

CogSys: Efficient and Scalable Neurosymbolic Cognition System via Algorithm-Hardware Co-Design

HPCA 2025

Zishen Wan^{†*}, Hanchen Yang^{†*}, Ritik Raj^{†*}, Che-Kai Liu[†], Ananda Samajdar[‡],
Arijit Raychowdhury[†], Tushar Krishna[†]

[†]Georgia Institute of Technology, Atlanta, GA

[‡]IBM Research, Yorktown Heights, NY

汇报人：宋青阳

2025年7月25日

Author

研究方向：计算机架构，VLSI和认知智能的交叉领域

研究聚焦于协同设计系统、架构及固态硬盘，以提升自主机器与神经符号AI的效能与鲁棒性，推动下一代具身代理应用（Embodied Agentic）的发展。

System :

- System for Resilient Autonomous Machines

Architecture:

- Architecture for Neuro-Symbolic and Embodied Intelligence
- Architecture for Efficient Autonomous Machines

Solid-State Hardware:

- Chip for Autonomous Machines and Memory-Centric Computing



Zishen Wan 万梓燊

Georgia Institute of Technology

zishenwan.github.io/

Contents

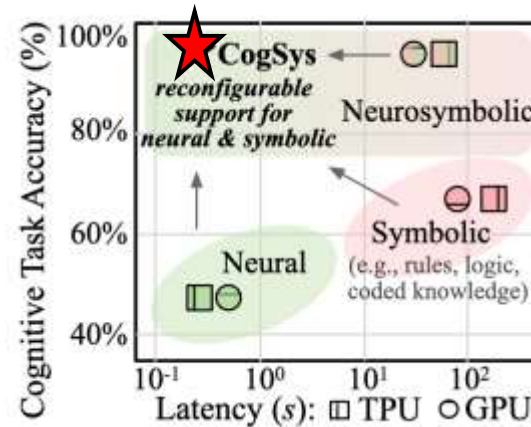
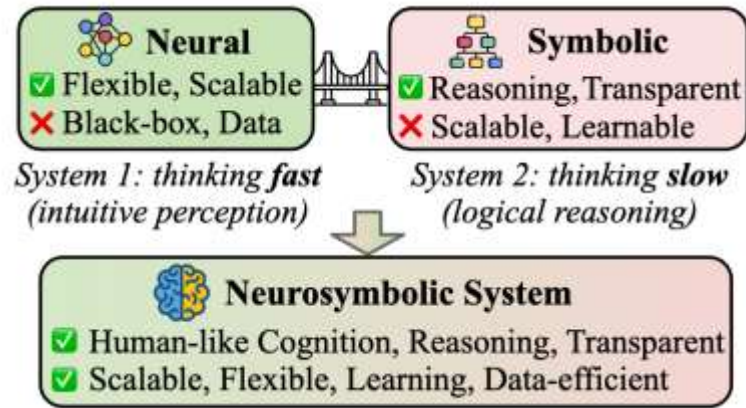
- Background
- Challenges
- Design
- Evaluation
- Thinking

Contents

- **Background**
- Challenges
- Design
- Evaluation
- Thinking

Background

Traditional Thinking System

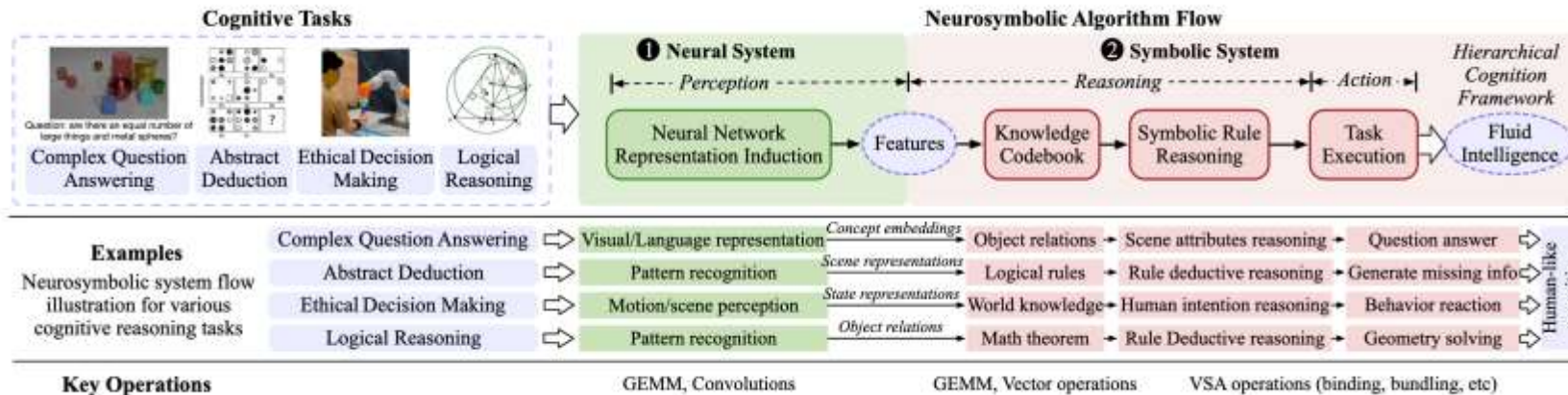


On GPU or TPU ?

CogSys

Better reasoning efficiency and cognitive capability.

Neurosymbolic Algorithm Flow



Contents

- Background
- **Challenges**
- Design
- Evaluation
- Thinking

Challenges

四种代表性Neurosymbolic AI工作的性能分析

TABLE I: Neurosymbolic models. Selected neurosymbolic AI workloads for analysis, representing a diverse of application scenarios.

Representative Neuro-Symbolic AI Workloads		Neuro-Vector-Symbolic Architecture [33]	Multiple-Input-Multiple-Output Neural Networks [60]	Probabilistic Abduction via Learning Rules in Vector-symbolic Architecture [32]	Probabilistic Abduction and Execution Learner [96]
Abbreviation		NVSA	MIMONet	LVRF	PrAE
Learning Approach		Supervised/Unsupervised	Supervised	Supervised	Supervised/Unsupervised
Compute Pattern	Neuro	CNN	CNN/Transformer	CNN	CNN
	Symbolic	VSA binding/unbinding (Circular Conv)	VSA binding (Circular Conv)	VSA binding/unbinding (Circular Conv)	Probabilistic abduction
Application Scenario	Use Case	Spatial-temporal reasoning, Fluid intelligence, Abstract reasoning	Multi-input simultaneously processing with single CNN/Transformer	Probabilistic reasoning, Analogy reasoning, Out-of-distribution (OOD) data processing	Spatial-temporal reasoning, Fluid intelligence, Abstract reasoning
	Advantage vs. Neural Model	Higher joint representation efficiency, Better reasoning capability, Transparency	Higher throughput, Lower latency, Compositional compute, Transparency	Stronger OOD handling capability, One-pass learning, Higher flexibility, Transparency	Higher generalization, Transparency, Interpretability, Robustness

现有Neurosymbolic AI
存在部署与扩展挑战！

计算层面

内存层面

流水线层面

Challenges

① Compute Kernels

Neurosymbolic AI包含异构的神经和符号内核，符号操作在CPU/GPU/TPU上执行效率低，硬件利用率和缓存命中率低，形成延迟瓶颈。

② Memory

由于向量符号操作需要大量元素流式传输，符号操作受内存限制。**Symbolic codebooks**（符号代码本）通常占用大量内存，计算期间需要缓存大量中间结果。

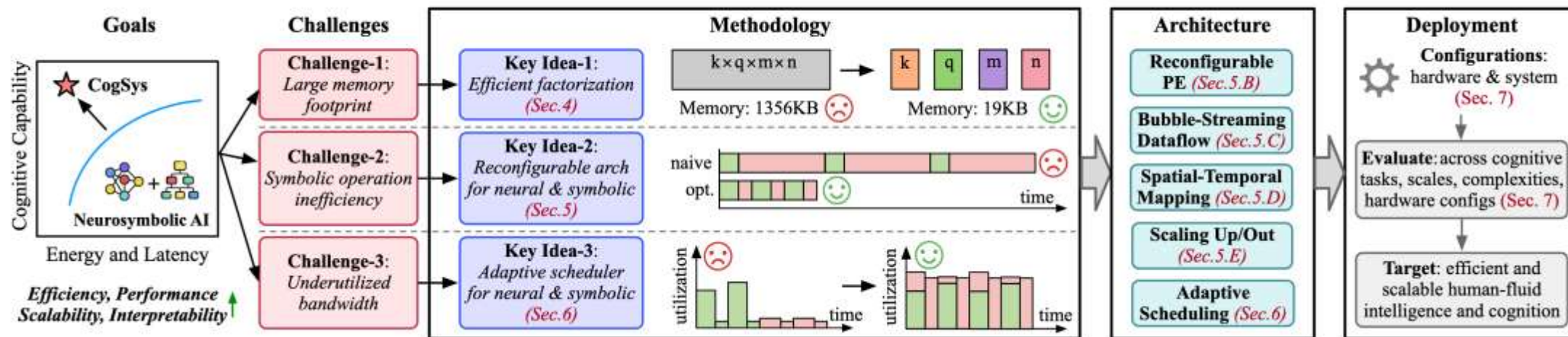
③ Dataflow and scalability.

相比于CNN，神经符号工作负载具有更复杂的控制流。符号模块通常具有**数据流不规则、数据依赖和顺序处理**的特性，导致并行可扩展性和效率的低下。


Challenges

- ① Algorithm level → □ 针对硬件的代码本优化，以减少内存占用和延迟
- ② Hardware level → □ 架构和数据流需针对VSA操作高效优化，且支持神经/符号内核的重构
- ③ System level → □ 架构需能高效且自适应地调度各种神经符号工作任务

CogSys overview

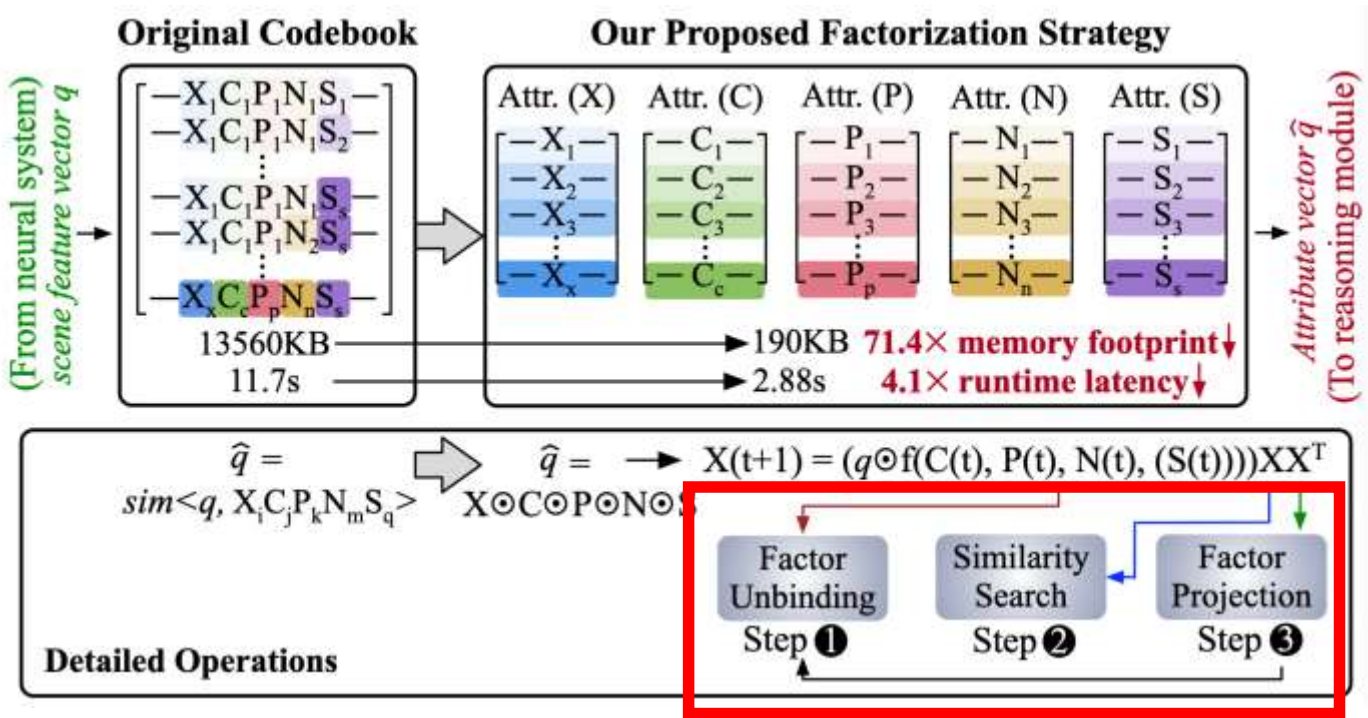


Contents

- Background
 - Challenges
 - **Design**
 - Evaluation
 - Thinking
- 
- A vertical blue line with three horizontal tick marks connects the 'Design' and 'Evaluation' items in the list. To the right of this line, three text labels are aligned with the tick marks: 'Algorithm level' in red, 'Hardware level' in gray, and 'System level' in gray.
- Algorithm level
 - Hardware level
 - System level

Design—Algorithm Optimization

符号分解策略（Symbolic Factorization Strategy）



本质
复杂目标 → 拆成若干因素
 q

迭代至收敛，即匹配所有查询条件

减小符号知识库内存占用

➤ Step1 : Factor Unbinding

去除其他因素的影响，提取组成查询向量 q 的每个因子 x

$$\tilde{x}^i(t) = q \oslash \prod_{\substack{f=1 \\ f \neq i}}^F \hat{x}^f(t)$$

➤ Step2 : Similarity Search

将解绑后的估计向量，与候选因子做点积，找出最相似者
(点积越大，代表越相似)

$$\alpha_i(t) = \tilde{x}^i(t) \cdot X_i$$

➤ Step3 : Factor Projection

利用上述相似度作为权重，对所有候选向量做线性加权，
得到下一轮估计

$$\hat{x}^i(t+1) = \text{sign}(\alpha_i(t) \cdot X_i^\top)$$

Design—Algorithm Optimization

随机性优化 (Stochasticity)

在Step 2、3 中注入高斯噪声、注入随机性以减少分解所需的迭代次数

运算精度优化 (Operator precision optimization)

为减少内存开销，对工作应用量化 (Quantization) 技术，模块采用8-bit 浮点或整数运算

➤ Step1 : Factor Unbinding

去除其他因素的影响，提取组成查询向量 q 的每个因子 x

➤ Step2 : Similarity Search

将解绑后的估计向量，与候选因子做点积，找出最相似者
(点积越大，代表越相似)

➤ Step3 : Factor Projection

利用上述相似度作为权重，对所有候选向量做线性加权，
得到下一轮估计

Algorithm optimization impact

	Accuracy (higher is better)	Latency (lower is better)	Memory (lower is better)
Factorization	Increase	Reduce	Reduce
Stochasticity	Increase / Reduce	Reduce	No Impact
Low-Precision	Reduce	Reduce	Reduce

Contents

- Background
- Challenges
- **Design**
- Evaluation
- Thinking

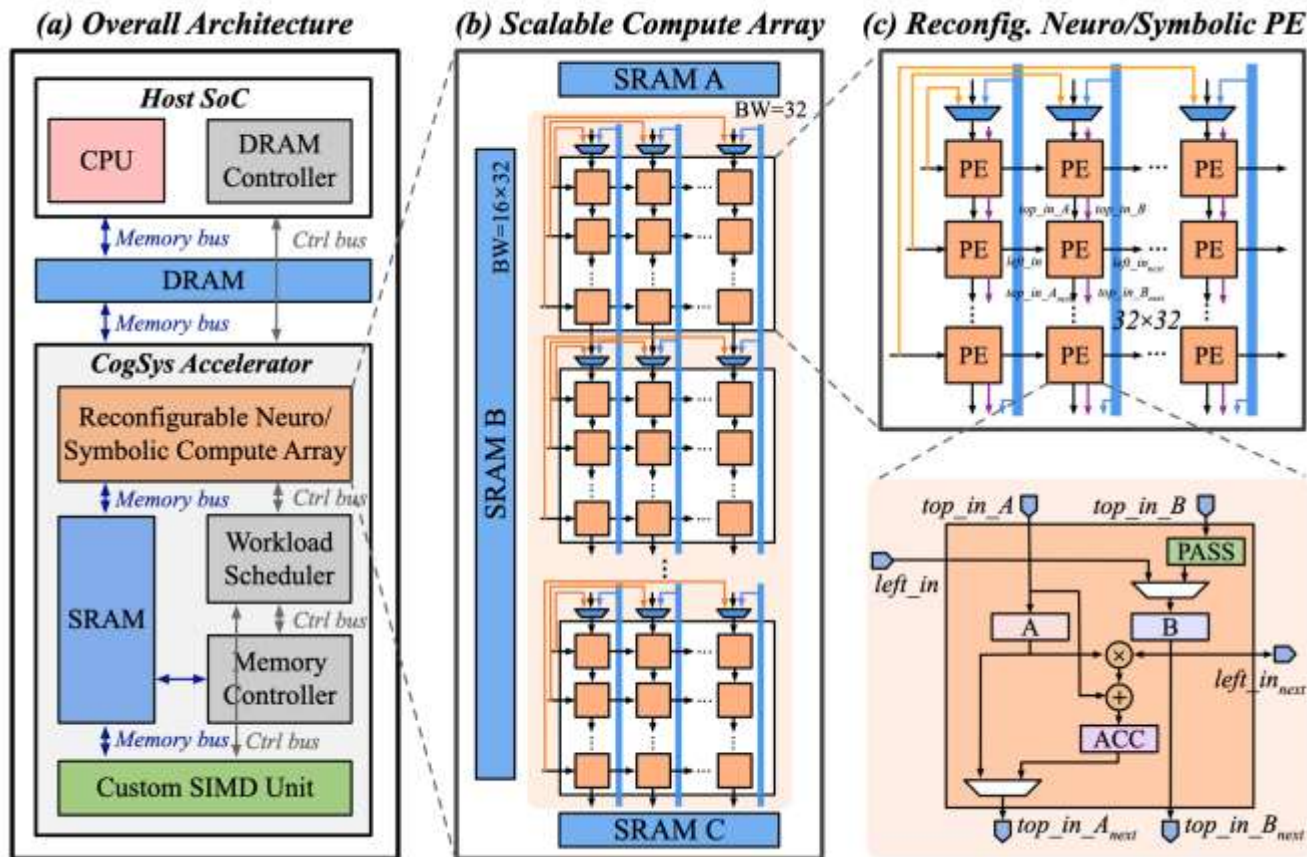
Algorithm level

Hardware level

System level

Design—Hardware Architecture

CogSys Architecture Overview



CogSys architecture features

- *reconfigurable neuro/symbolic processing elements (nsPE)*
- *bubble streaming (BS) dataflow*
- *adaptive spatial-temporal (ST) mapping*
- *scalable array architecture*
- *customized SIMD units*

Design—Hardware Architecture

Reconfigurable Neuro/Symbolic PE (nsPE)

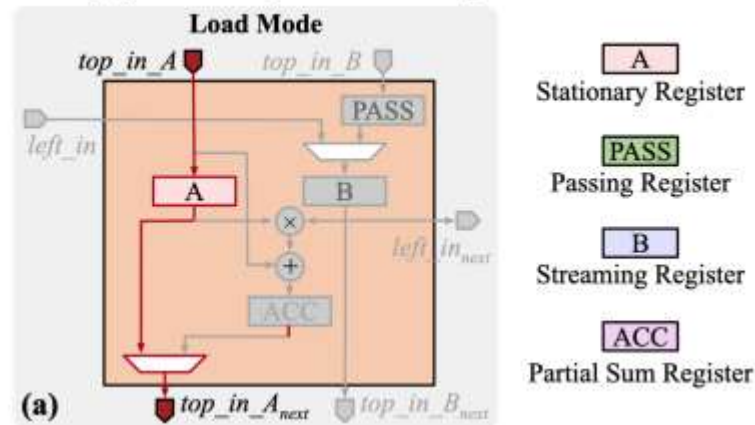
核心思想：与其为神经运算和符号运算设计两种不同的、独立的处理单元，不如设计一种**可重构**的处理单元（nsPE），让它能够灵活切换模式来执行这两种任务，这样就减小了硬件面积。

三种模式

加载模式

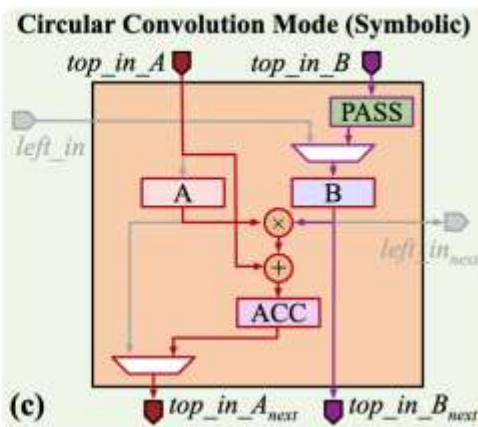
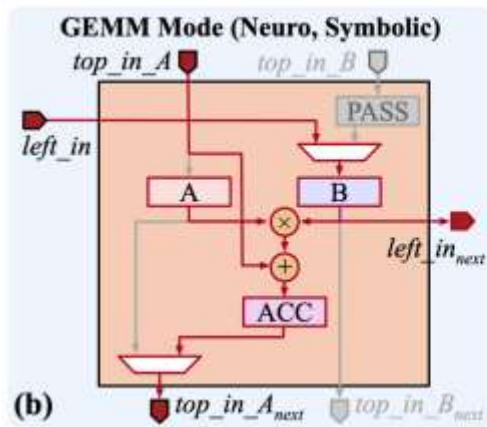
目的：这是计算前的准备阶段，用于把数据预加载到 nsPE 的寄存器中。

Reconfig. Neuro/Symbolic PE Operation Mode



GEMM模式

目的：执行矩阵乘法（GEMM）或其他类似运算，这是神经网络计算的核心。此时 nsPE 就等同于一个标准的 TPU处理单元。



循环卷积模式

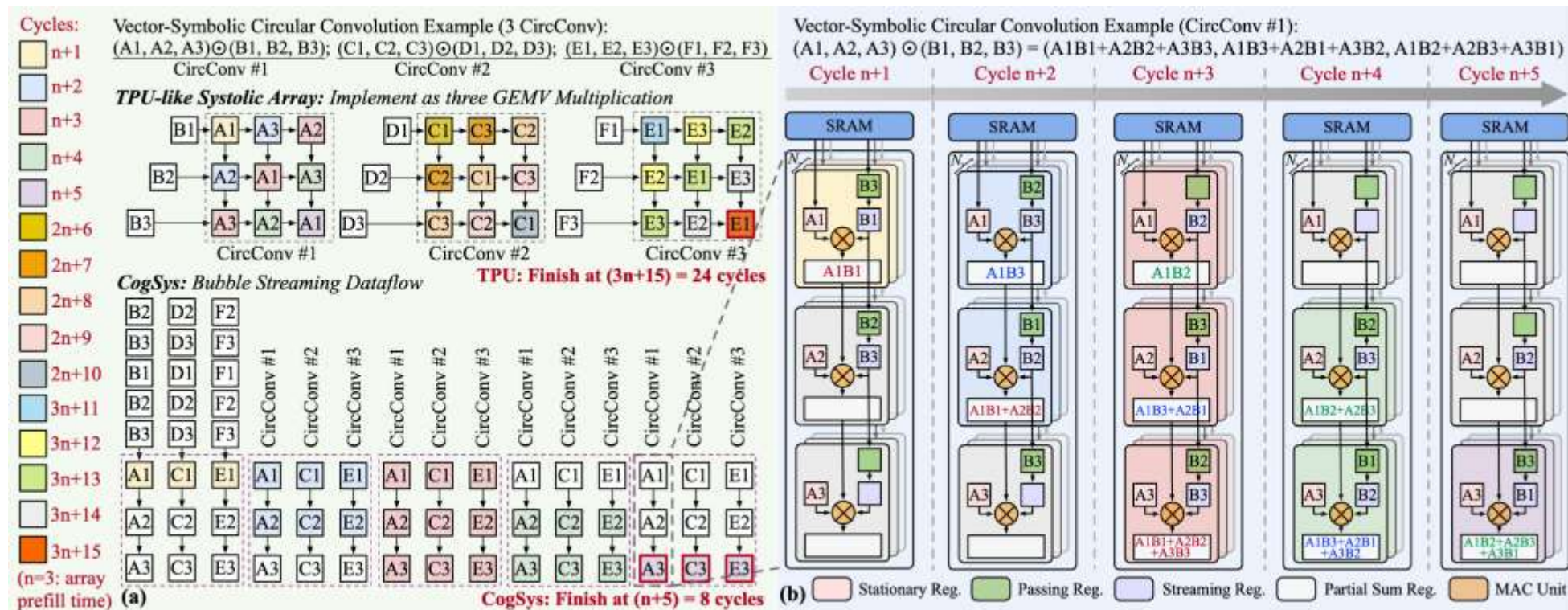
目的：执行循环卷积，这是向量符号架构中的核心运算。

Design—Hardware Architecture

Efficient Bubble Streaming Dataflow

提出了一种硬件数据流（Bubble），即在处理单元中加入一个“气泡”（一个周期延迟），相较于TPU：

- 把 $O(d^2)$ 的内存占用降低到了 $O(d)$
- 把串行的计算模式变成了并行的列式计算
- 把一个内存密集型的低效操作，转变成了计算密集型的高效操作

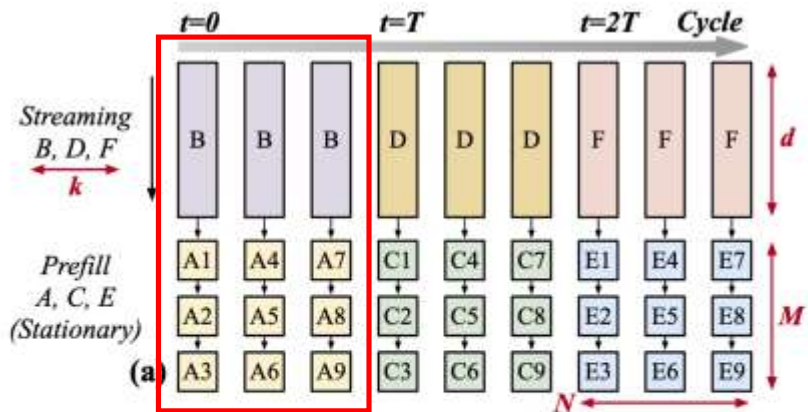


Design—Hardware Architecture

Adaptive Spatial and Temporal Mapping Strategy

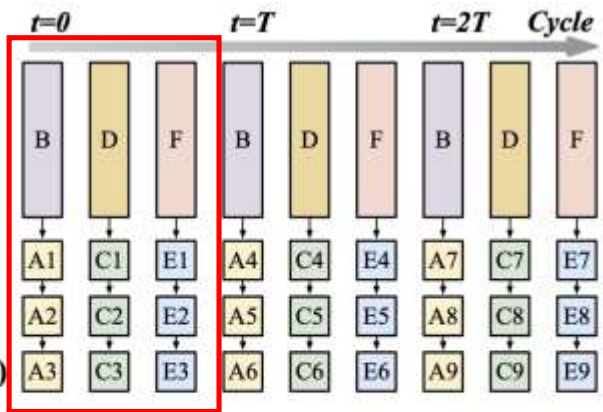
空间映射 (Spatial Mapping)

核心思想：将一个计算问题在空间上分解，并映射到硬件的所有空间资源上 (N列)



时间映射 (Temporal Mapping)

核心思想：将多个独立任务映射到不同的硬件空间资源上，让它们时间上并行



	(a) Spatial Mapping	(b) Temporal Mapping
Latency	$k \times \lceil d / (N \times M) \rceil \times T$	$\lceil k / N \rceil \times \lceil d / M \rceil \times T$
Mem Reads under full util.	$2d$ per T cycles	$(d + M) \times N$ per T cycles

一次完成一个任务 (串行)，
对内存带宽友好，但总延迟高

N个任务并行，总延迟低，
但对内存压力大

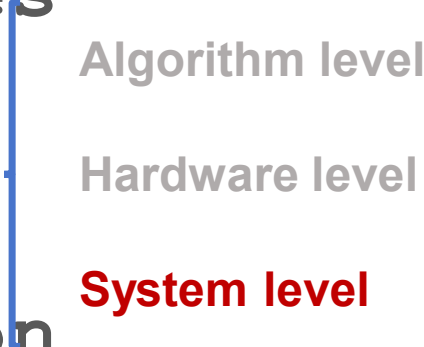
何时选择空间映射？

当任务数量 k 很少，但向量维度 d 非常巨大，远超单列的处理能力 ($d \gg M$) 时。此时应当将所有硬件资源 (N列) 集中起来加速这一个大任务。

何时选择时间映射？

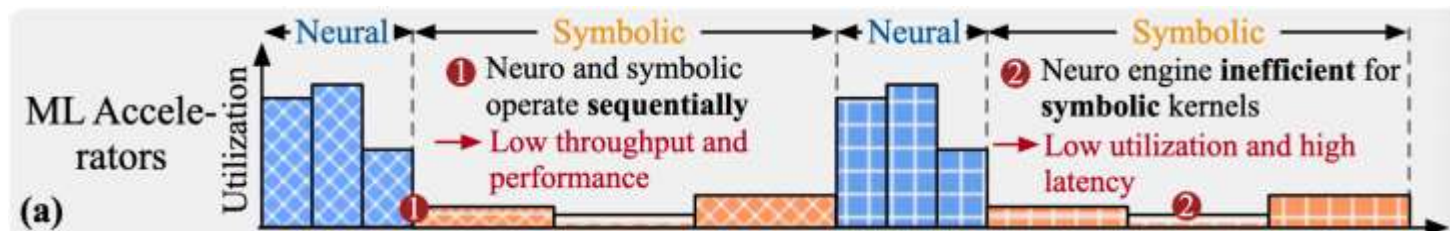
当任务数量 k 非常大时。此时，并行处理带来的好处远远大于增加的内存带宽开销。将 k 个任务分成 k/N 批次并行处理，延迟会大大降低。

Contents

- Background
 - Challenges
 - **Design**
 - Evaluation
 - Thinking
- 
- A vertical blue line with three horizontal tick marks connects the 'Design' and 'Evaluation' items in the list. To the right of this line, the following text is aligned with the tick marks:
- Algorithm level
 - Hardware level
 - System level**

Design—Scheduling Strategy

Neurosymbolic system-level challenges (on traditional GPU/TPU)

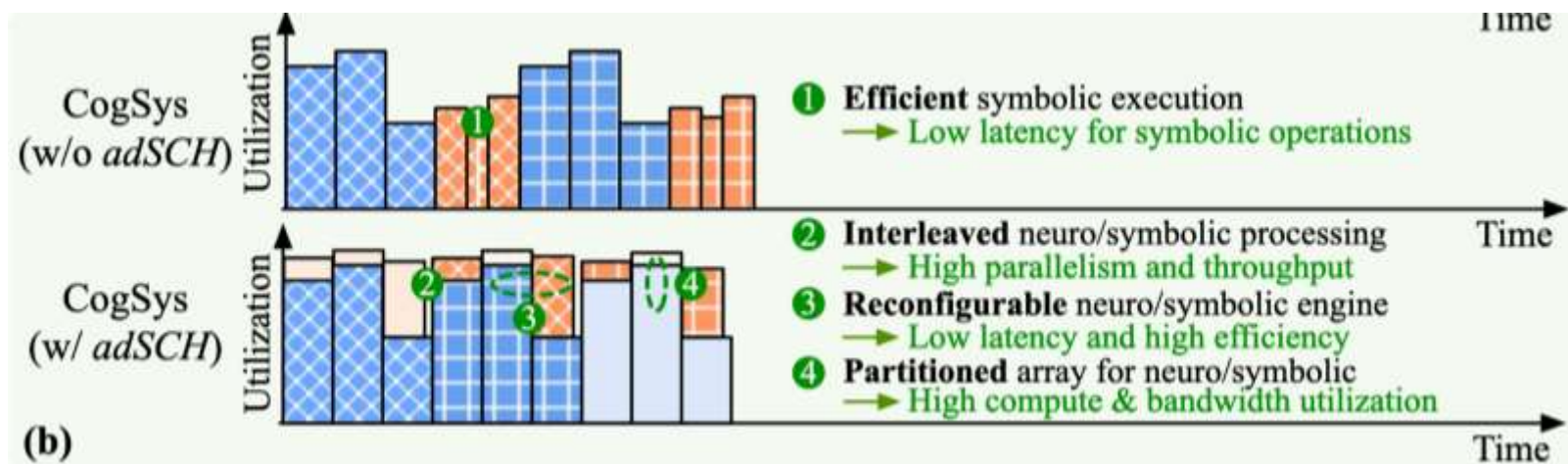


神经与符号组件的频繁交互
导致了高延迟和低吞吐量

异构的神经和符号内核导致计算
阵列利用率和ML加速器效率低下



Adaptive workload-aware scheduling (**adSCH**) strategy (on CogSys)



无 adSCH :

- 神经与符号操作不能交错执行, 阻塞严重
- 总体资源利用率低 (利用率上下波动大)

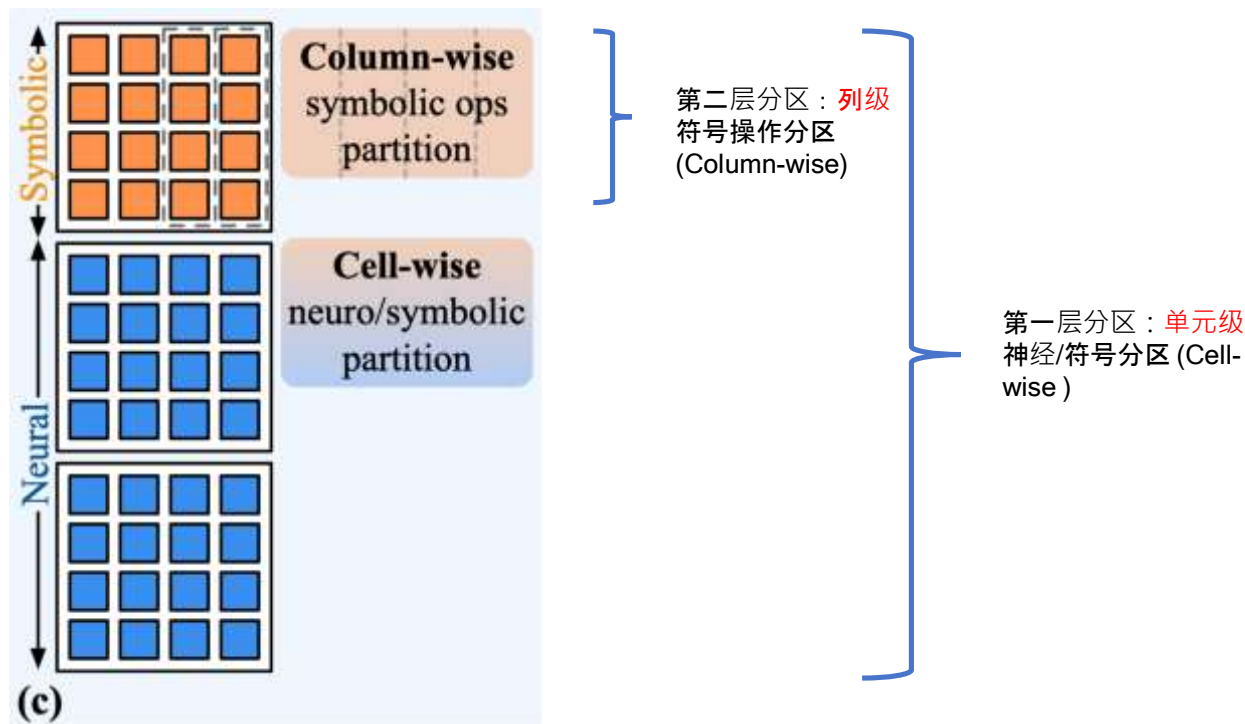
有 adSCH :

- ② 通过时间片调度、子任务拆分, 使两类任务可以并发交错执行
- ③ 系统根据任务动态调整重构阵列结构 (神经密集型 or 符号密集型)
- ④ 阵列资源被“分区”分配给不同类型任务

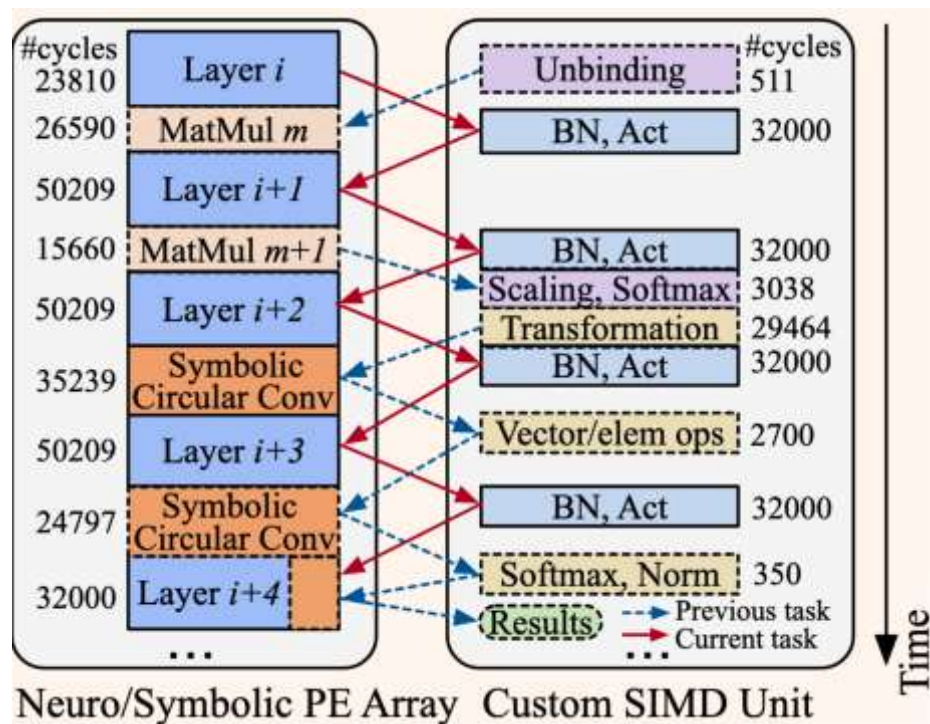
Design—Scheduling Strategy

Partitioned array for neuro/symbolic

- 为神经/符号任务分区的阵列



Example :



Contents

- Background
- Challenges
- Design
- **Evaluation**
- Thinking

Evaluation

A. Experimental Setup

➤ Datasets

spatial-temporal reasoning datasets :

RAVEN , I-RAVEN, PGM , CVR and SVRT

➤ Algorithm setup

在基于VSA的神经符号工作上评估CogSys :

NVSA、MIMONet and LVRF

➤ Baselines

TX2, Xavier NX, RTX GPU, Xeon CPU, and ML accelerators

(TPU, MTIA, Gemmini)

HW	General Purpose Processor/SoC				CogSys (Ours)	HW	ML Accelerator			CogSys (Ours)
	CPU Xeon	RTX GPU	TX2	NX			TPU-like	MTIA-like	Gemmini-like	
Power	145W	250W	15W	20W	1.48W	SRAM	4.5MB	4.5MB	4.5MB	4.5MB
						#PE	1 128×128	16 32×32	64 16×16	16 32×32

Layout of Proposed CogSys Accelerator



Accelerator Specs

Technology	28 nm	DRAM BW	700 GB/s
#Arrays	16	Frequency	800 MHz
Size of Each Array	32x32	Voltage	1 V
#SIMD PEs	512	Power	1.48 W
SRAM	4.5 MB	Area	4.0 mm ²

➤ Hardware setup

设计实现:

语言: RTL

工艺: TSMC 28纳米

工具: Synopsys (综合), Cadence (布局布线)

物理规格:

面积: 4.0 mm²

功耗: 1.48 W (平均)

频率: 0.8 GHz

Evaluation

B. Algorithm Optimization Performance

➤ Factorization accuracy

将CogSys与最先进的分解器进行比较。可见在对象提取方面，
分解准确率有轻微提升

Test	2×2 Grid	3×3 Grid	Left-Right	Up-Down	Center	O-IC	DistFour	Average
[50]	95.8%	94.7%	96.1%	95.6%	94.9%	95.3%	94.5%	95.3%
CogSys	95.7%	95.2%	96.1%	95.7%	95.3%	95.5%	94.4%	95.4%
Test	Constant	Progression	XOR	AND	OR	Arithmetic	Distribution	Average
[50]	93.3%	93.5%	93.9%	93.7%	93.5%	93.1%	92.7%	93.4%
CogSys	93.3%	93.6%	93.9%	93.6%	93.7%	93.4%	92.7%	93.5%

➤ Reasoning accuracy

在注入随机性优化和量化技术后，在保持相当的推理准确率的同时，CogSys节省了4.75倍的内存占用；
以及在台积电28纳米技术下7.71倍的面积极省和4.02倍的功耗节省

Datasets	NVSA [33]	CogSys (+Factorization & Stoch.)	CogSys (+Quant.)
RAVEN [95]	98.5%	98.7 \pm 0.3%	98.6 \pm 0.4%
I-RAVEN [36]	99.0%	99.0 \pm 0.3%	98.8 \pm 0.4%
PGM [11]	68.3%	68.6 \pm 0.8%	68.4 \pm 1.0%
#Parameters	38 MB	32 MB	8 MB

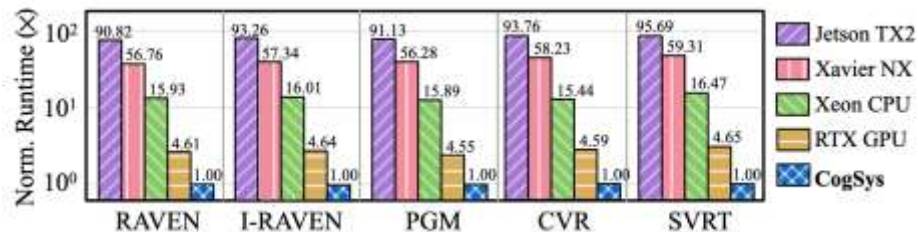
Arithmetic Precision		FP32	FP8	INT8
CogSys Accuracy (NVSA=98.5%)		98.9%	98.9%	98.7%
Reconfigurable Array 16 32×32 PEs	Area (mm ²)	28.9	9.9	3.8
	Power (mW)	4468.5	1237.8	1104.6
Custom SIMD Unit 512 PEs	Area (mm ²)	2.01	0.28	0.21
	Power (mW)	297.0	64.8	80.4
Reconfig. Array Area Overhead vs. SA		<1%	4.8%	12.1%

Evaluation

C. CogSys Accelerator Performance

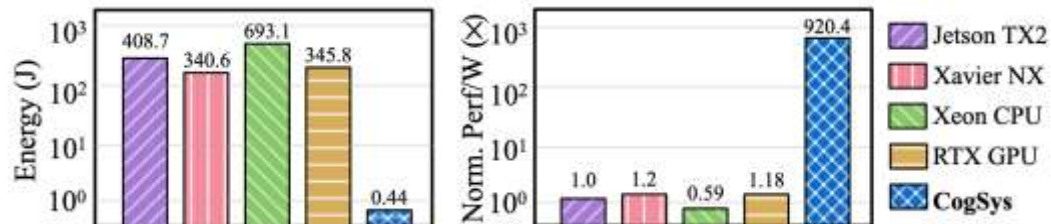
➤ *Performance improvement*

在五个不同难度级别的推理任务上进行基准测试，可见CogSys在所有数据集上都有极大的加速效果（均<0.3s）



➤ *Energy efficiency improvement*

对CogSys加速器的能耗和效率进行了基准测试，可见相比较于传统硬件在能耗方面有几个数量级的提升

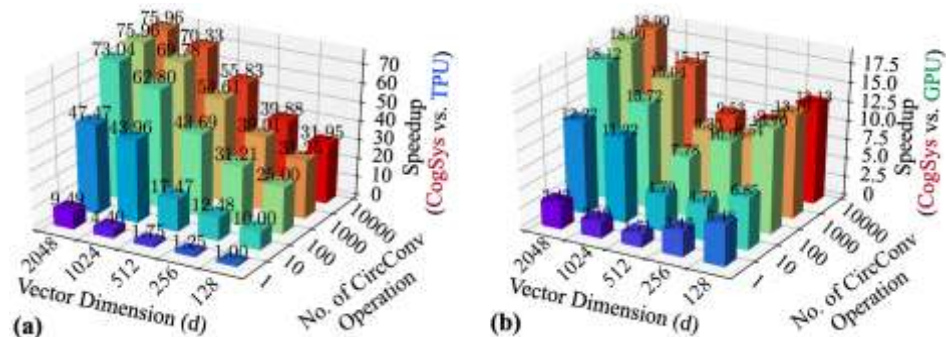


Evaluation

C. CogSys Accelerator Performance

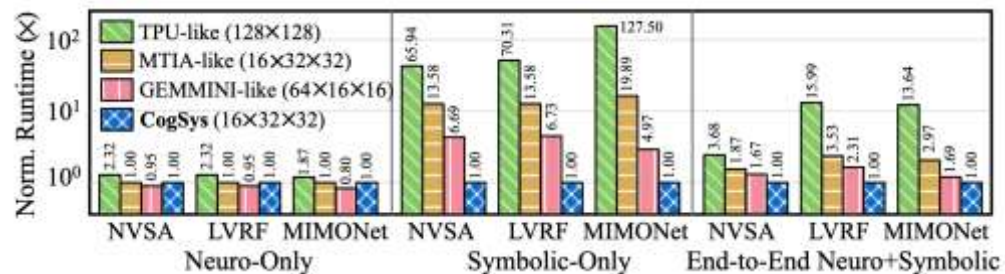
➤ Comparison with TPU/GPU

在不同的向量维度和操作数量下，对GPU和TPU上的上的CircConv进行基准测试，可见CogSys在所有维度都表现出优异的加速比提升



➤ Comparison with ML accelerators

与当前的ML加速器相比，CogSys在神经网络操作上有相似的性能，同时在符号操作上有巨大提升

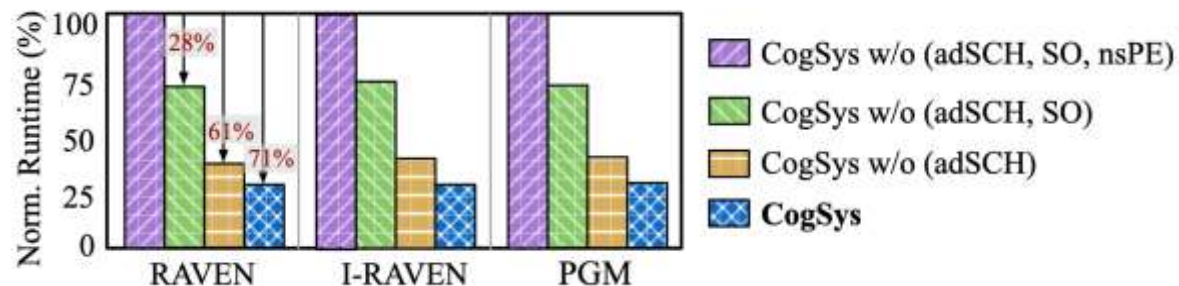


Evaluation

C. CogSys Accelerator Performance

➤ *Ablation study on the proposed hardware techniques*

对asSCH策略、可扩展架构SO和可重构PE进行消融实验：adSCH策略可以将运行时缩短28%，可扩展阵列SO和可重构PE进一步优化总运行时间的减少至61%和71%



➤ *Ablation study of necessity of co-design*

用于验证算法与硬件的协同效果。对算法和硬件进行消融实验：仅使用我们提出的CogSys算法优化，运行时降低到NVSA算法的89.5%；当同时使用提出的CogSys算法优化和我们的加速器时，运行时可大幅度降低到1.76%

Neurosymbolic Cognitive Solution Algorithm @ Hardware	Normalized Runtime (%) on				
	RAVEN [95]	I-RAVEN [36]	PGM [11]	CVR [94]	SVRT [20]
NVSA [33] @ Xavier NX	100	100	100	100	100
CogSys Algorithm @ Xavier NX	89.5%	88.9%	90.7%	87.6%	88.4%
CogSys Algorithm @ CogSys Accelerator	1.76%	1.74%	1.78%	1.72%	1.69%

Contents

- Background
- Challenges
- Design
- Evaluation
- **Thinking**

Thinking

这篇 paper 的 idea 能不能应用在自己的工作上面？

- 本文提出了一种“算法-硬件协同设计”的神经符号加速器 **CogSys**。该设计通过可重构的处理单元(*PE*)和分区的计算阵列(*Partitioned Array*)，实现了对异构任务(神经网络+符号逻辑)的高效处理。这种设计方法可以用于协助在边缘设备上实现复杂认知模型的部署。
- 另外，本文提出了 **adSCH** 的自适应调度策略。该策略通过交错执行，智能地将不同计算任务调度到合适的硬件单元上(*PE*阵列或*SIMD*单元)，从而在物理层面最大化硬件利用率和系统吞吐量。这个调度思想可以被泛化，用于优化异构硬件上的推理性能。

这篇 paper 的思想能否泛化？

- 对于包含异构计算任务的系统，我觉得都可以引入 **CogSys** 的可重构或分区(思想，通过让硬件自适应地匹配计算任务，而非让任务去被动适应硬件，从而达到更高的系统效率。



東南大學
SOUTHEAST UNIVERSITY

Q & A

汇报人：宋青阳

2025年7月25日