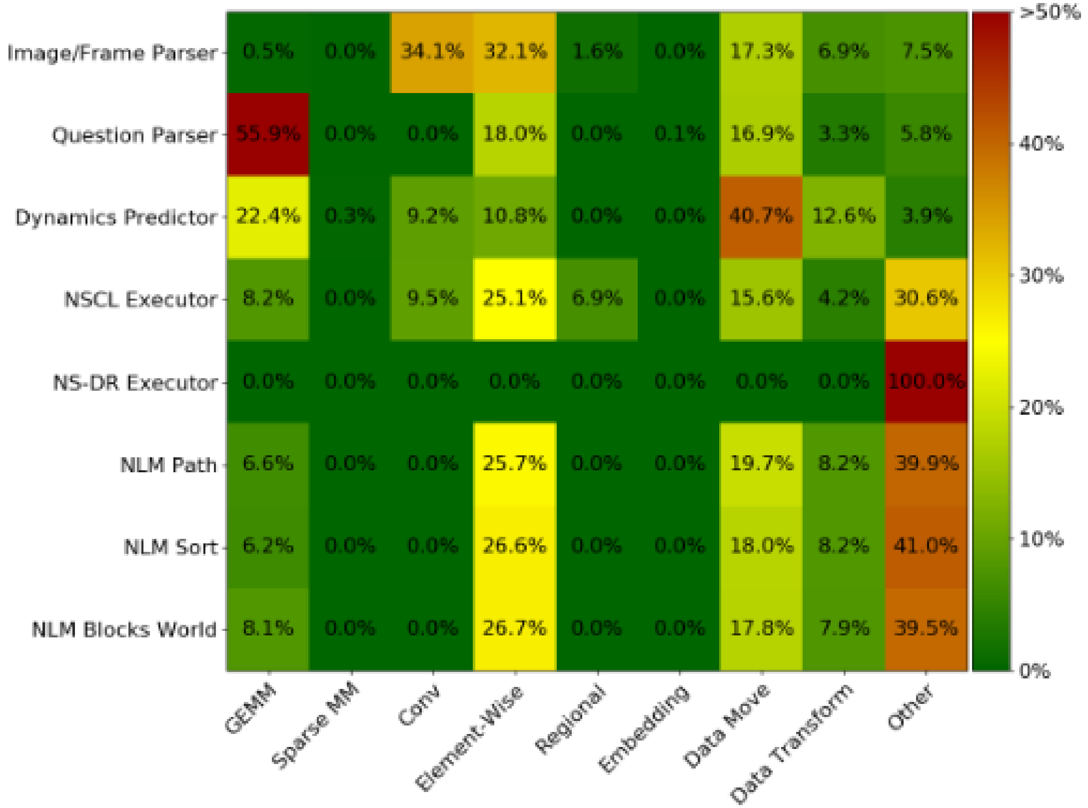
Neuro-Symbolic AI: An Emerging Class of AI Workloads and their Characterization

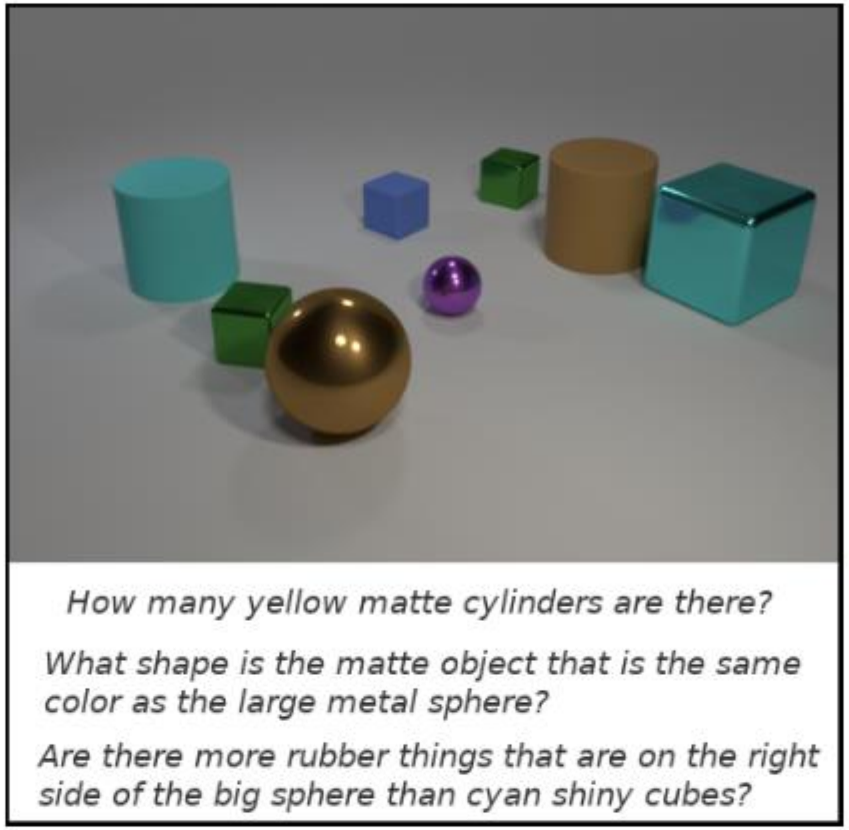
by Zachary Susskind, Bryce Arden, Lizy K. John, Patrick Stockton, Eugene B. John

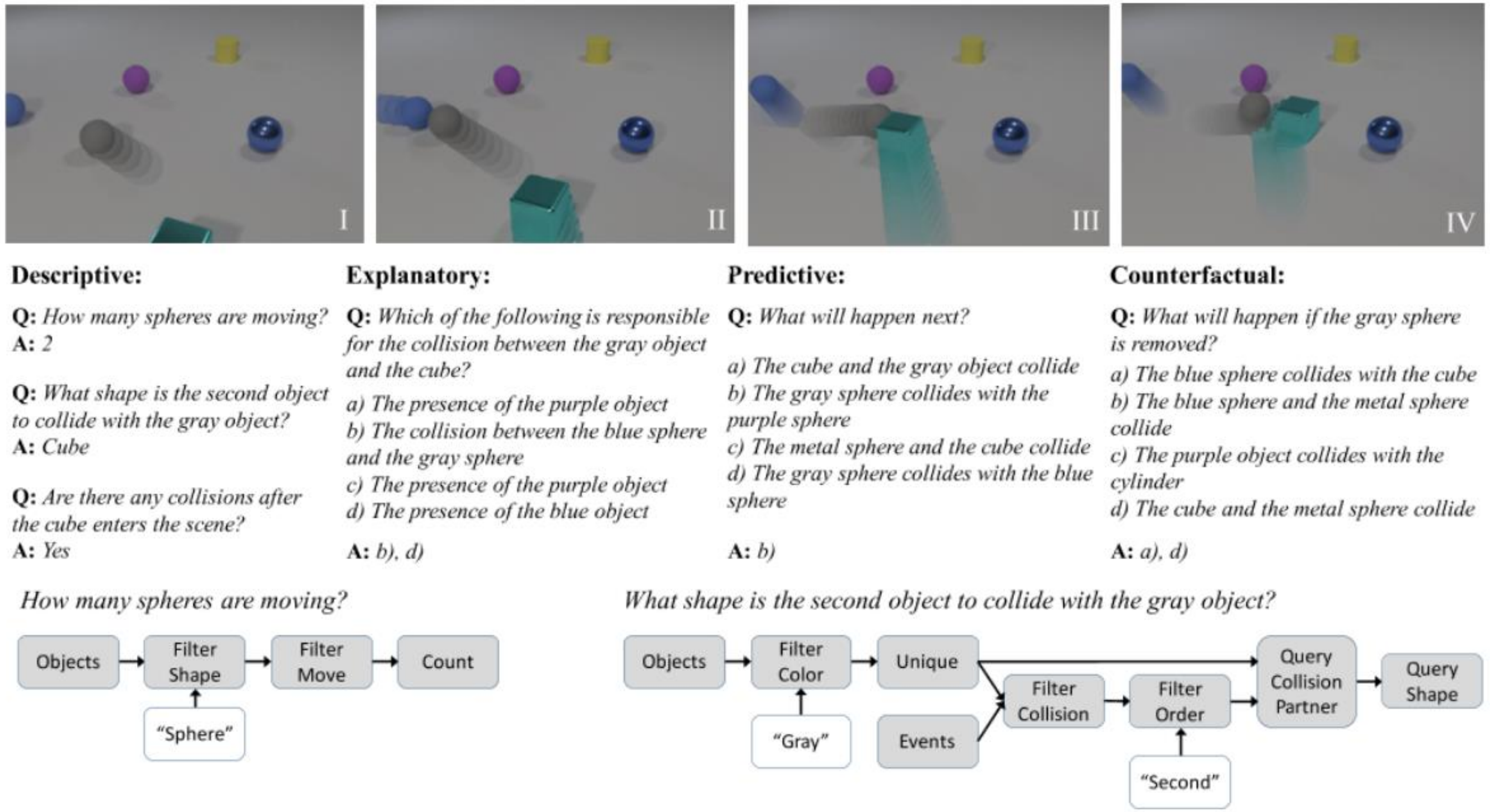


Outline

- Preliminaries
  - CLEVRER; Forward Chaining
  - Model and Data Parallelism
- Introduction and Motivation
- Methodology
- Results and Conclusion

Preliminaries (models)

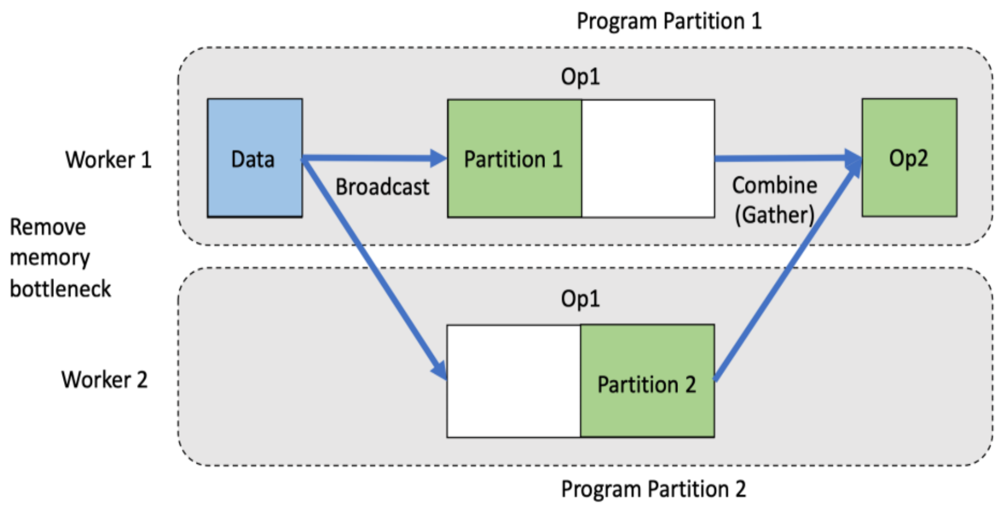




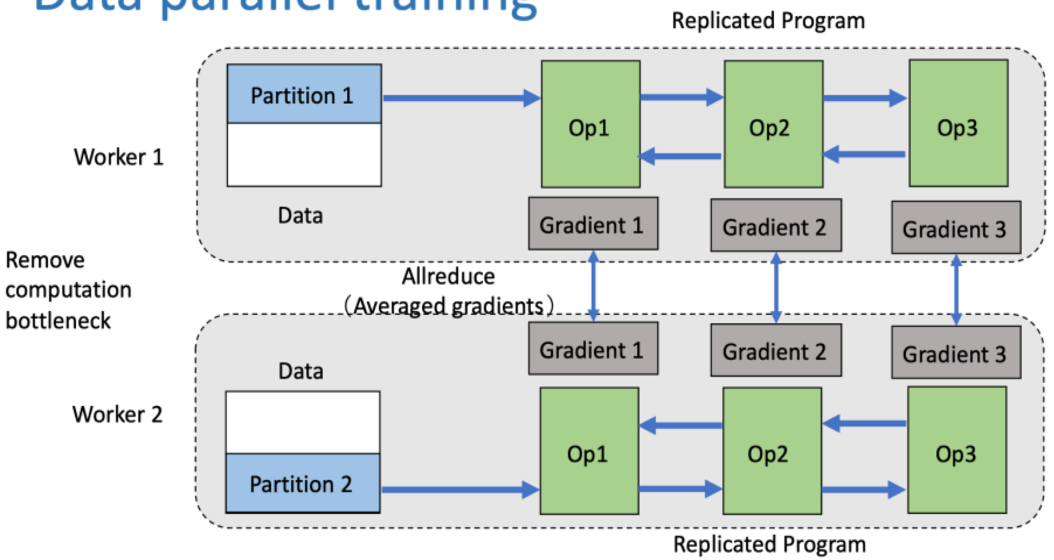

```
friendof(pedro, tom).
likes(X,Y) :- friendof(X, Y).
```

Preliminaries II (analysis)

Model parallel training: Intra-operator

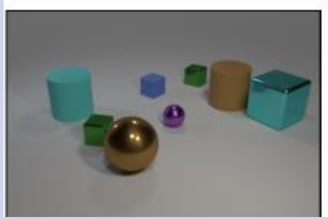
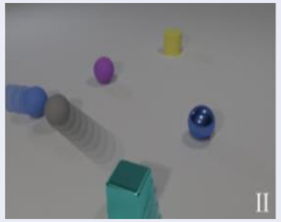
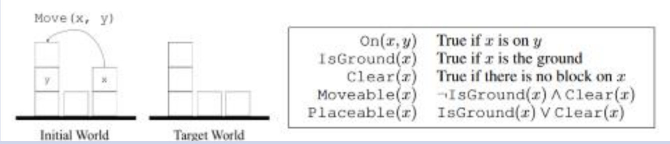


Data parallel training



Slides from Luo, Mai *Fundamentals of Distributed Machine Learning (2022)*

Introduction

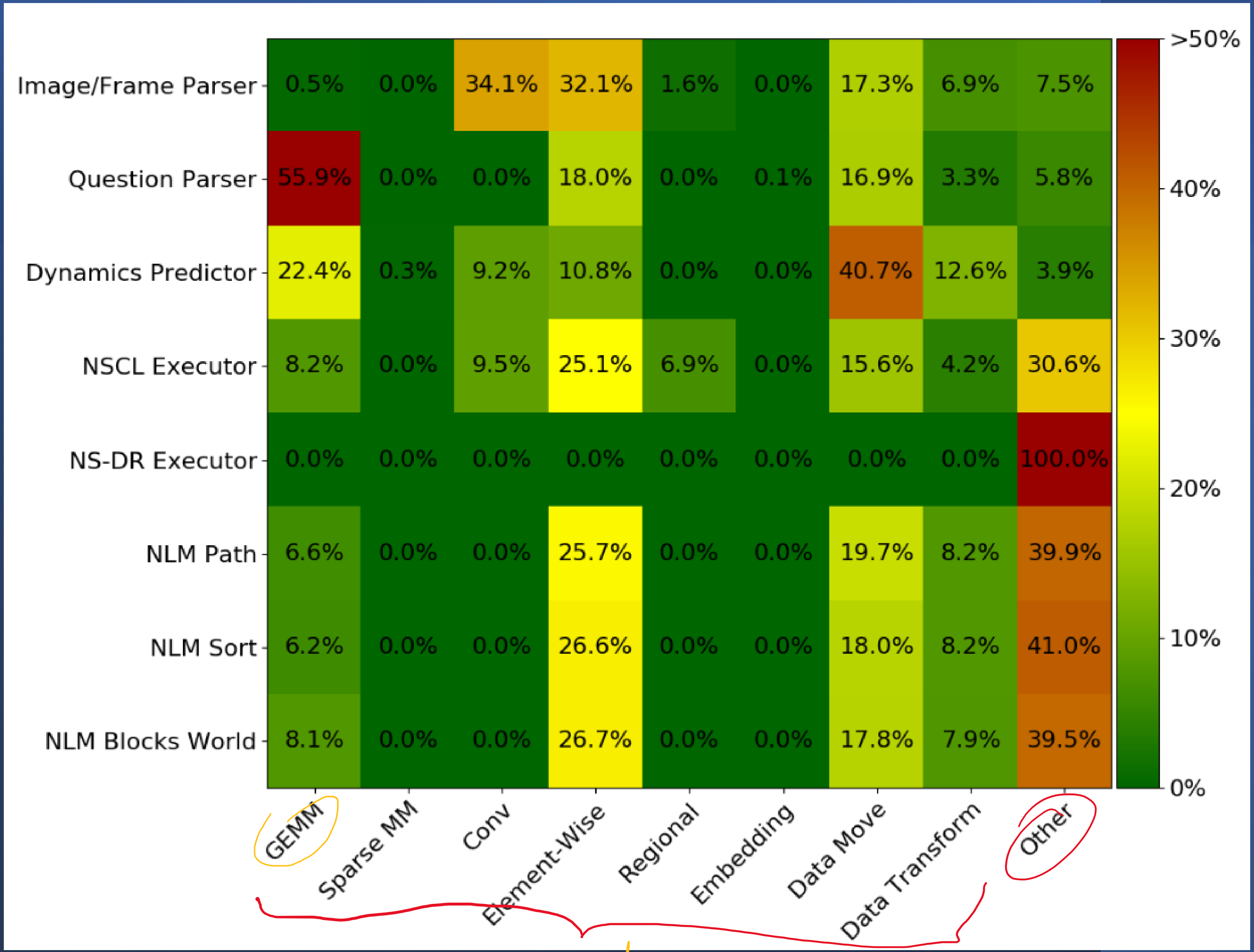
Model	Submodules	Task	Dataset
NSCL by IBM/MIT	• Image Parser (Mask R-CNN) • Question Parser (Open NMT) • Symbolic Executor	Query-driven Relational reasoning over images	CLEVR 
NS-DR by IBM/MIT	• Video Frame Parser (Mask R-CNN) • Question Parser (Open NMT) • Dynamics Predictor (Learned physics by PropNet) • Symbolic Executor	Query-driven relational reasoning over video	CLEVRER 
NLM by Google	• No Submodels	Program-driven reasoning	Sort, Family tree, and Block's world 

Motivation of function profiling

- Identify potential parallelism
- Identify bottlenecks to optimize

Function profiling based on runtime

Workload	Examples	Comments
Dense matrix multiplication		Highly parallelizable unless one matrix's dimension is small.
Sparse matrix multiplication		Look-up for non-zero values
Convolution		Theoretically parallelizable, but practically challenging. Use im2col algorithm to convert convolution into a GeMM, but need a lot of data movement
Element wise tensor	Activation function, normalization	
Regional operations	Pooling	
Embedding lookup	One-hot to embedding	Parallelization is challenging: training != testing (look-up table)
Data movement	Tensor duplication, host-device transfer or tensor assignment	
Data transformation	Transpose, tensor reordering, coalescing	



Operations

Note:
→ inference
↳ single input

TABLE III

RUNTIMES AND RUNTIME BREAKDOWNS FOR SINGLE INPUTS TO THE MODELS DISCUSSED IN THIS PAPER.

Model	GEMM	Sparse MM	Conv	Element-Wise	Regional	Embedding	Data Move	Data Transform	Other	Total
Image/Frame Parser	0.19ms	0ms	11.8ms	11.1ms	0.54ms	0ms	6.0ms	2.4ms	2.6ms	34.6ms
Question Parser	166ms	0ms	0ms	53.5ms	0ms	0.27ms	50.1ms	9.9ms	17.3ms	297ms
Dynamics Predictor	715ms	9.9ms	294ms	345ms	0ms	0ms	1300ms	403ms	125ms	3200ms
NSCL Executor	39.9us	0us	46.4us	122.4us	33.5us	0.0us	76.0us	20.5us	149.5us	488.3us
NS-DR Executor	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	12.9ms*	12.9ms
NLM Path	1.2s	0s	0s	4.7s	0s	0s	3.6s	1.5s	7.3s	18.3s
NLM Sort	2.6s	0s	0s	11.1s	0s	0s	7.5s	3.4s	17.1s	41.7s
NLM Blocks World	635ms	0ms	0ms	2100ms	0ms	0ms	1400ms	618ms	3100ms	7850ms

Results and Conclusion

Question parser's computational time depends on input sequence's length ≈ 22 words

Dynamics predictor could be faster by optimizing coalescing.

Symbolic program executors have small parallelization opportunities

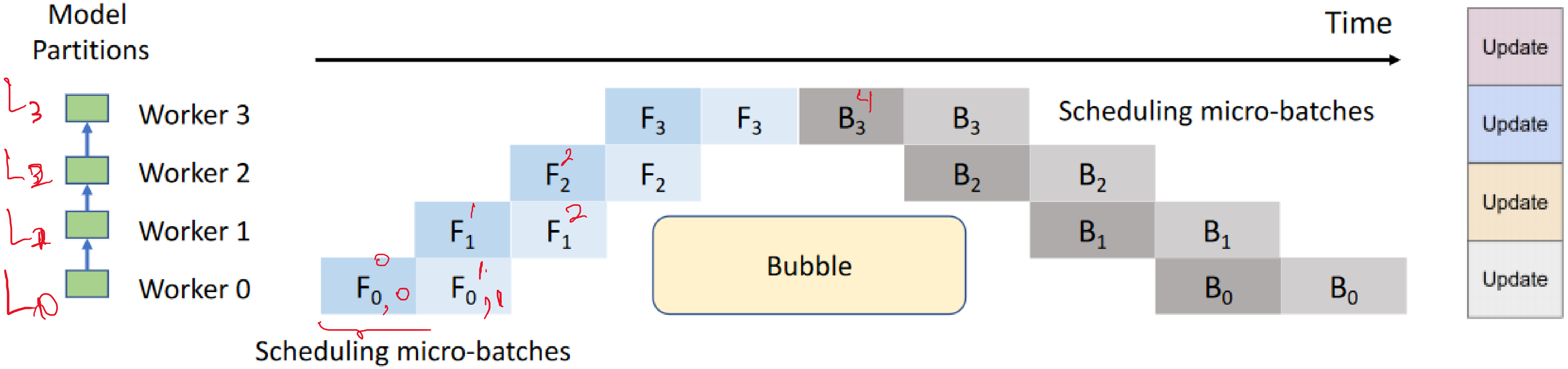
NLM also pose challenges for parallelization due to low operational intensity

Pipeline parallelism 1.01

$A^{t}_{i,j}$

i = worker
j = micro-batch
t = time

Optimising micro-batch size



- **Small micro-batch** reduces bubble size; but incur large micro-batch scheduling overheads
- **Large micro-batch** incurs large bubble; but come with small micro-batch scheduling overheads
- Optimal micro-batch size must **balance bubble size and scheduling overheads**

Slide taken from Luo, Mai *Fundamentals of Distributed Machine Learning (2022)*