Attention: Tutorial is being recorded

ASTRA-sim

AI System Tradeoff & Resource Analysis simulator

Synthetic Chakra Execution Trace Generation for Distributed ML Workloads

Changhai Man, Jinsun Yoo, Tushar Krishna
Georgia Institute of Technology

ASTRA-sim Tutorial
@ISCA 2026
June 26, 2026

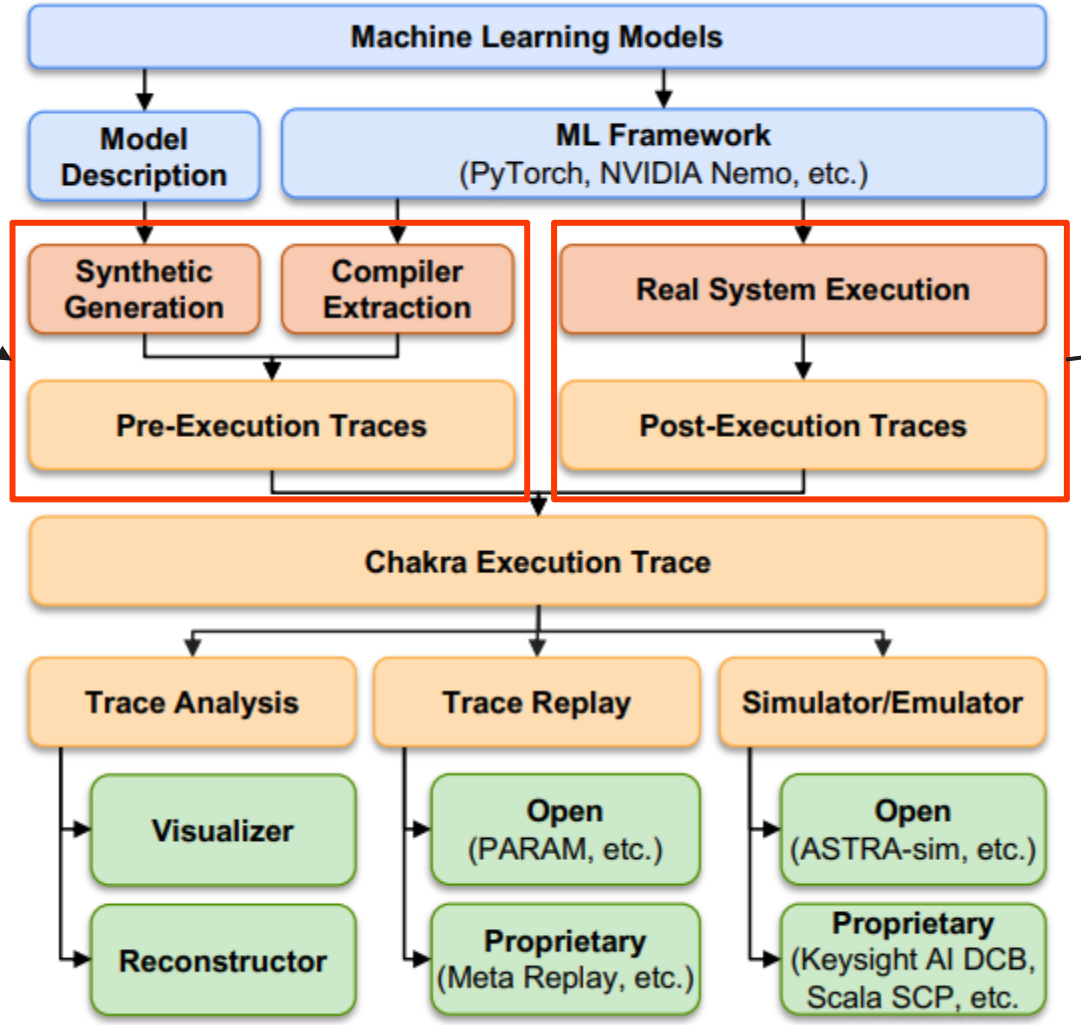
https://astra-sim.github.io/tutorials/isca-2026
https://astra-sim.github.io

STAGE — github, paper

Flint — paper

Obtaining Traces in Chakra Ecosystem

Focus of this talk



See Chakra wiki for details
https://github.com/mlcommons/chakra/wiki/

Why we need pre-execution Chakra ETs

Large-scale trace collection is expensive

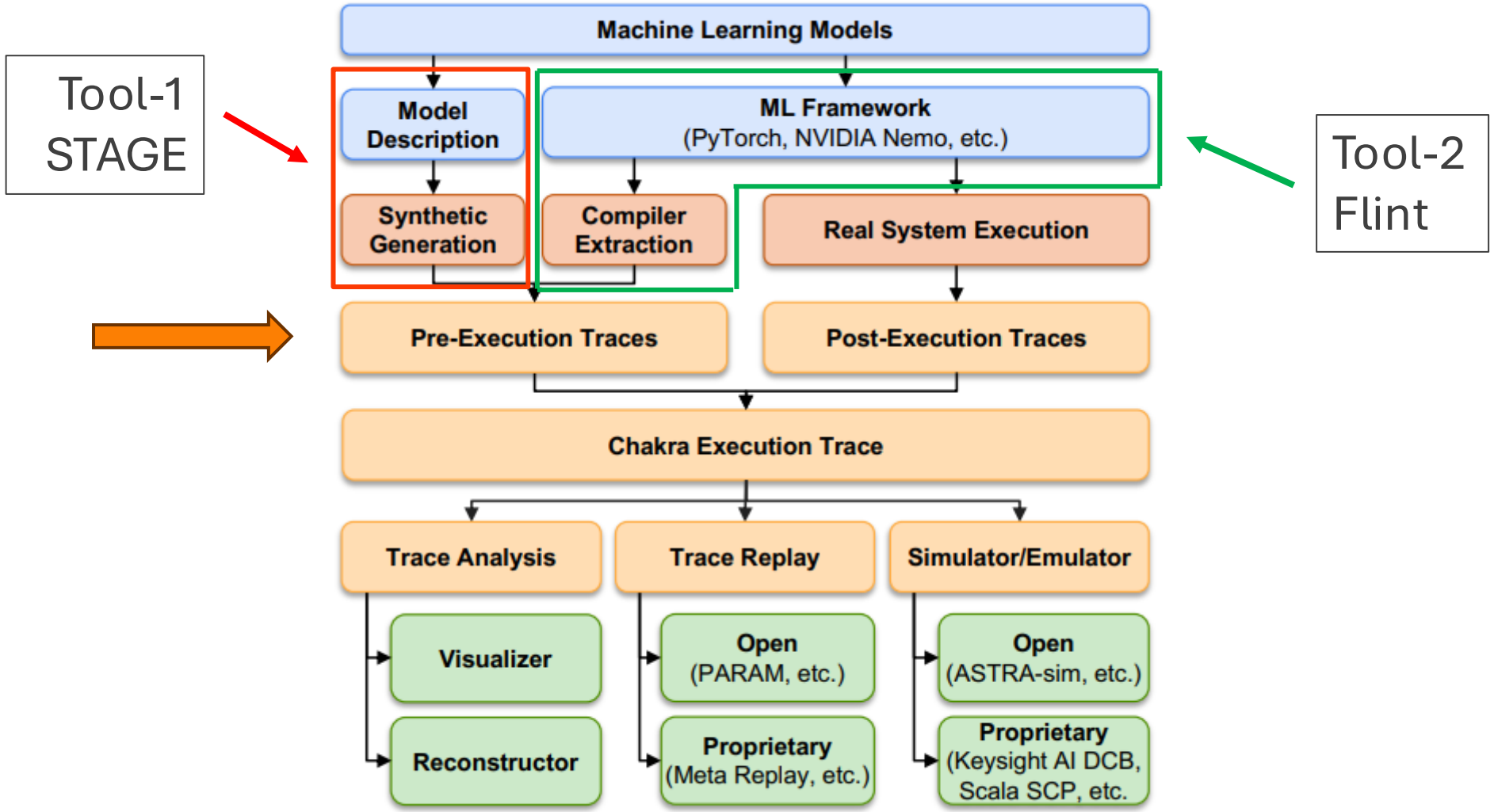
Real-system Chakra ET collection requires:

- Access to large GPU clusters
- Working distributed implementation
- Full workload execution
- Profiling and post-processing
- Re-collection when model / system / parallelism changes

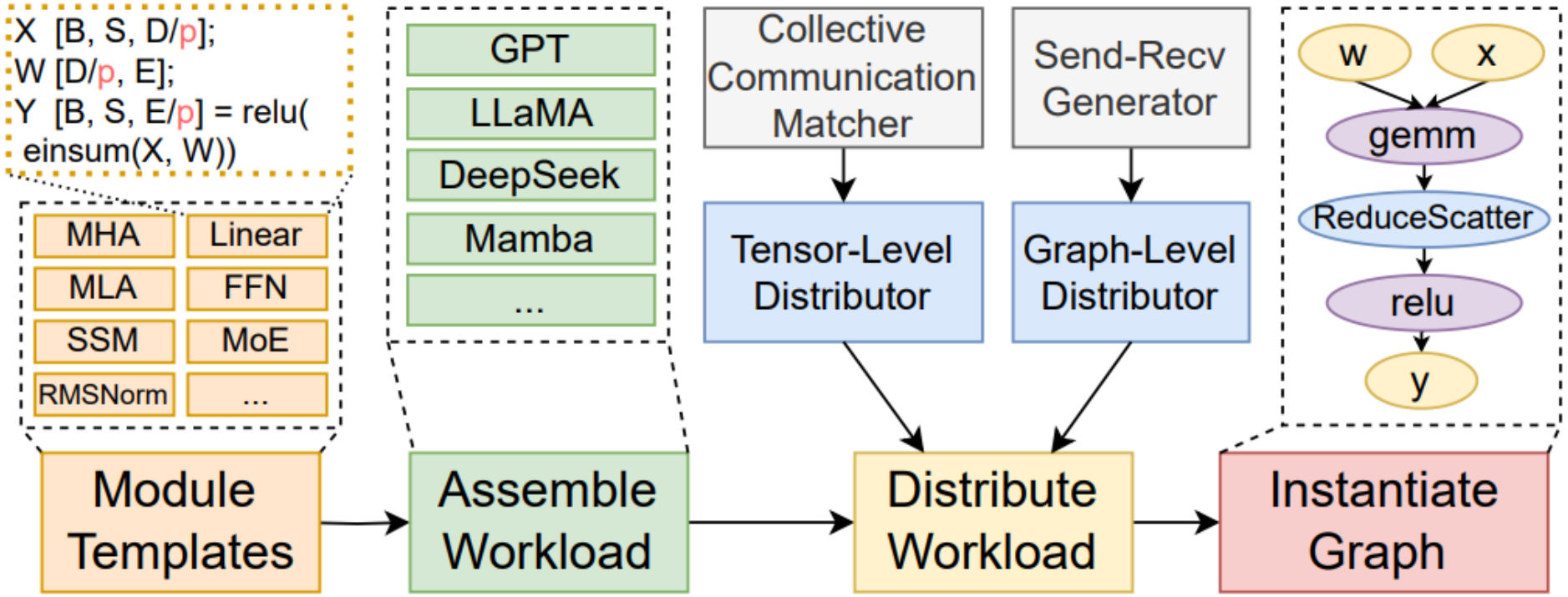
And it is unsuitable for large-scale **workload design space exploration**, and we need a cheap way to get Chakra ETs.

Solution: Pre-Execution Chakra ETs, that we synthesize useful Chakra ETs *without* running and tracing the workload from real system.

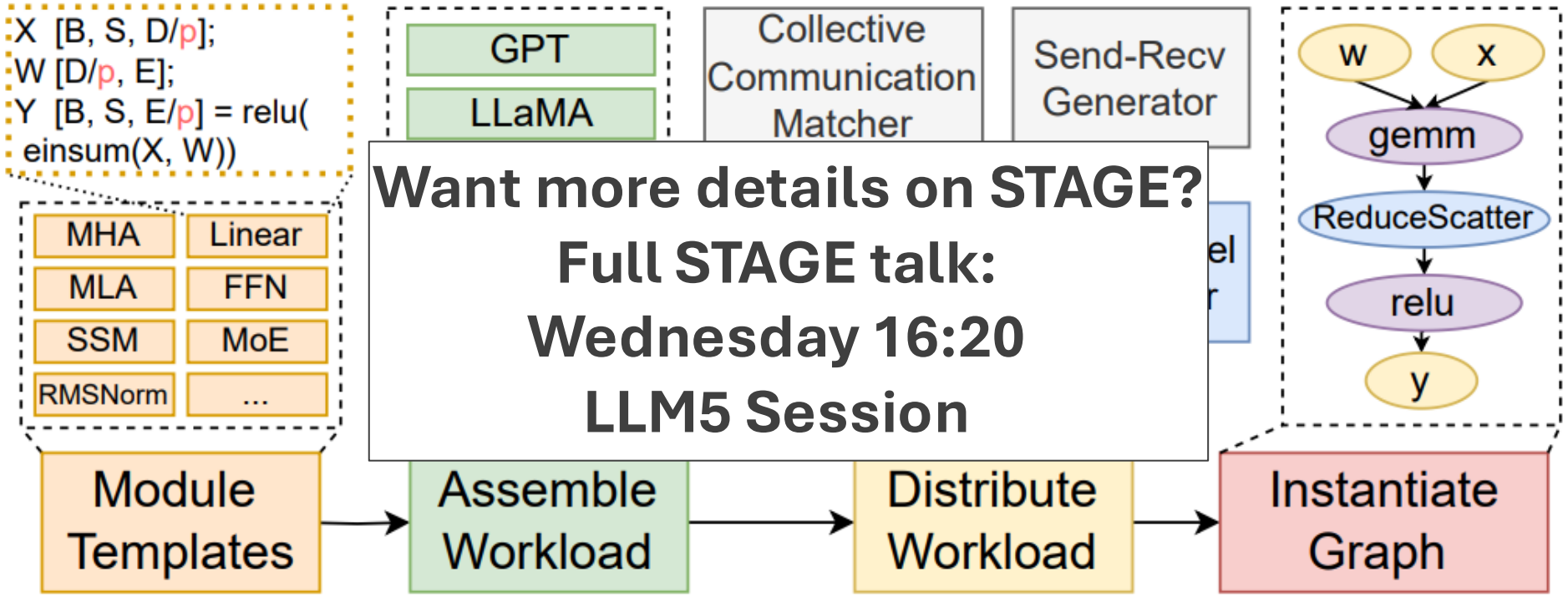
Pre-Execution Traces in Chakra Eco-System



STAGE: Chakra Generator through Symbolic Tensor Graph



STAGE: Chakra Generator through Symbolic Tensor Graph



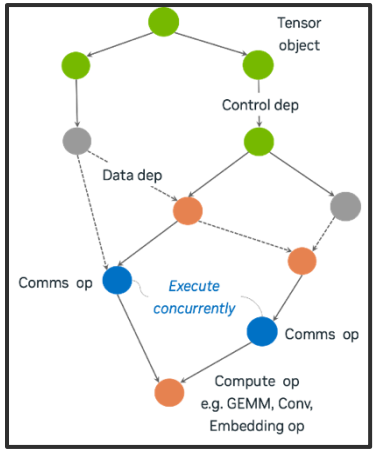
Flint: Chakra Generation from Source Code

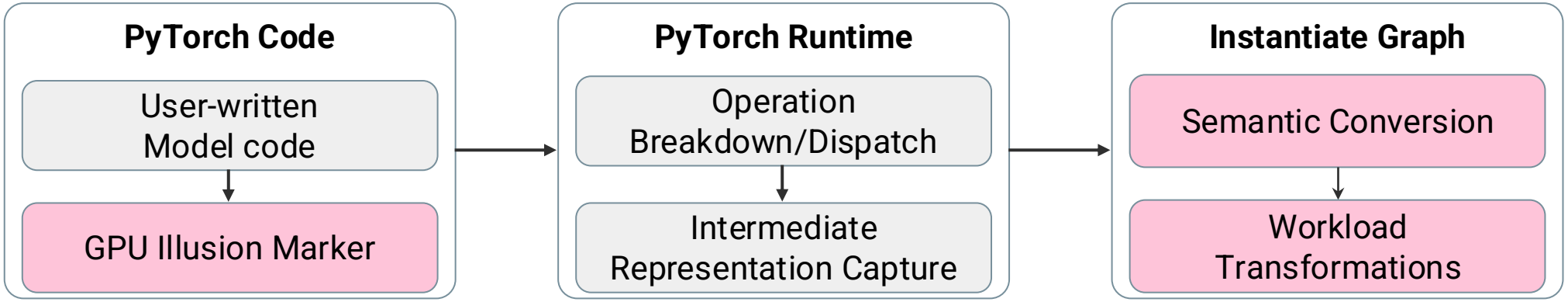
```python
import torch.nn as nn
from torch.distributed.tensor.parallel
  import parallelize_module

class Attention(nn.Module):
  def __init__(self):
    self.wq =
nn.Linear(dim,nhead*head_dim)

  def forward(x):
    xq,xk,xv = self.wq(x), # ...
    output = self.sdpa(xq,xk,xv)
    return self.wo(output)

def train():
  module = Attention()
  parallelize_module(module)
  return module(input)
```





STAGE Demo

Check following link for the demo scripts:

https://github.com/astra-sim/stage/tree/main/demo/isca26