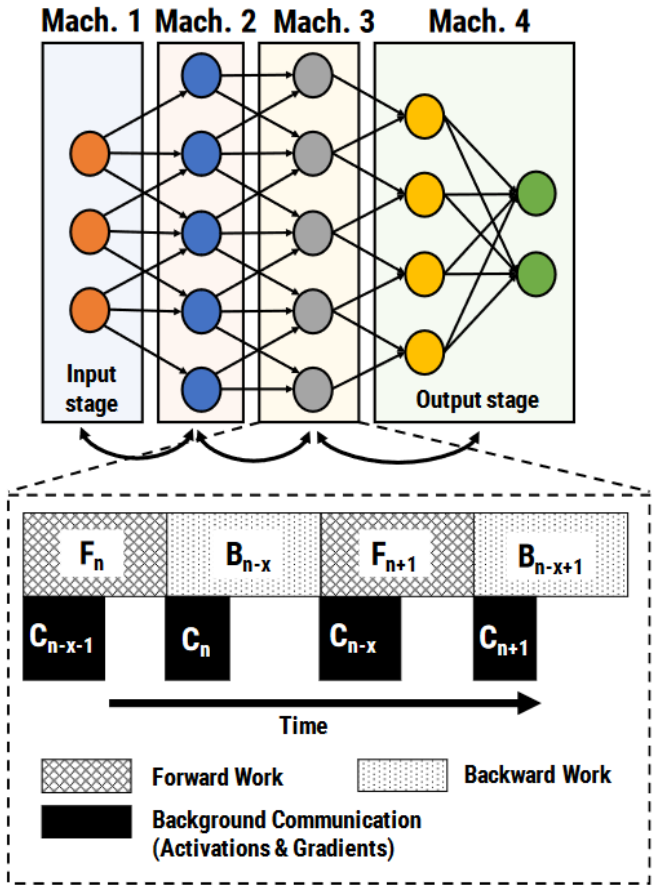




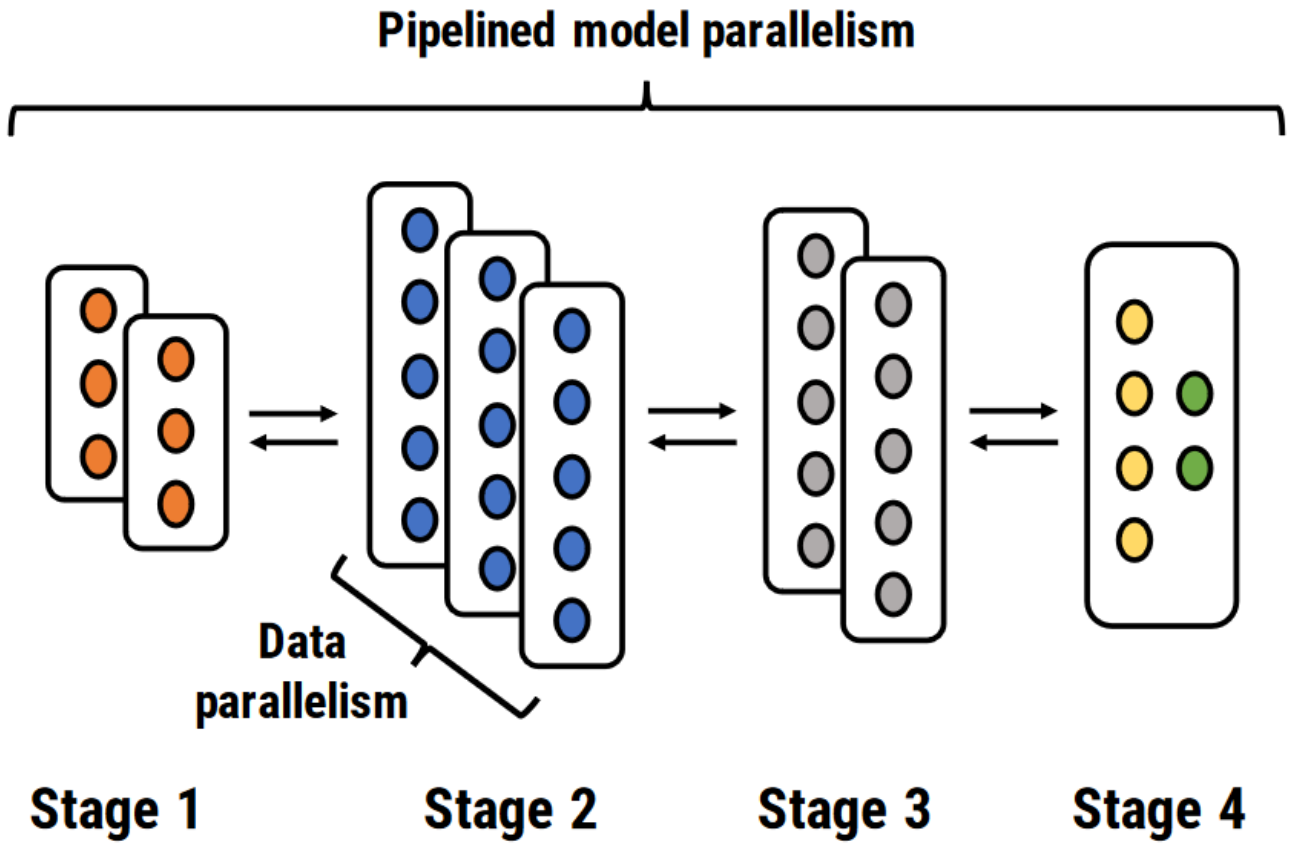
Всё равно не лезет?



Пайплайнинг



Оптимальное разбиение



Профилирование

Обозначения:

- l – номер слоя, m – число машинок
- T_l - время, потраченное на слой l (forward + backward pass)
- a_l - размер выходов слоя l
- w_l - размер весов слоя l
- C_l - время, потраченное на передачу следующему слою (выводится на основе a_l и скорости сети)
- W_l^m - время, необходимое для синхронизации весов при использовании сервера параметров (тоже вычисляется)

Начинаем разбивать

Задача: разбить всю нашу модельку на стадии, выделив каждой сколько-то машинок

Заметка: эта задача аналогична ускорению самой медленной стадии (потому что гоняем параллельно)

Начинаем разбивать

Обозначения:

- $A(j,m)$ - время самой медленной стадии в пайплайне между 1 и j слоями на m машинках
- $T(i \to j,m)$ – время стадии с i по j слой на m машинках
- $T(i \to j,m) = \frac{1}{m}\max\left(\sum_{l=i}^{j} T_l, \sum_{l=i}^{j} W_l^m\right)$

Оптимальное разбиение

1. $A(j,m) = T(1 \to j, m)$

2. $A(j,m) = min_{1 \le i \le j} min_{1 \le m' \le m} max \begin{cases} A(i, m - m') \\ 2 \cdot C_i \\ T(i+1 \to j, m') \end{cases}$

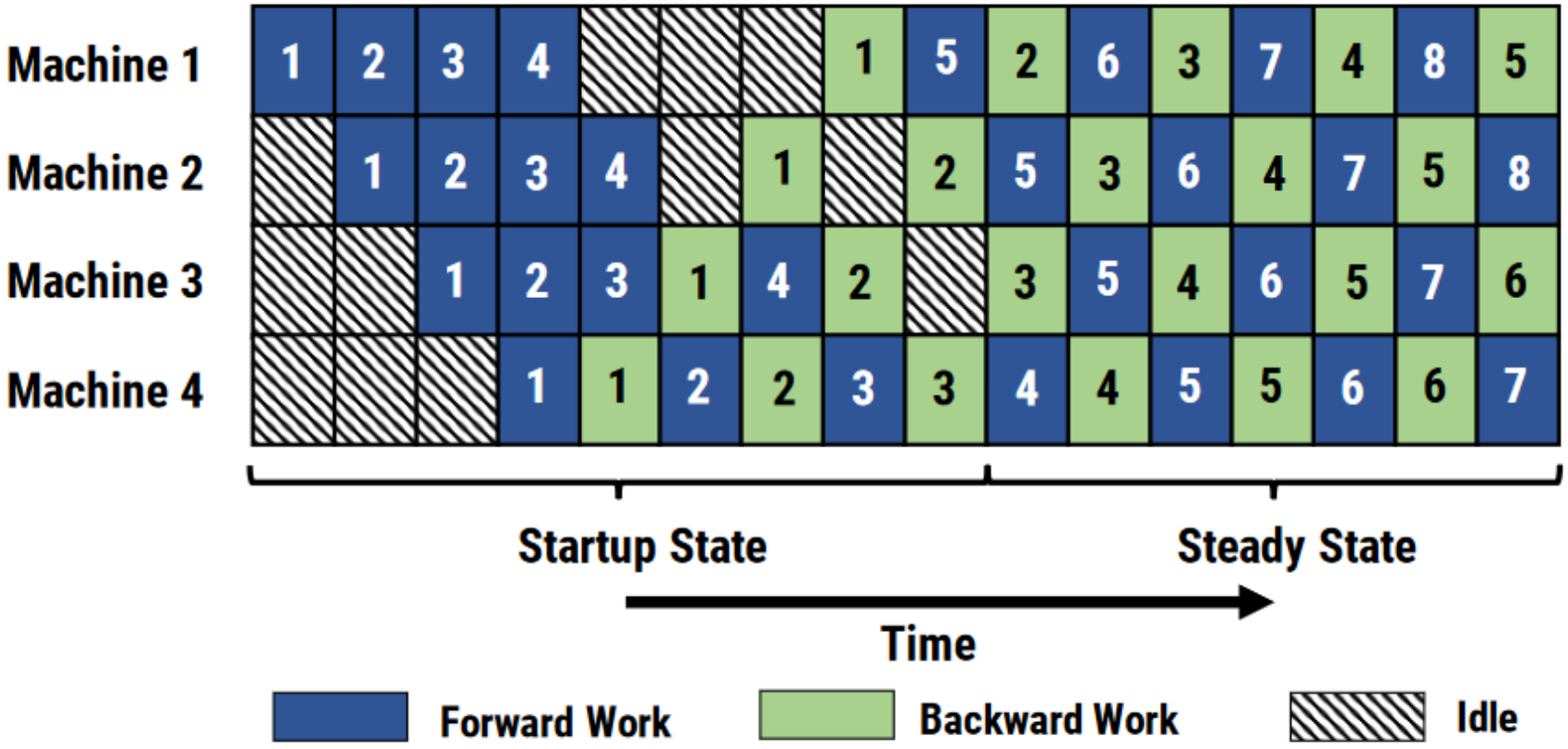
Время работы

- Время, необходимое для решения каждой подзадачи равно $O(NM)$
- Всего у нас NM подзадач, поэтому итоговое время такого анализа будет $O(N^2M^2)$
- N – общее число слоёв
- M – общее число располагаемых нами машин

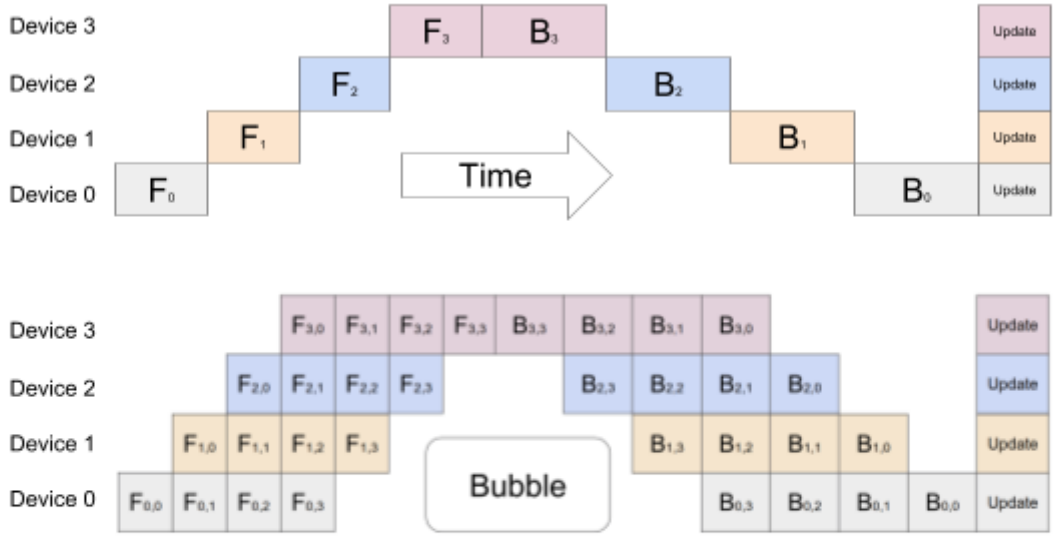
NOAM

- На основе полученного разбиения можно найти оптимальное число батчей, которые надо вкинуть в систему для достижения стабильного состояния:
  ceil(число машин всего / число машин в 1 стадии)
- NOAM = NUM_OPT_ACTIVE_MINIBATCHES

Распределение работы



Чем плох пайплайнинг?



Top: The naive model parallelism strategy leads to severe underutilization due to the sequential nature of the network. Only one accelerator is active at a time. Bottom: GPipe divides the input mini-batch into smaller micro-batches, enabling different accelerators to work on separate micro-batches at the same time.

Transformers, roll out!

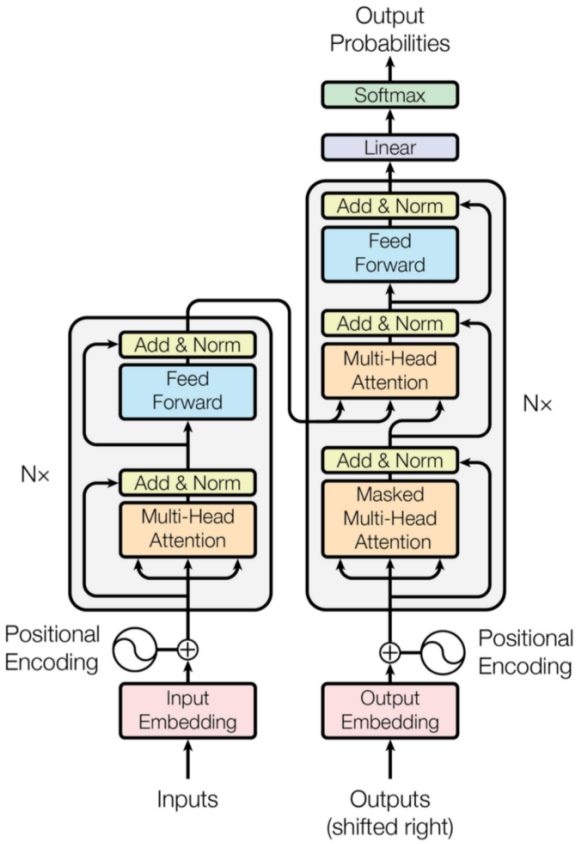
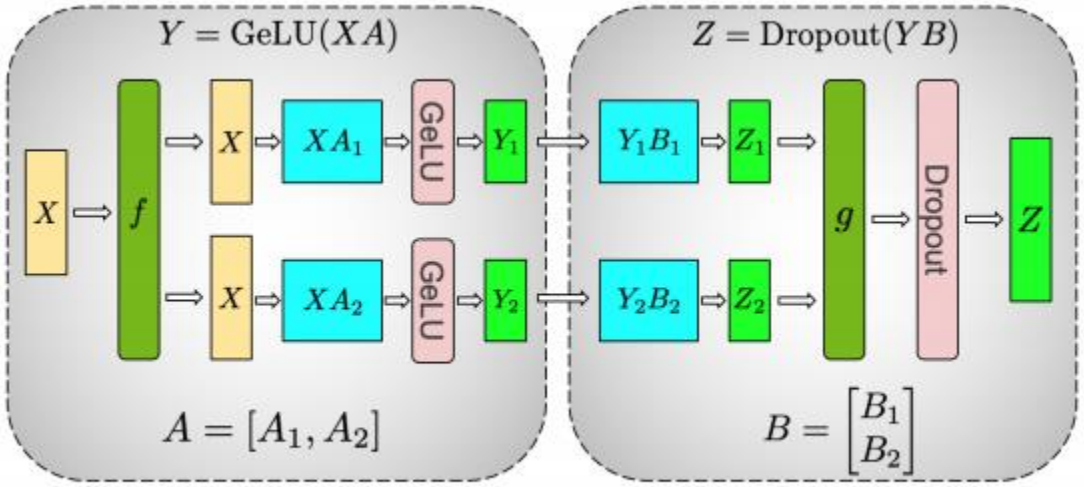


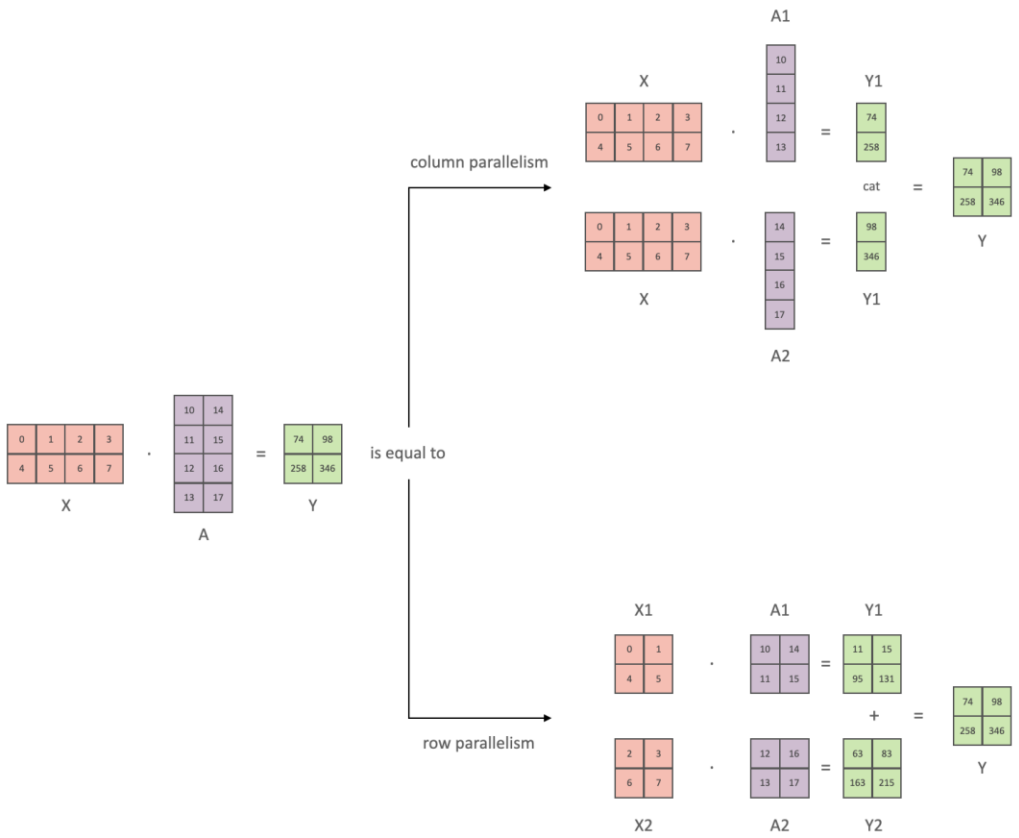
Figure 1: The Transformer - model architecture.

Параллелим MLP



(a) MLP

Параллелим MLP



Параллелим MLP

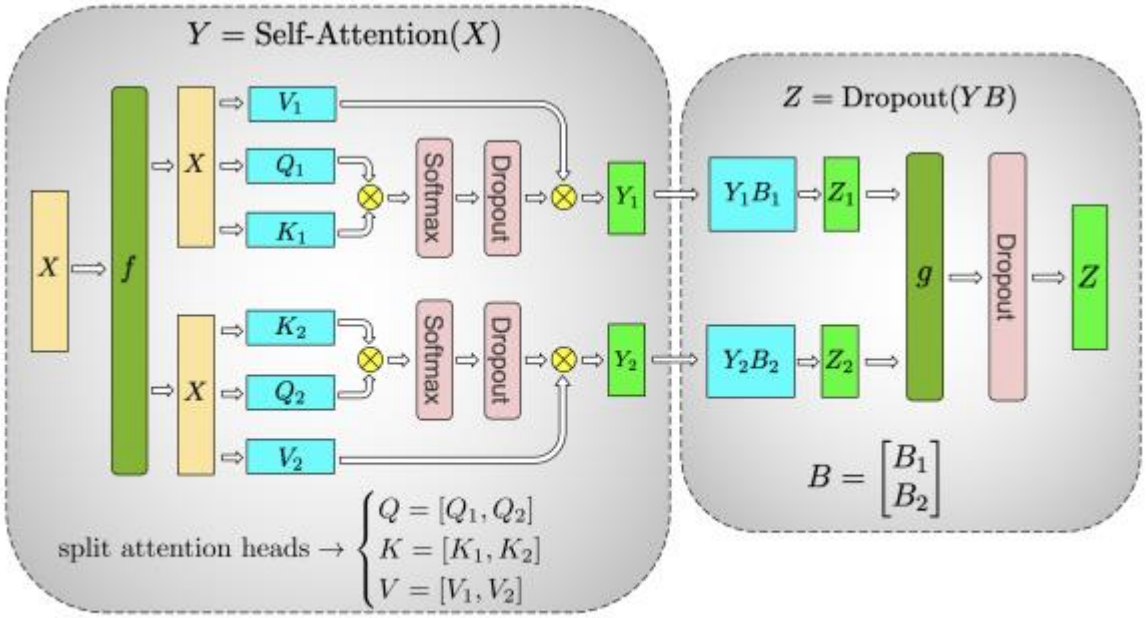
$$X = [X_1, X_2], A = \begin{bmatrix} A_1 \\ A_2 \end{bmatrix}$$

$$GeLU(X_1A_1 + X_2A_2) \neq GeLU(X_1A_1) + GeLU(X_2A_2)$$

$$X = X, A = [A_1A_2]$$

$$[Y_1Y_2] = [GeLU(XA_1)\ GeLU(XA_2)]$$

Параллелим Self-Attention



(b) Self-Attention

Параллелим Эмбеддинги

- $E_{H \times v}$, где H – скрытая размерность, v – размер словаря
- $[Y_1 Y_2] = [XE_1, XE_2]$
- $Y_{b \times s \times v}$, где b – размер батча, s – размер последовательности

Источники

- Training Deep Nets with Sublinear Memory Cost: https://arxiv.org/pdf/1604.06174.pdf
- Репа на гитхабе про чекпоинтинг: https://github.com/cybertronai/gradient-checkpointing
- PipeDream: Fast and Efficient Pipeline Parallel DNN Training: https://arxiv.org/pdf/1806.03377.pdf
- Статья про Model Parallelism: https://huggingface.co/docs/transformers/parallelism
- Megatron-LM: Training Multi-Billion Parameter Language Models Using Model Parallelism: https://arxiv.org/pdf/1909.08053.pdf
- Статья про трансформеры: https://medium.com/inside-machine-learning/what-is-a-transformer-d07dd1fbec04
- FlexFlow: https://arxiv.org/pdf/1807.05358.pdf
- Mesh-TensorFlow: https://arxiv.org/pdf/1811.02084.pdf