Performance Optimization of LSTM Training on GPU

Haibin Lin, Yiming Wu
Parallel Computer Architecture and Programming, 15-418

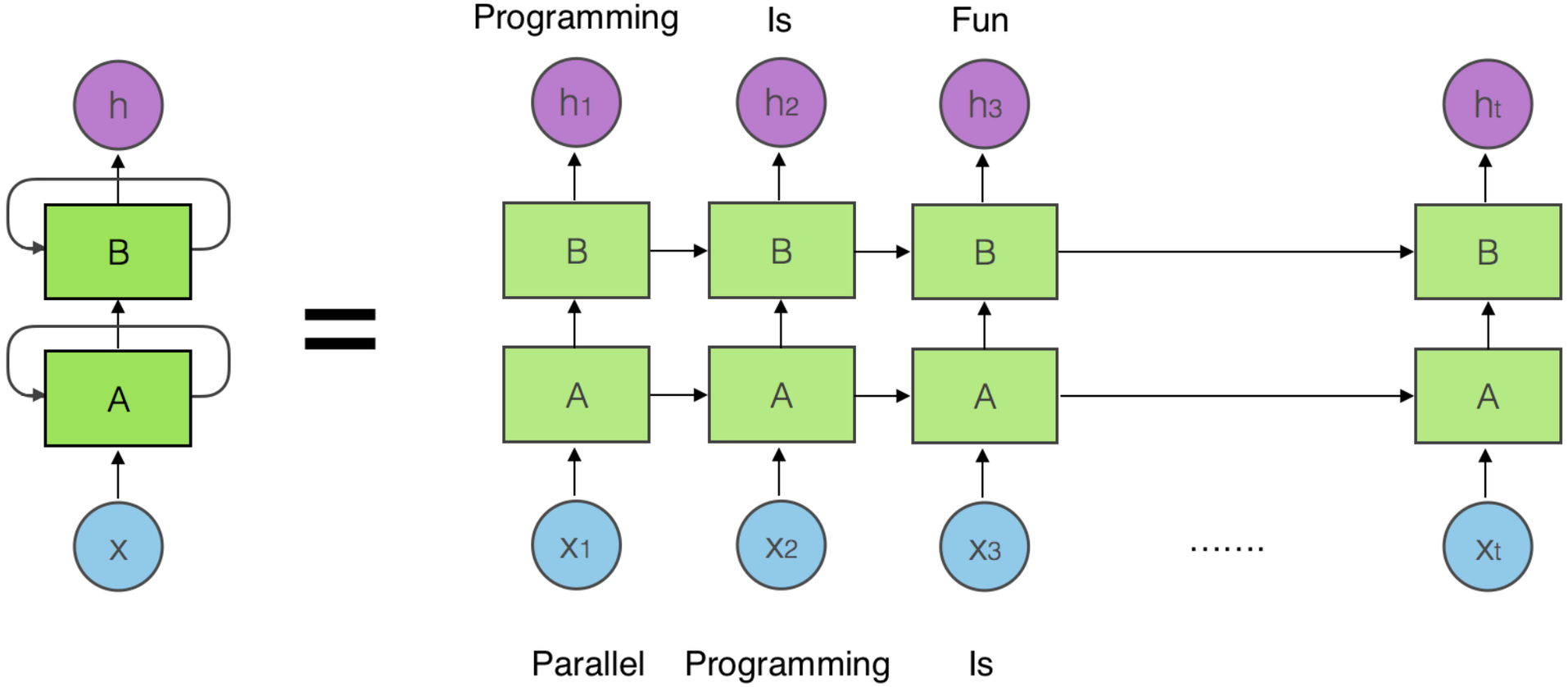
MSU Computer Science Department

Speed Up LSTM Training on GPU

What is LSTM

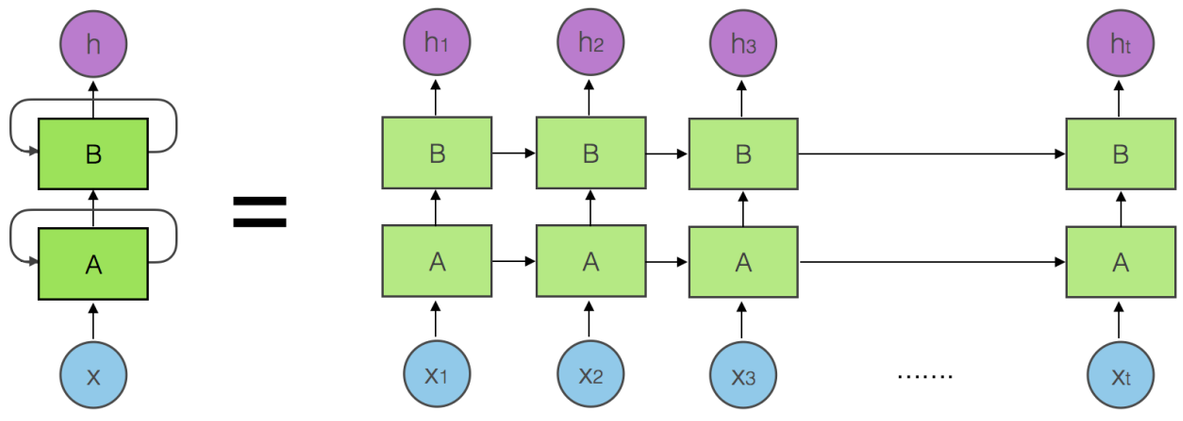
Long Short Term Memory networks (LSTMs) – a special recurrent neural network capable of learning long-term dependencies.

LSTM can learn long-term dependencies by using previous output as the next input.
Based on previous words, it can predict next word to be 'Fun'

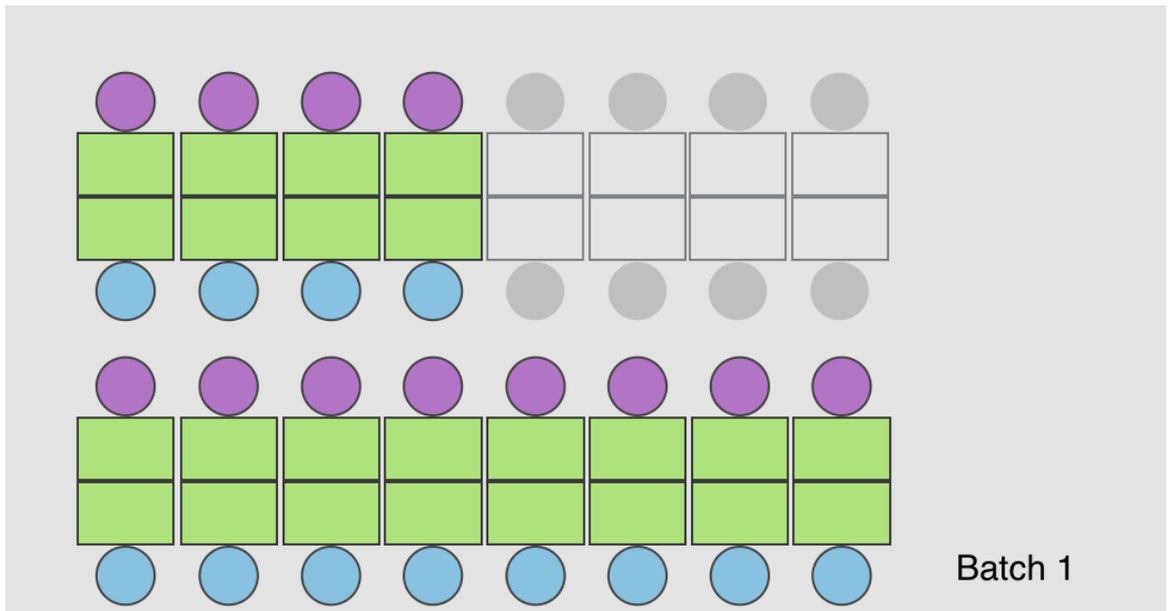


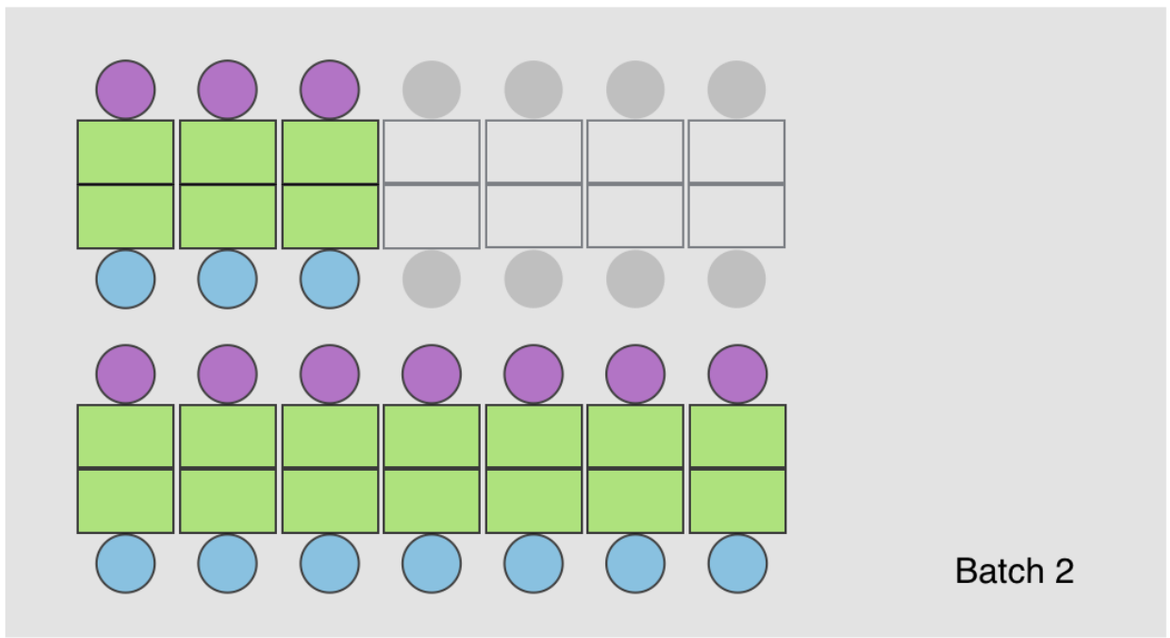
LSTM training procedure

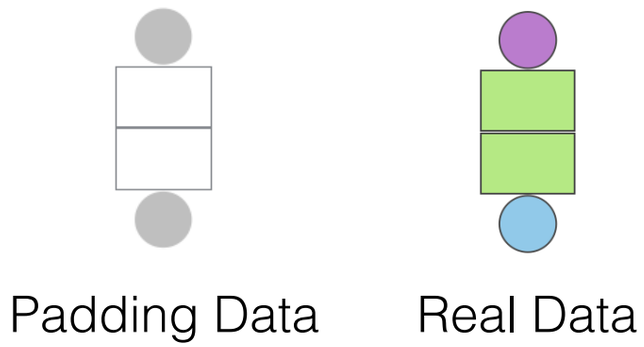
```
While (loss not converge):
    load_batch();          ← Load a batch of sentences
    unroll_LSTM();         ← Unroll to the longest sentence

    for each sentence in batch:
        for each unrolled LSTM unit:   ← Need to consider unit dependency
            forward_evaluate(sentence);

        grad = compute_grad(loss_func, shared_params, sentence);
    update(shared_params, grad);   ← Update parameters, shared per layer
```



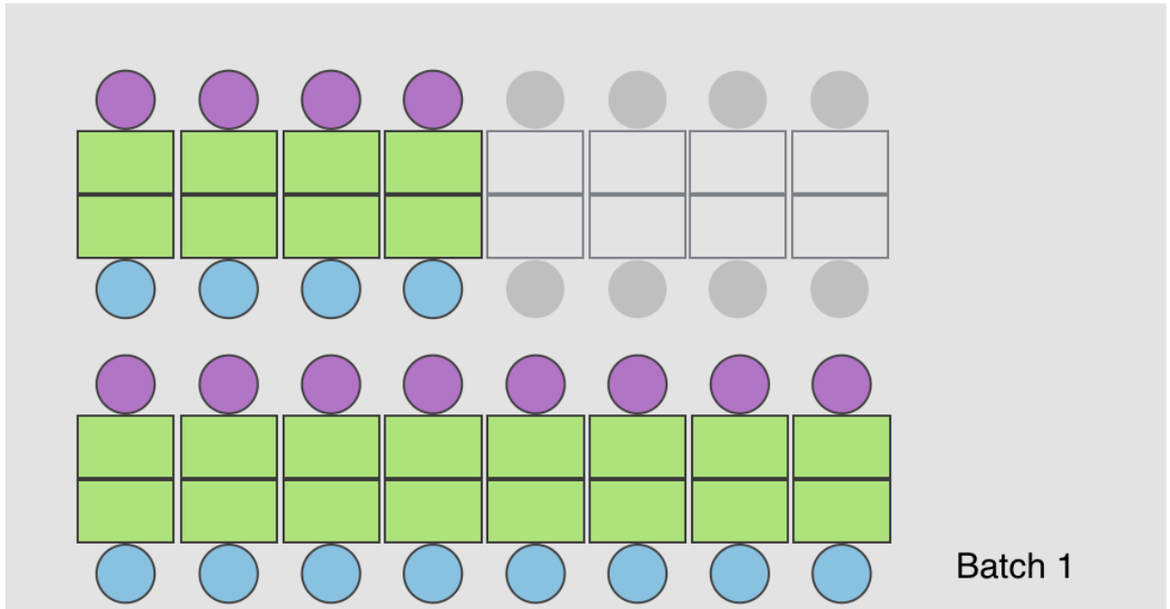
Padding data brings in unnecessary work

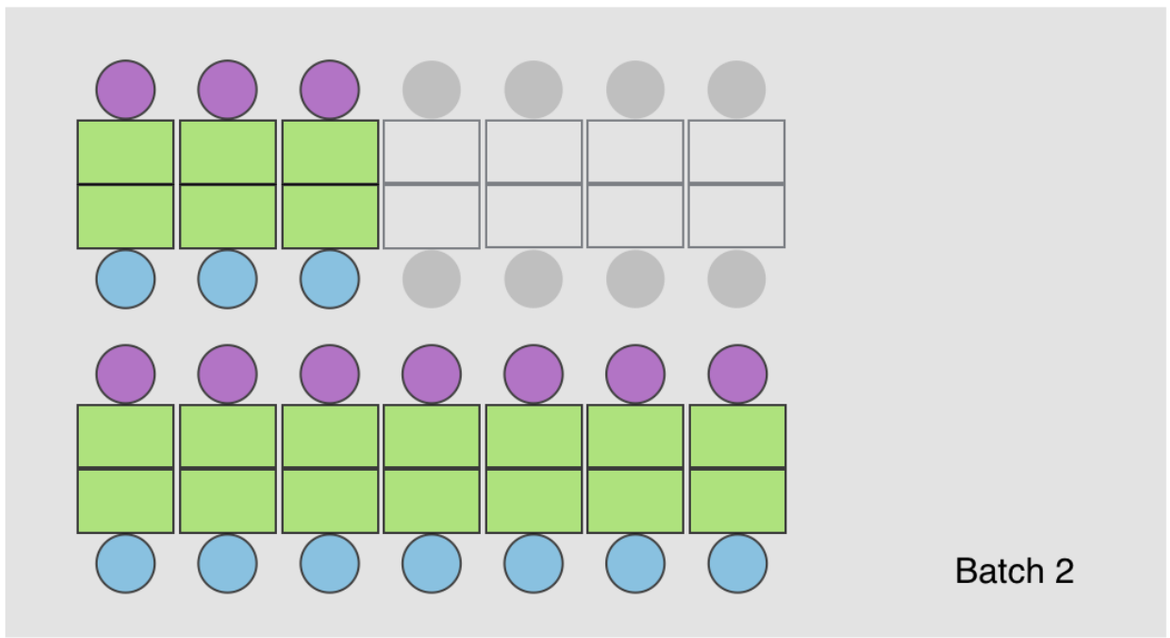






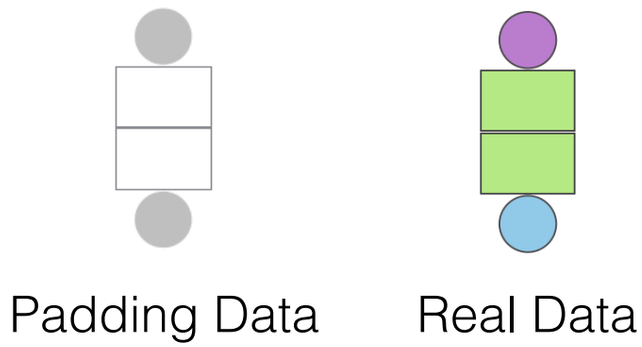
Padding data brings in unnecessary work



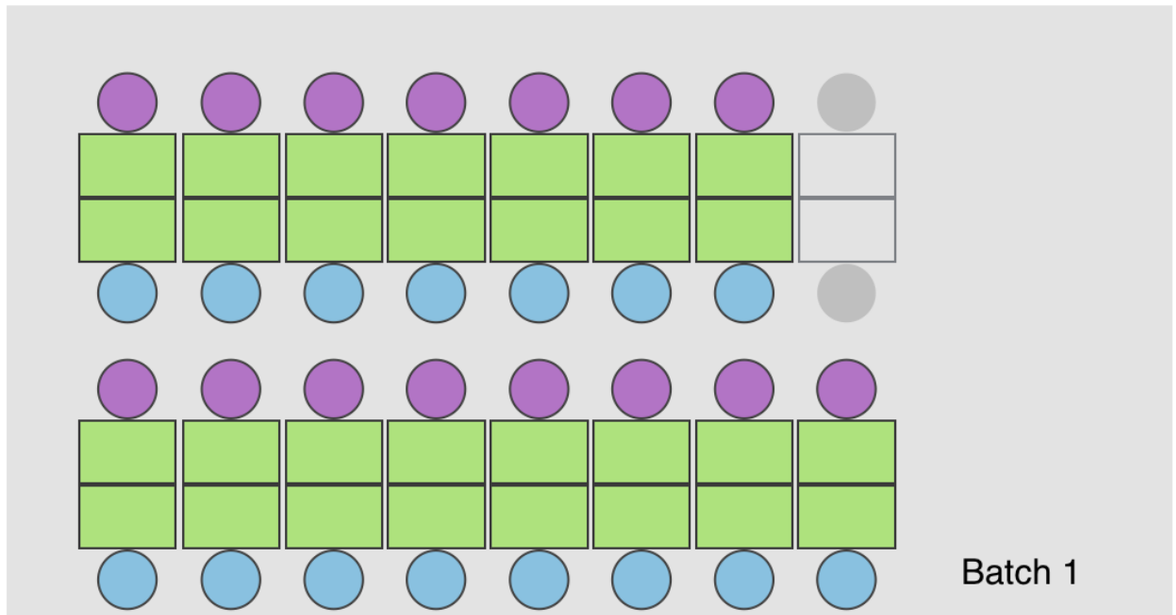


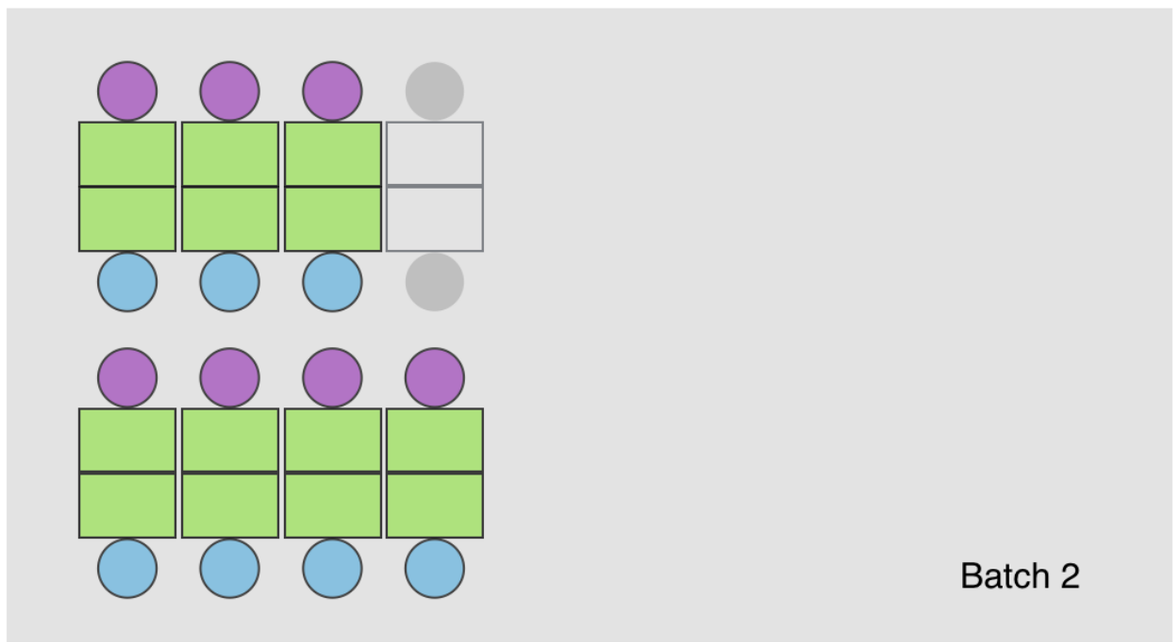
Ideal workload:
4 + 8 + 3 + 7 = 22 words

Padded workload:
8 + 8 + 7 + 7 = 30 words ~ 36% more works



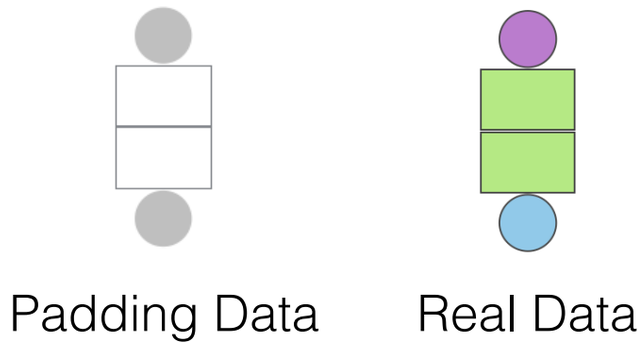
Bucketing: Gathering sequences of similar length



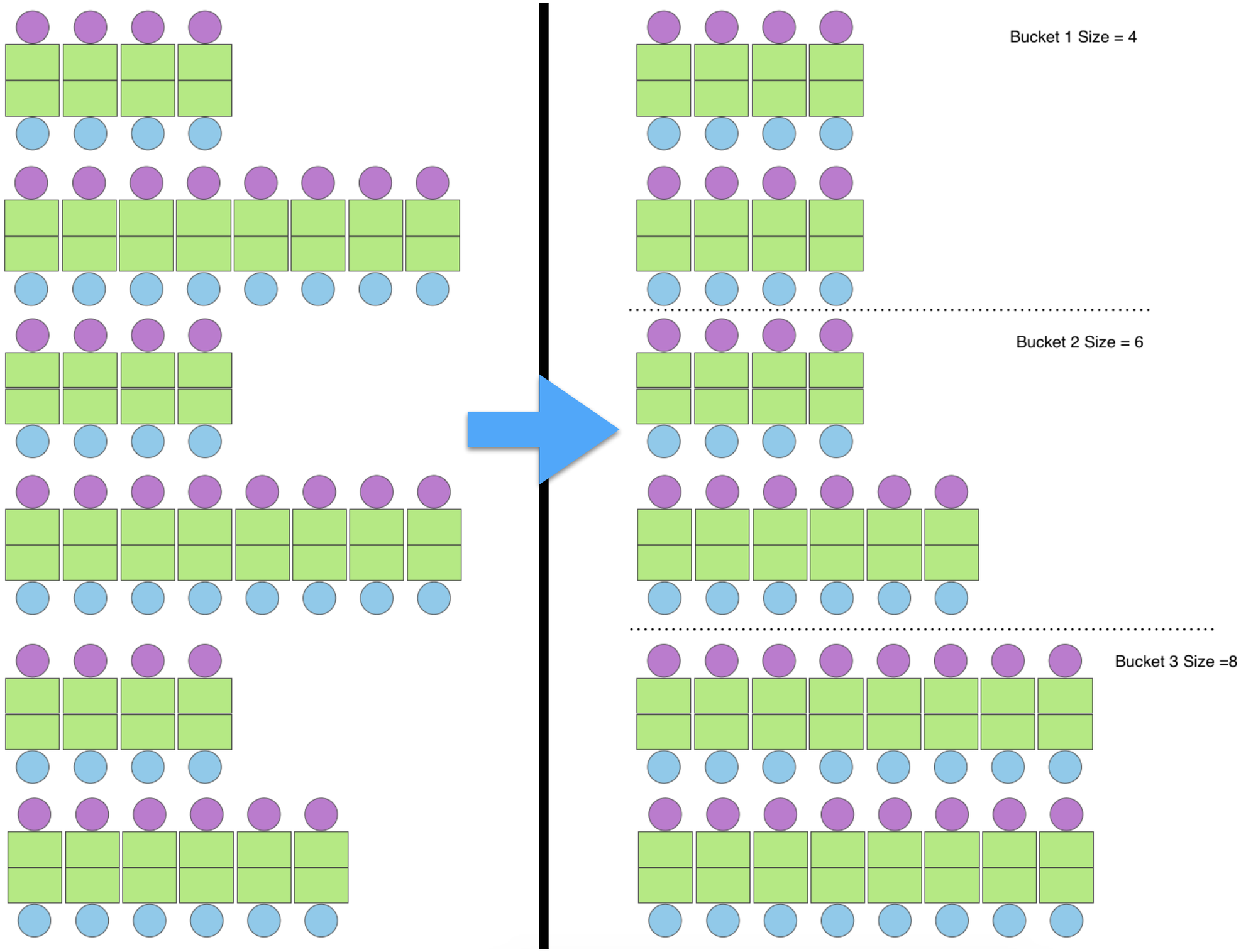


Ideal workload:
4 + 8 + 3 + 7 = 22 words

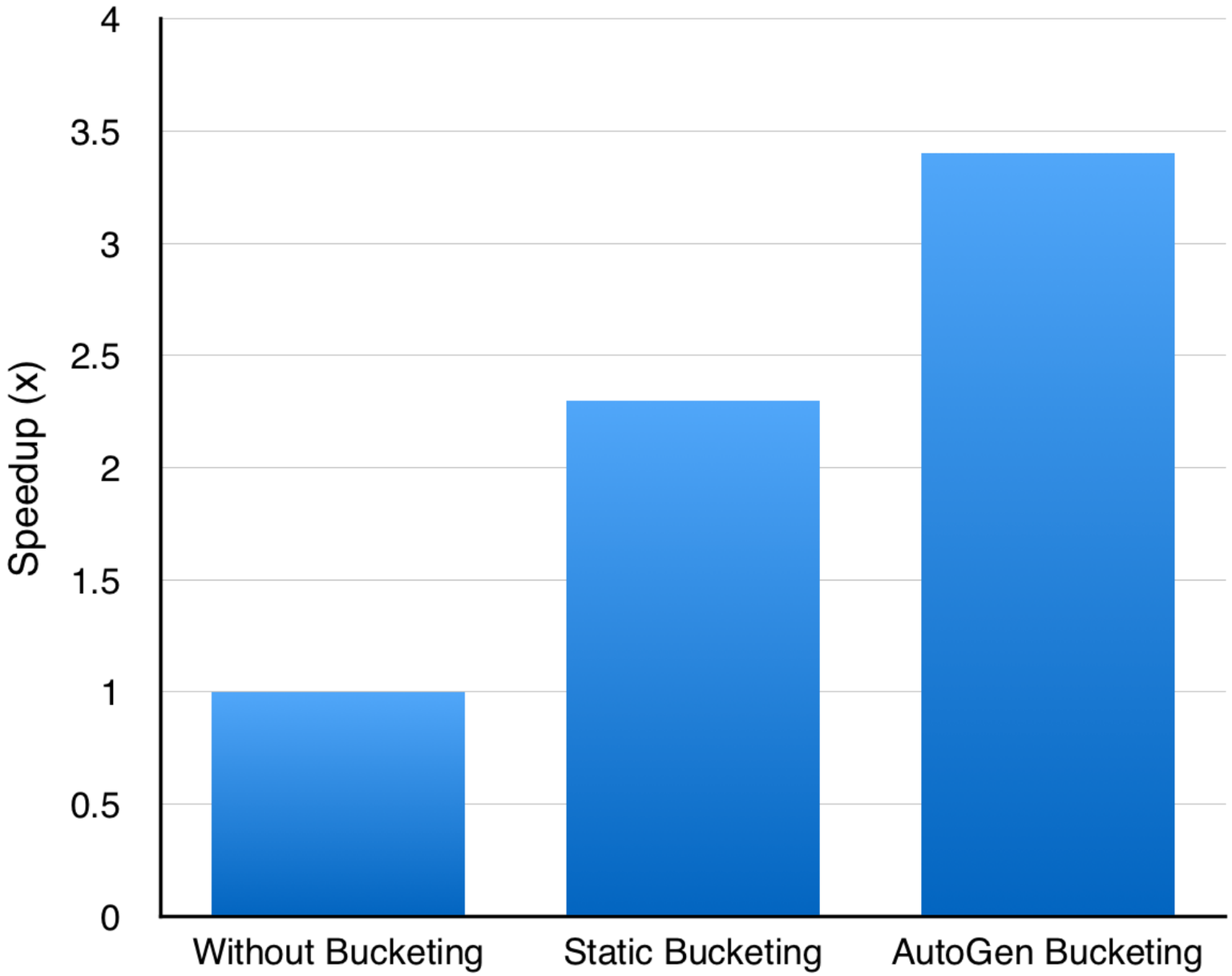
Bucketing workload:
8 + 8 + 4 + 4 = 24 words ~ ideal



Auto-generate buckets greedily to ensure similar lengths based on input



Auto-generating buckets from data gets the best speed up



* We trained 8-layer LSTM on PennTree Bank dataset with one GTX750
* Static bucketing means a user specified fixed bucketing configuration
* **3.3x** faster with autogen bucketing compared to no bucketing

Speed Up LSTM Training on Multiple GPUs

LSTM training procedure

```
While (loss not converge):
    load_batch();
    unroll_LSTM();

    for each sentence in batch:
        for each unrolled LSTM unit:
            forward_evaluate(sentence);

        grad = compute_grad(loss_func, shared_params, sentence);
        update(shared_params, grad);
```

← Data Parallel

← Model Parallel

Speed Up LSTM Training on Multiple GPUs

“Data Parallel”

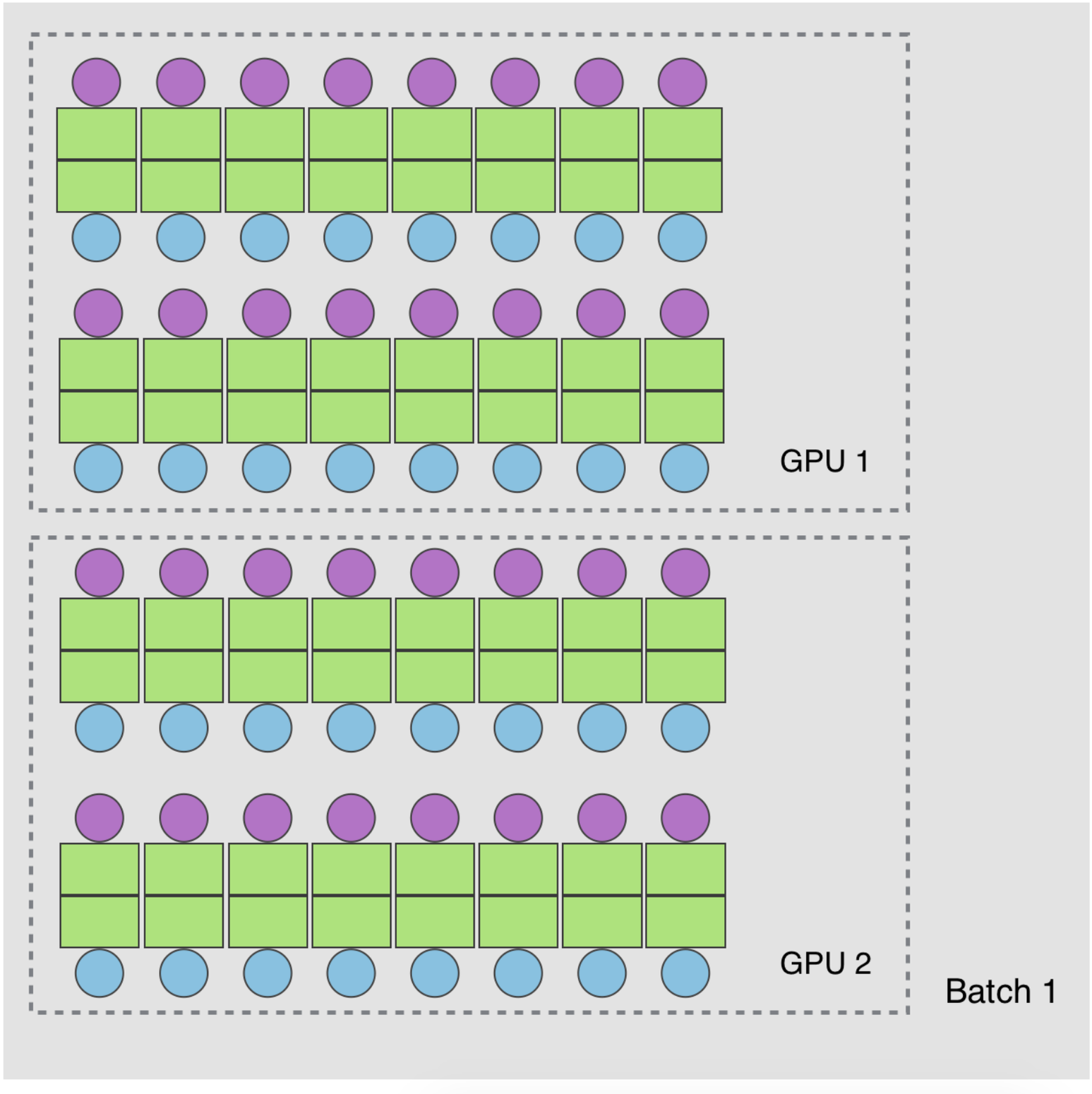
LSTM training procedure with DP

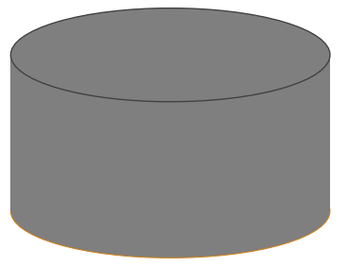
```
While (loss not converge):
    load_batch();
    unroll_LSTM();

    for each sentence in batch:
        for each unrolled LSTM unit:
            forward_evaluate(sentence);

        grad = compute_grad(loss_func, shared_params, sentence);
    update(shared_params, grad);
```

← Partition data over GPUs

← Synchronize with Centralized Parameter Storage

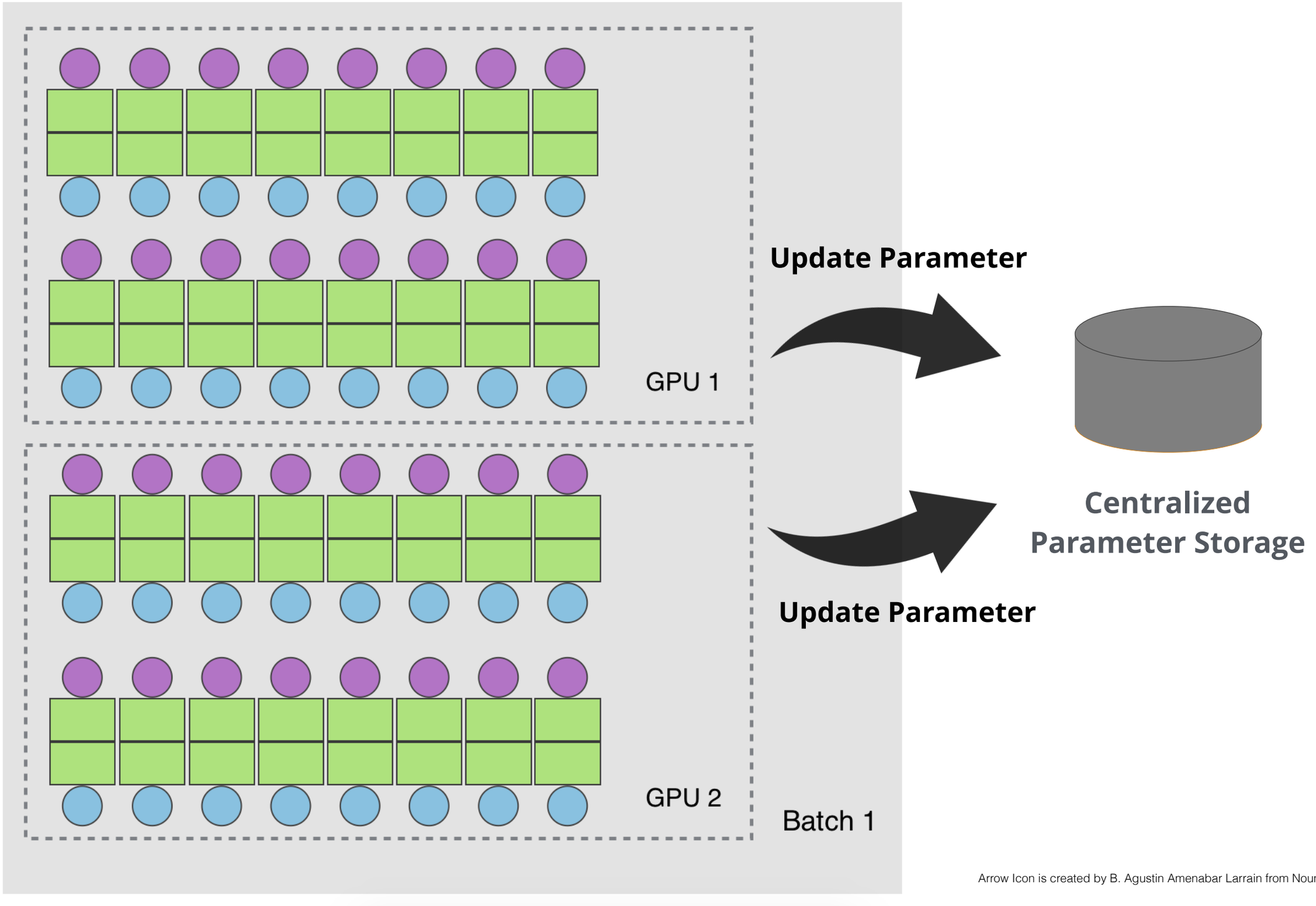
DP 101: Data is the primitive





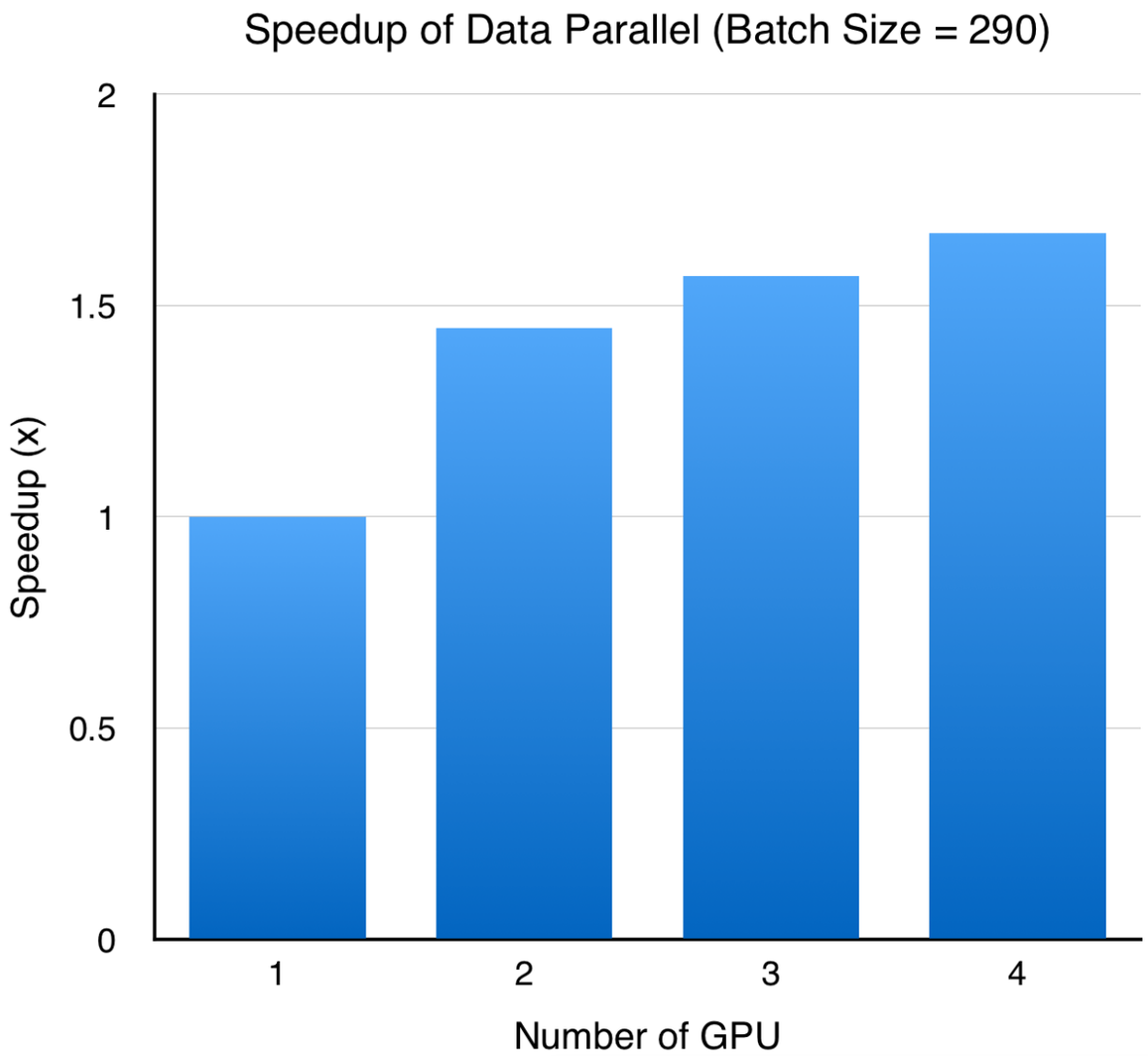
Centralized
Parameter Storage

DP 101: Data is the primitive



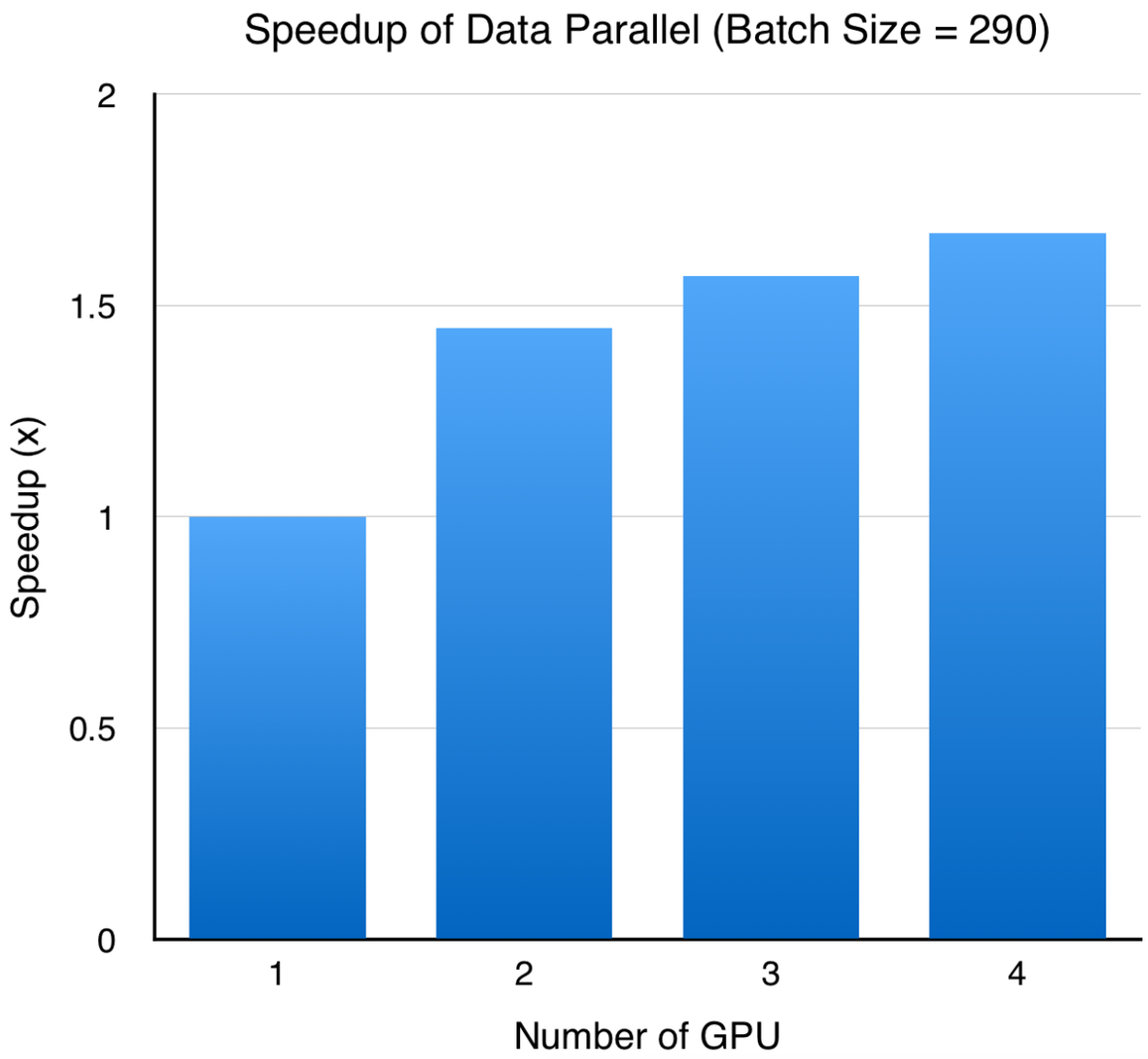
Arrow Icon is created by B. Agustin Amenabar Larrain from Noun Project

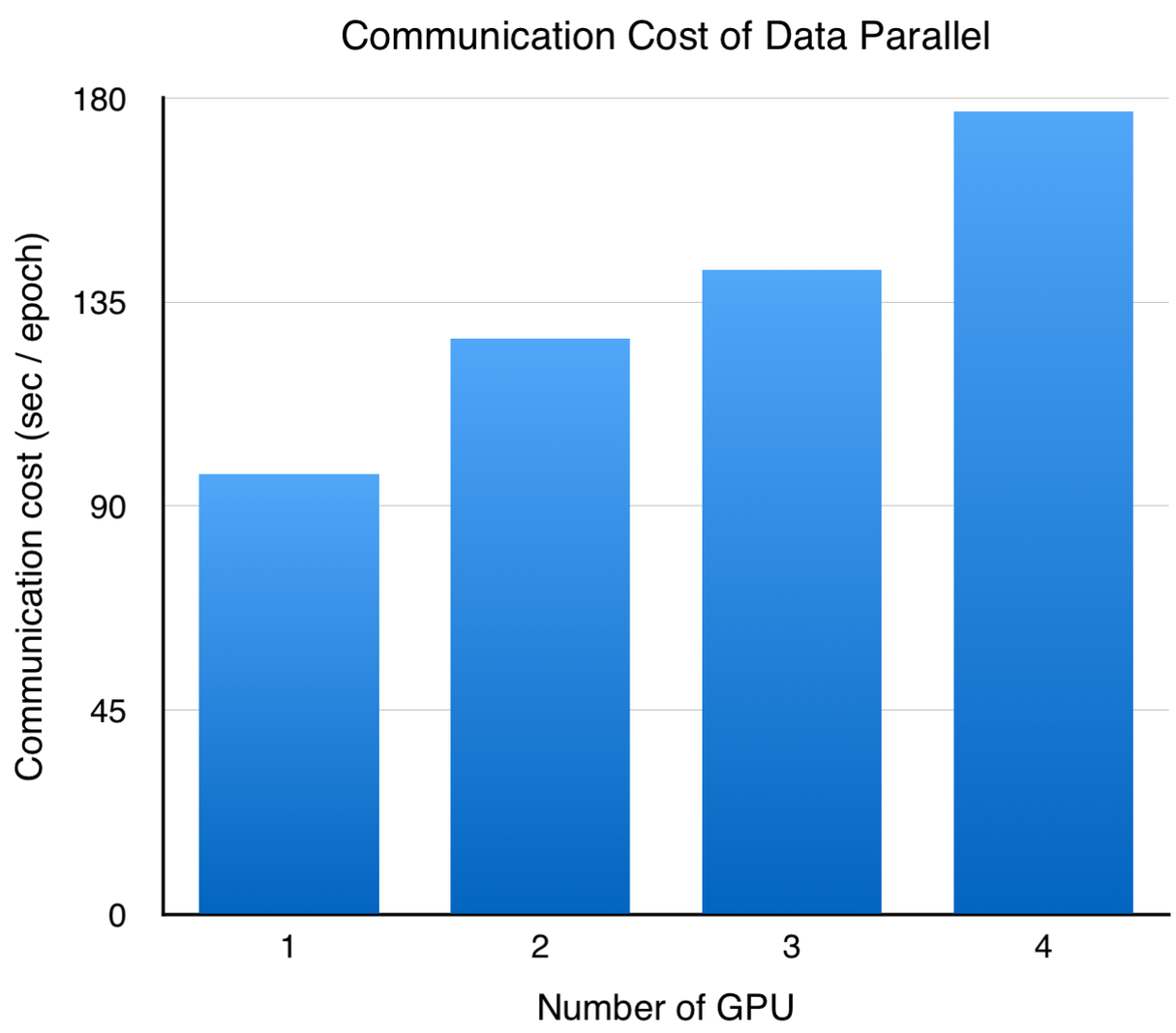
DP gains some speedup on multi-GPU…



* Trained 8-layer LSTM with PennTree Bank dataset, with 1-4 GTX750 on single machine

DP gains some speedup on multi-GPU…





* Experiment with no computation and only communication

Speed Up LSTM Training on Multiple GPUs

“Model Parallel”

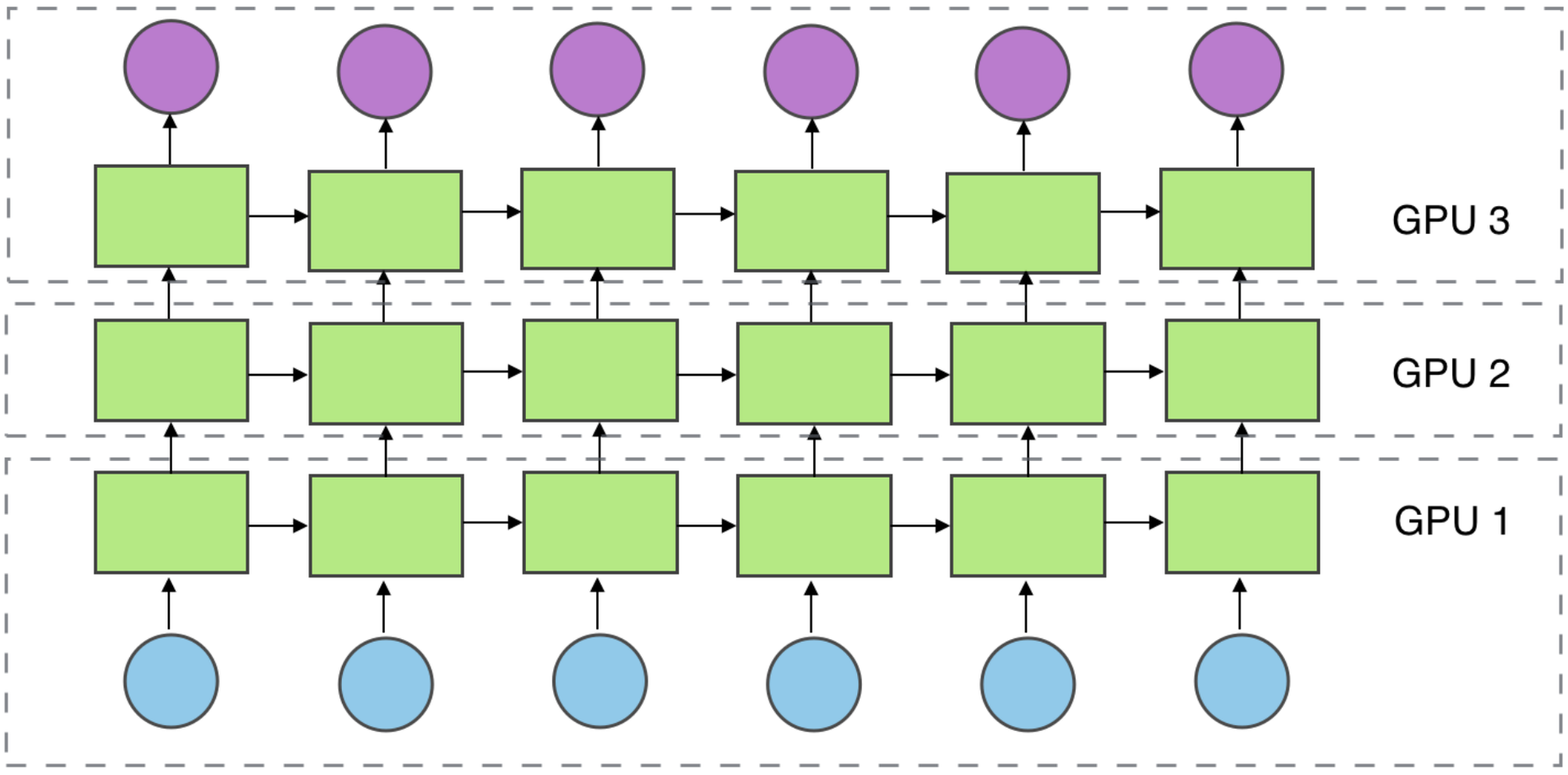
LSTM training procedure with MP

```
While (loss not converge):
    load_batch();
    unroll_LSTM();

    for each sentence in batch:
        for each unrolled LSTM unit:
            forward_evaluate(sentence);

        grad = compute_grad(loss_func, shared_params, sentence);
        update(shared_params, grad);
```

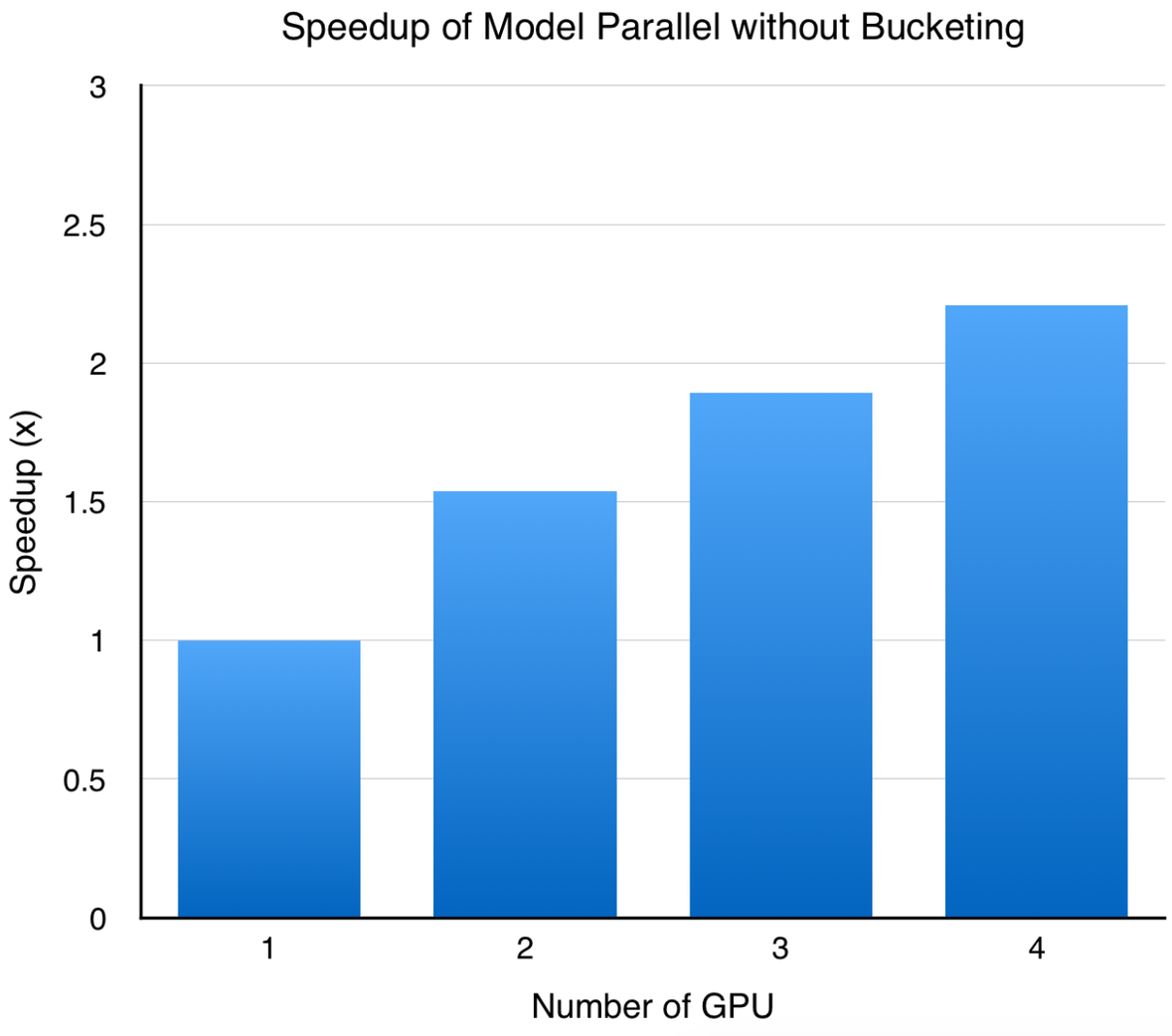
← Partition the LSTM layers over GPUs

← No need for Centralized Parameter Storage

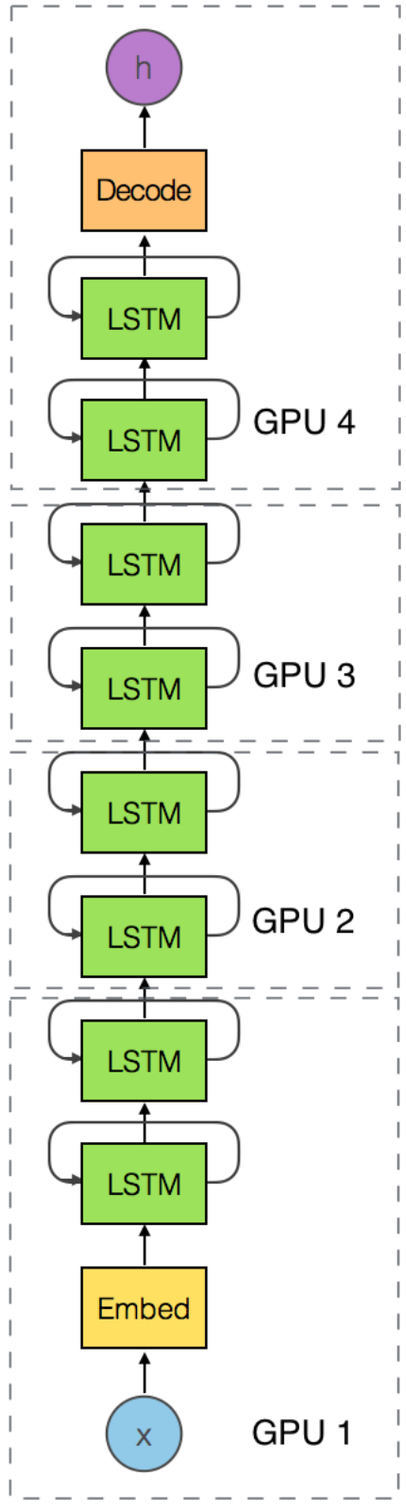
MP 101: Using multi GPU as a pipeline



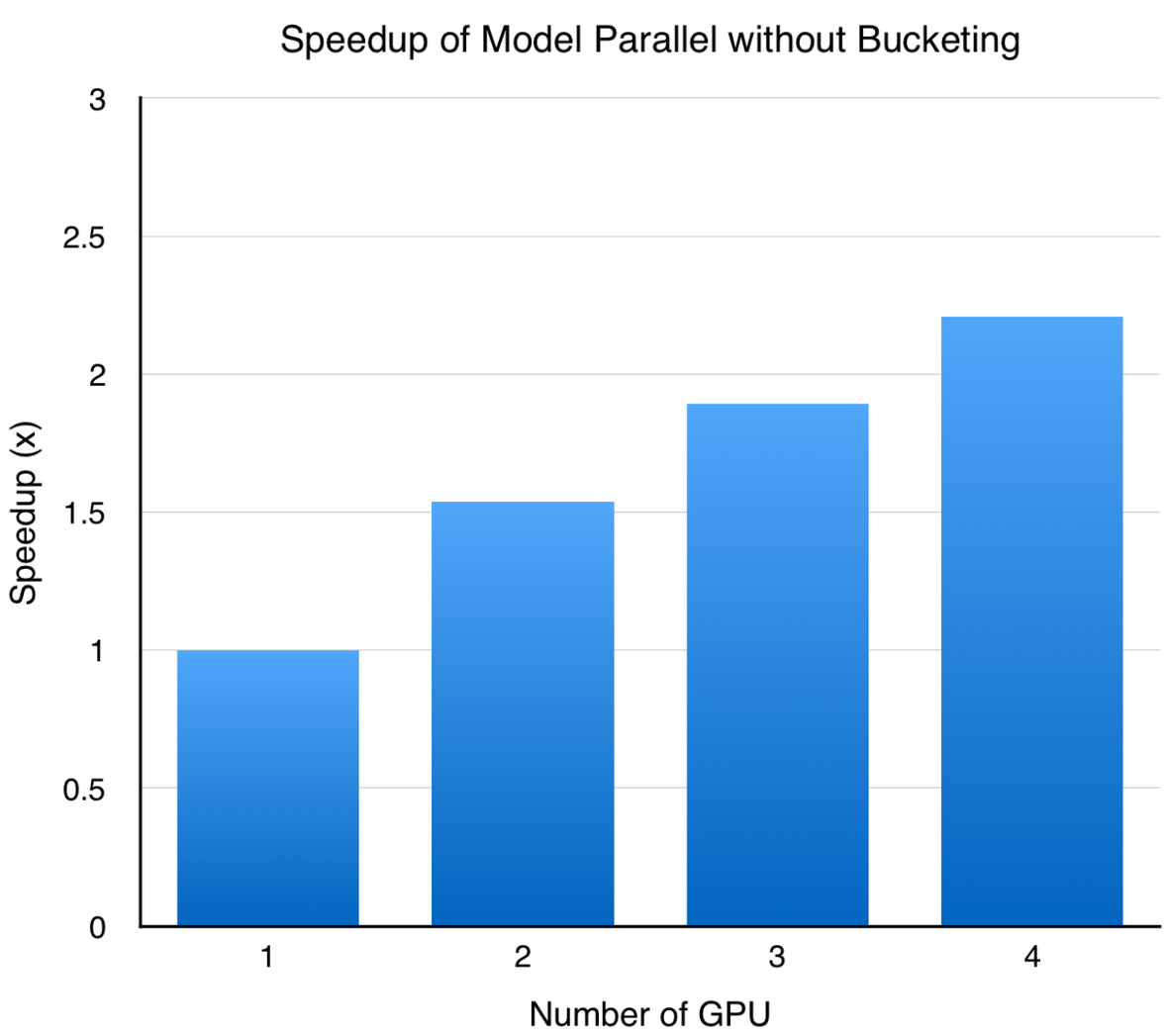
MP 101: Using multi GPU as a pipeline



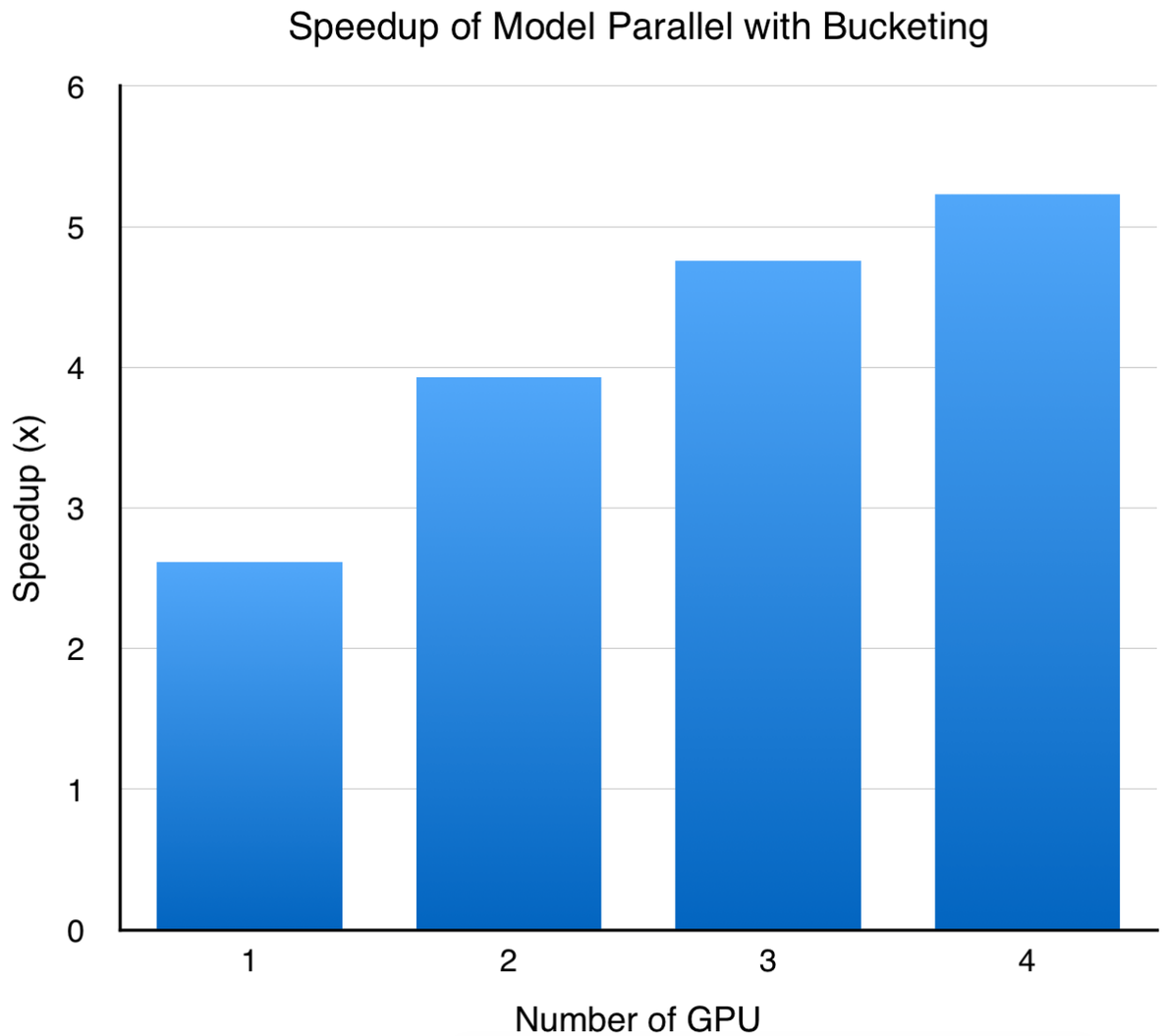
* **2.3x** speedup with 4 GTX750 on single machine



Combining MP with Bucketing



* **2.3x** speedup with 4 GTX750 on single machine



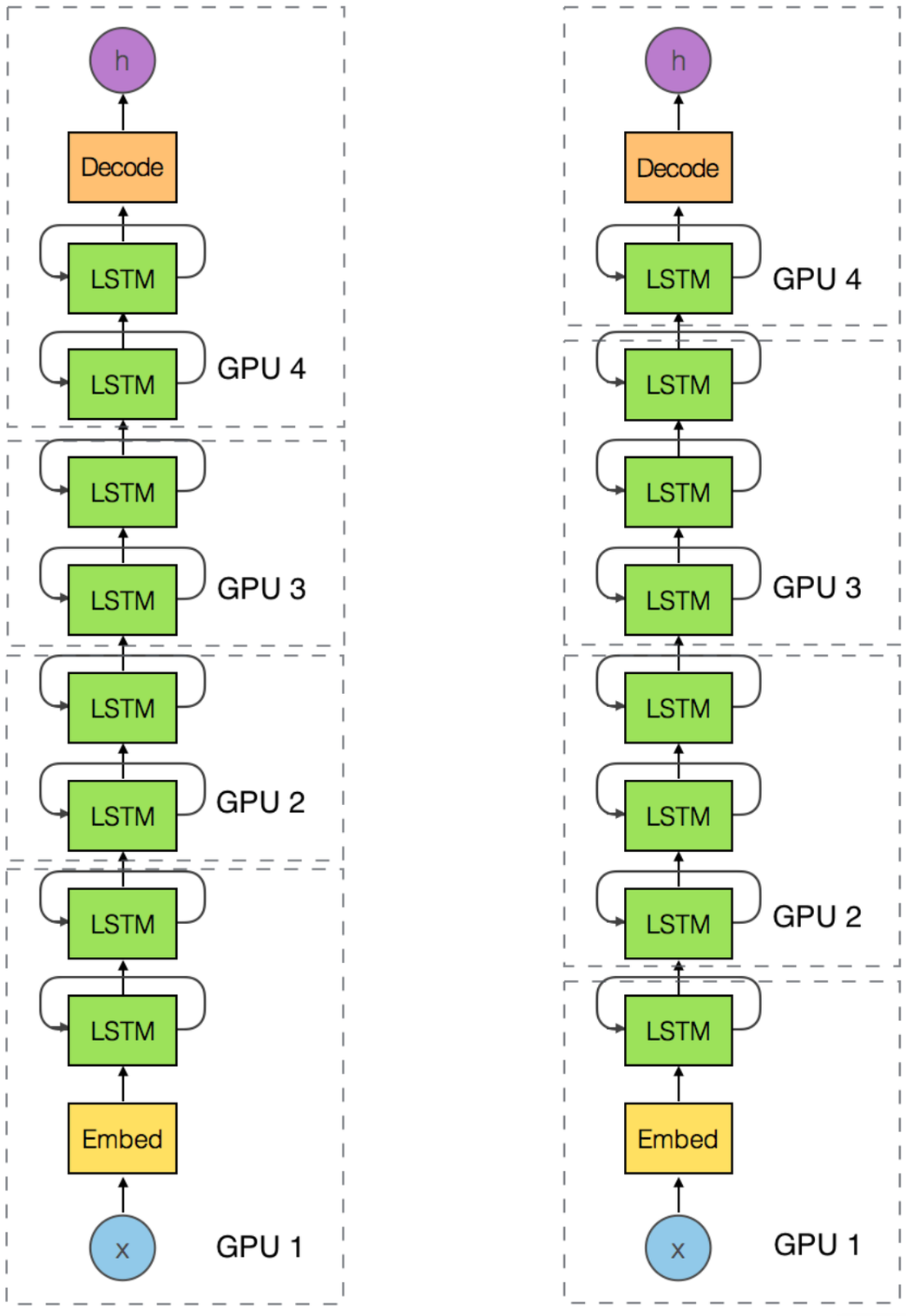
* **5.3x** speedup with 4 GTX750 and bucketing on single machine

Conclusion: MP is better in training LSTM on multi GPU

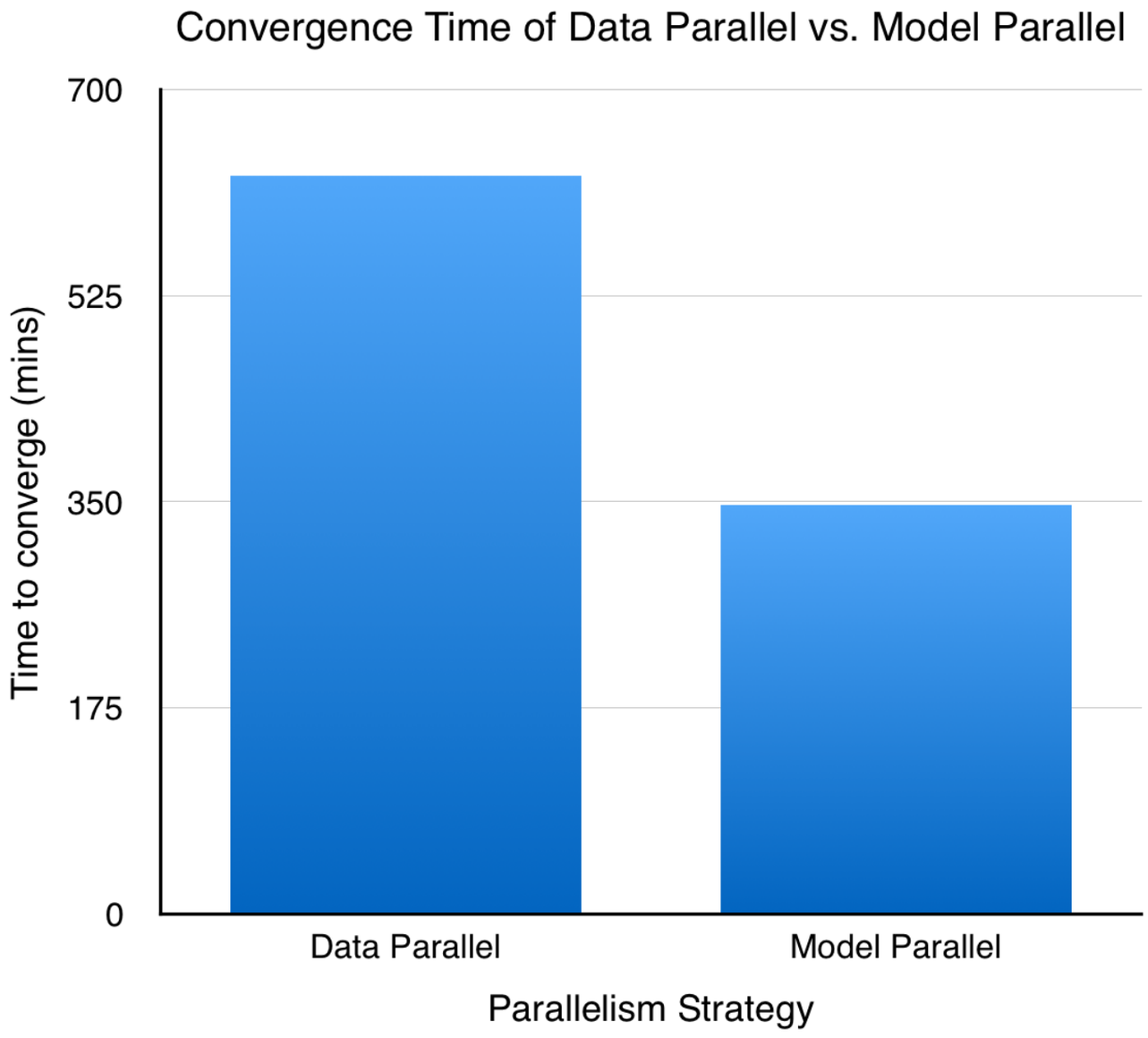
	Data Parallelism	Model Parallelism
Communication Cost	High	Low
Workload Balance	Balanced	Requires insight of both ML and System
Converge Speed	Slower	Faster

Q & A

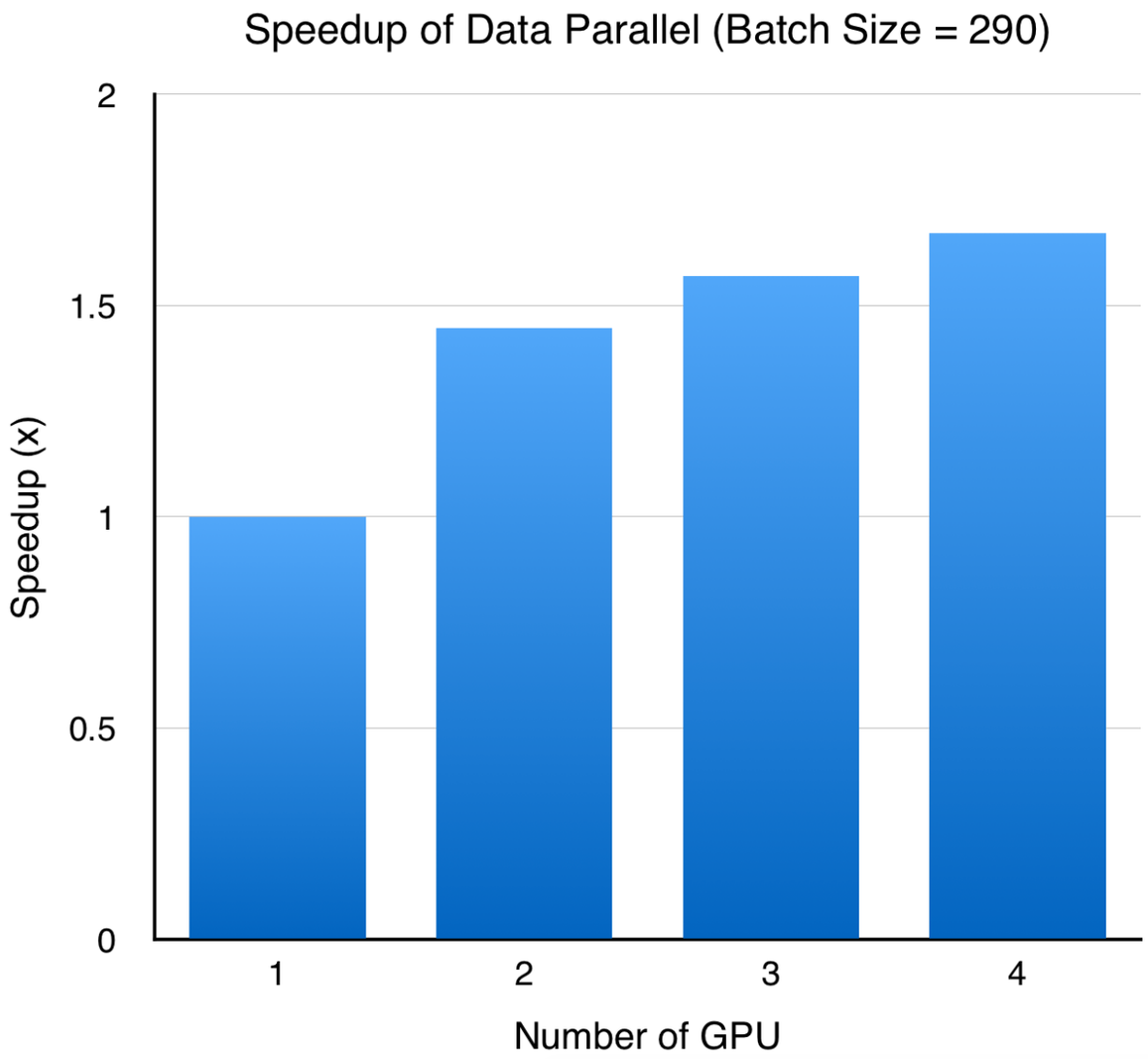
MP requires good knowledge of workload

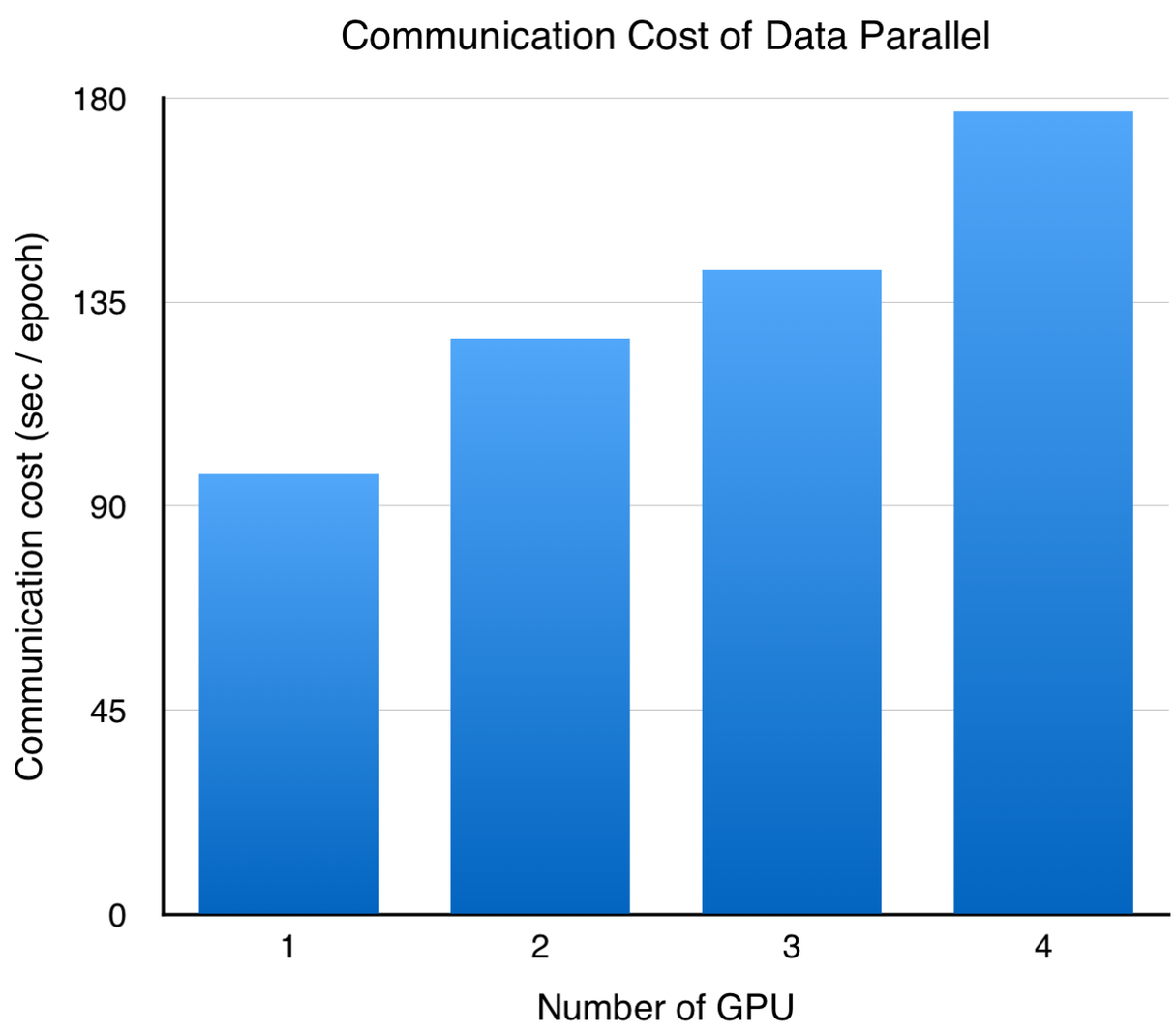


Converge Time of MP and DP



DP does not scale well on multi-GPU





* Experiment with no computation and only communication

Increasing computation intensity helps DP a little