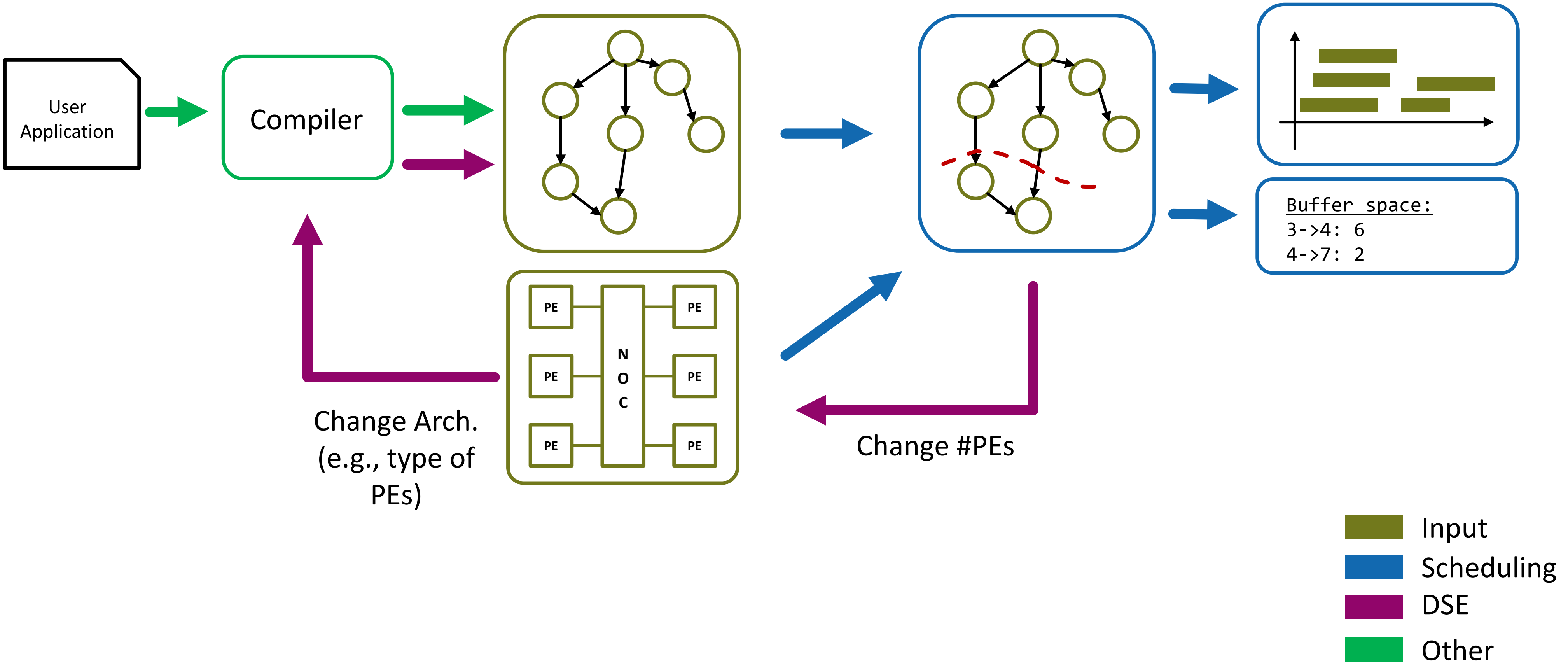
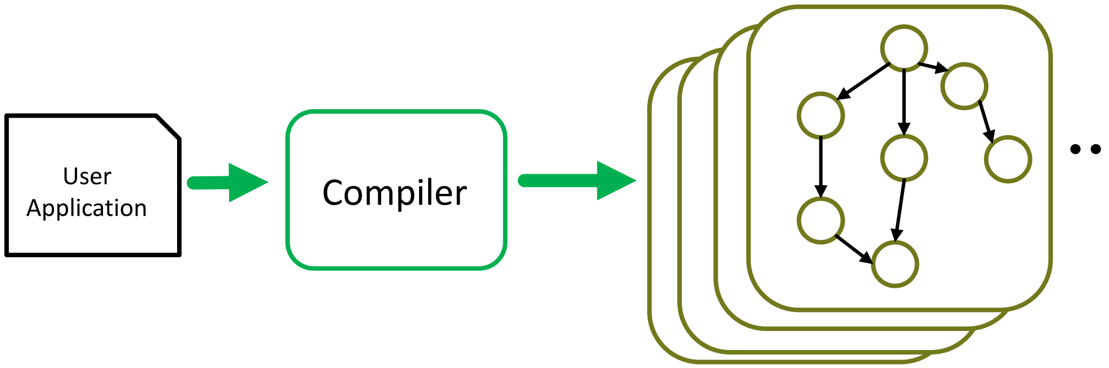
ASA: Moving beyond Scheduling

The need for a “compiler”



The need for a "compiler"

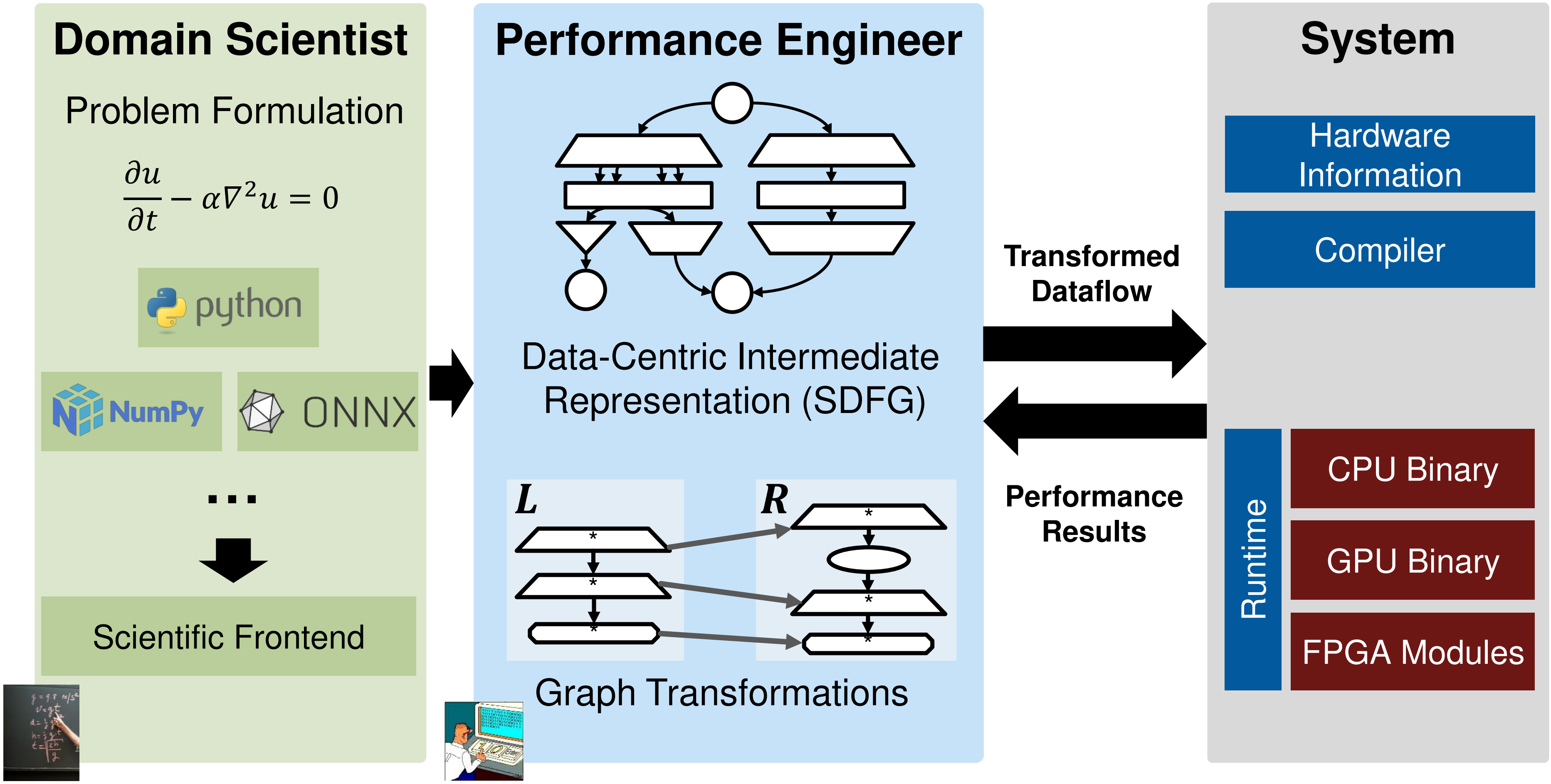


For the compiler, we want:

- Generate all the canonical task graphs
- Evaluate each of them: scheduling or other approaches?

🟫 Input
🟦 Scheduling
🟪 DSE
🟩 Other

DaCe Overview



Domain Scientist

Problem Formulation

$$\frac{\partial u}{\partial t} - \alpha \nabla^2 u = 0$$

python

NumPy

ONNX

...

Scientific Frontend

Performance Engineer

Data-Centric Intermediate Representation (SDFG)

Graph Transformations

Transformed Dataflow

Performance Results

System

Hardware Information

Compiler

Runtime

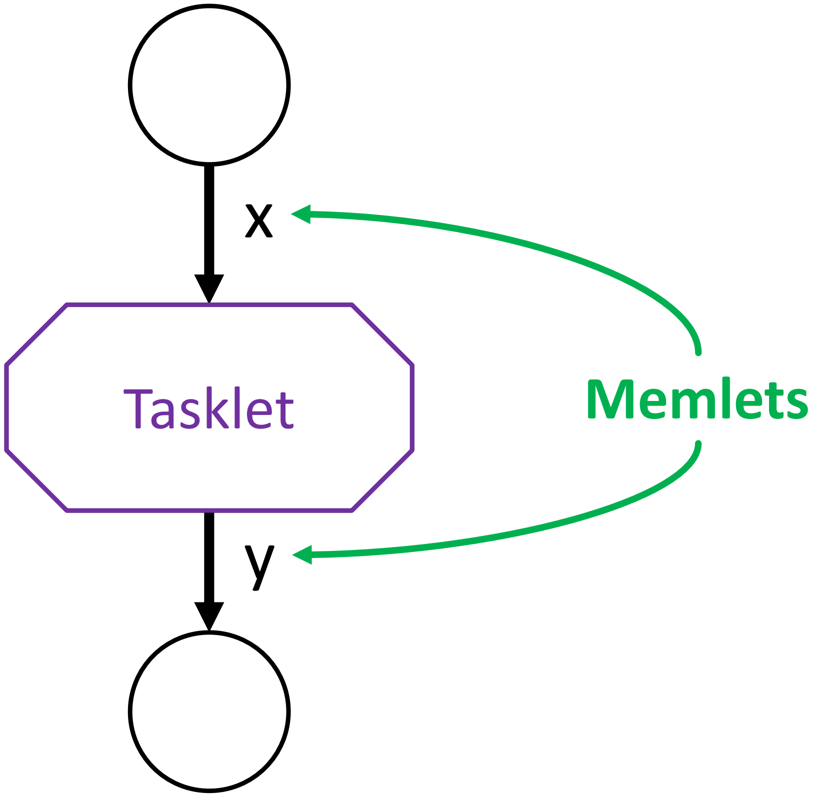
CPU Binary

GPU Binary

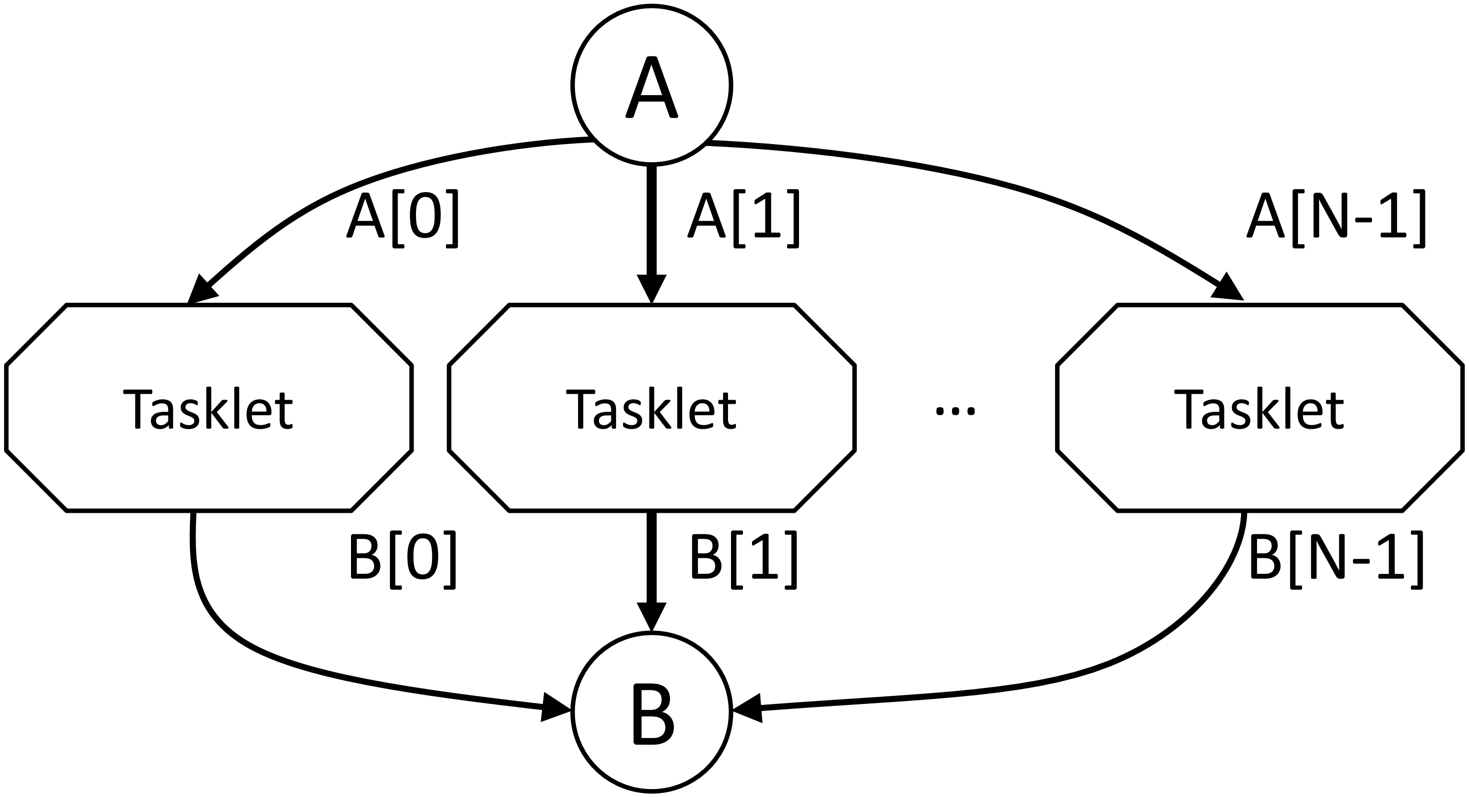
FPGA Modules

Ben-nun T., de Fine Licht J., Ziogas A.N, Schneider T. and Hoefler T. Stateful Dataflow Multigraphs: A Data-Centric Model for Performance Portability on Heterogeneous Architectures. In SC 2019

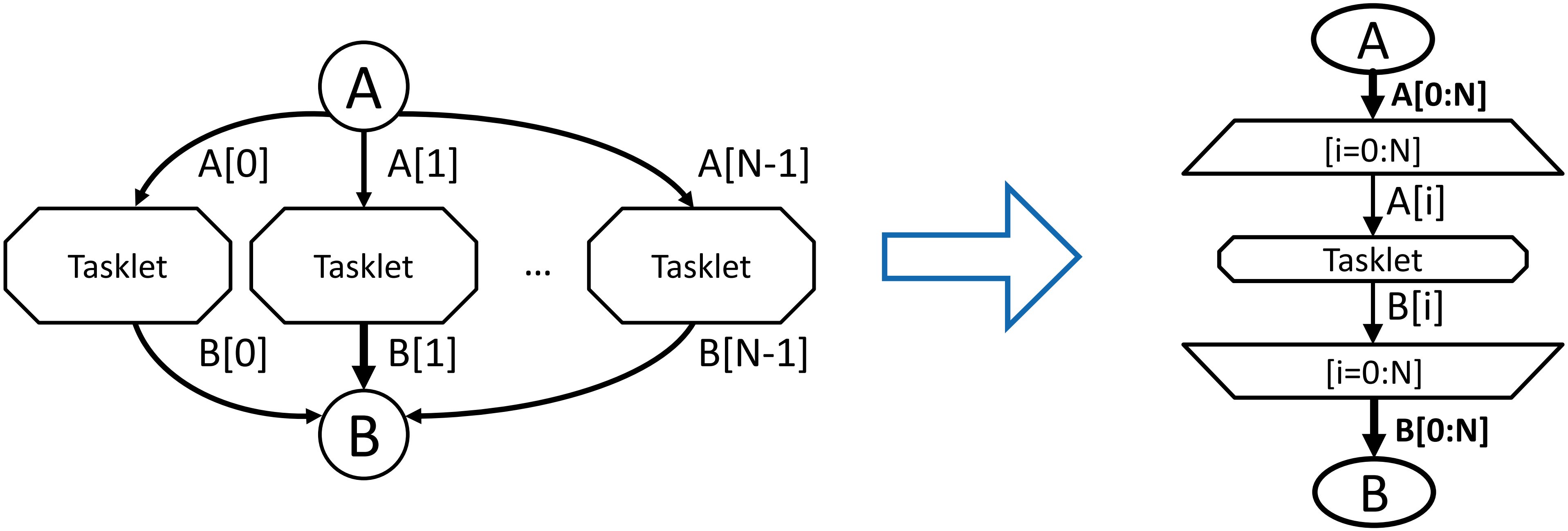
Dataflow Programming in DaCe



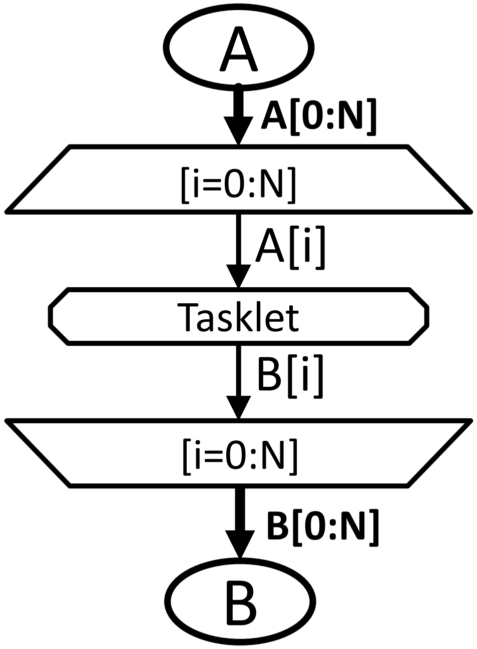
Parallel Dataflow Programming

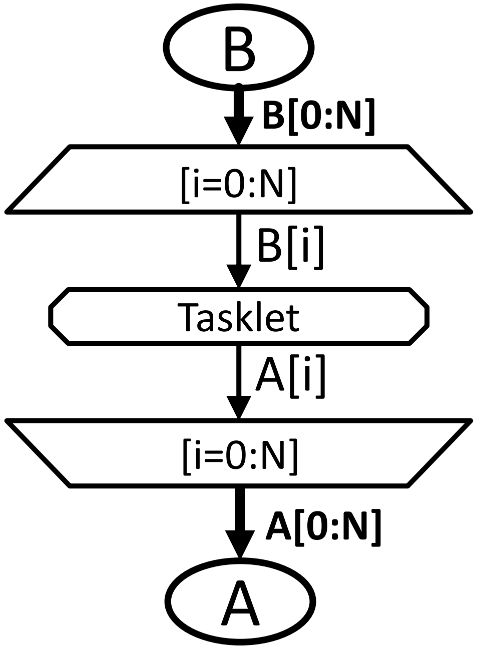


Parallel Dataflow Programming

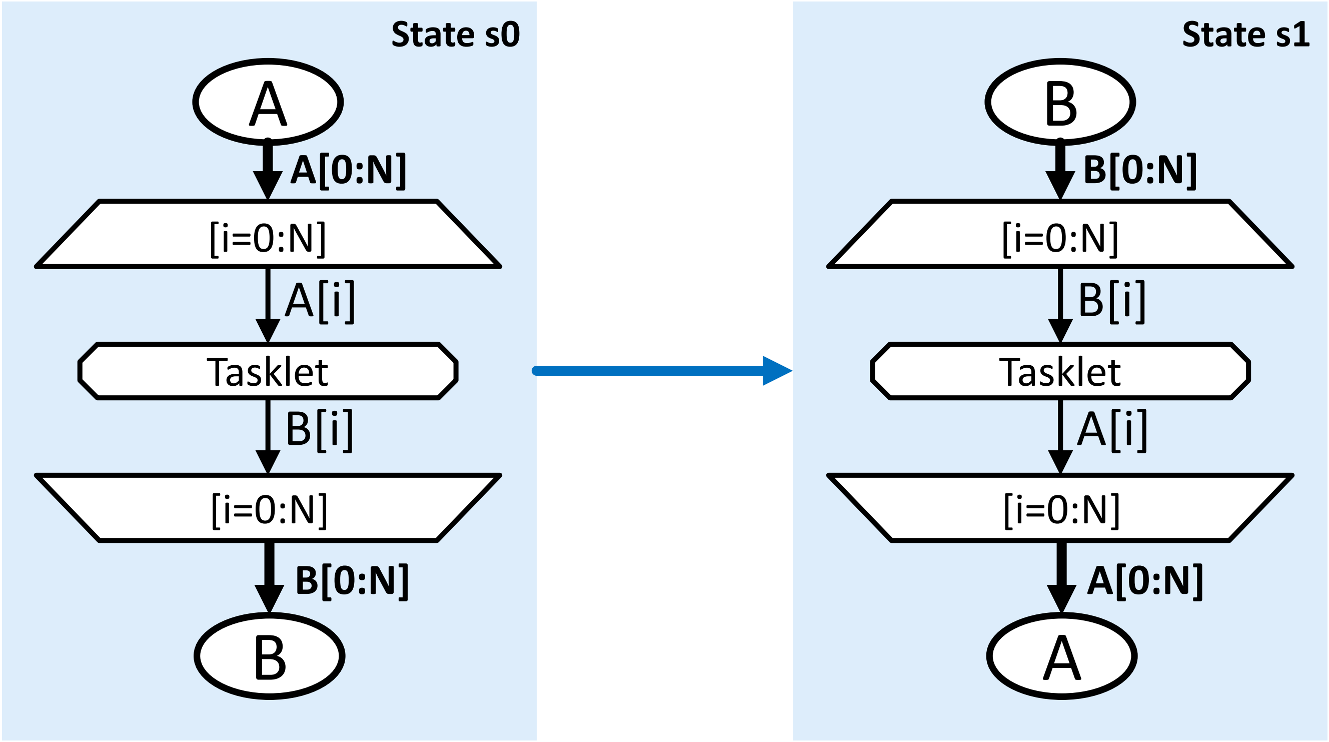


Stateful Dataflow Parallel Programming in DaCe

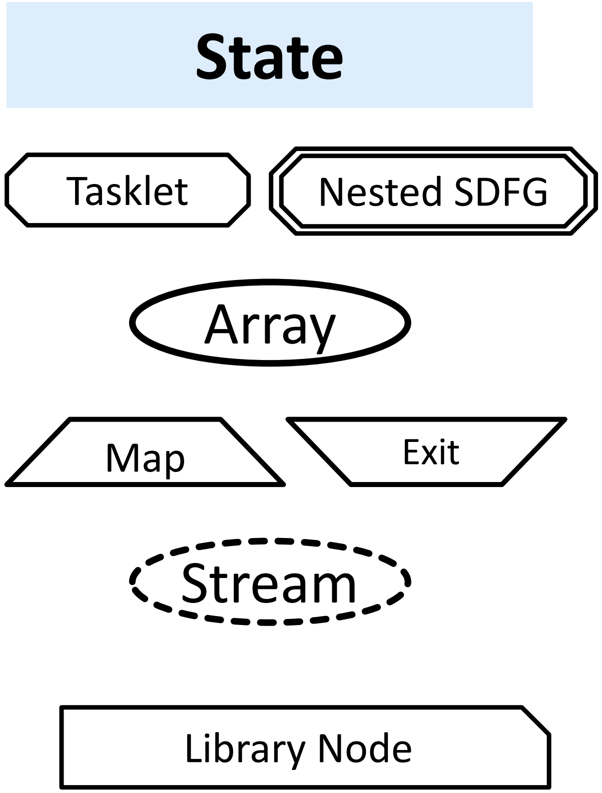




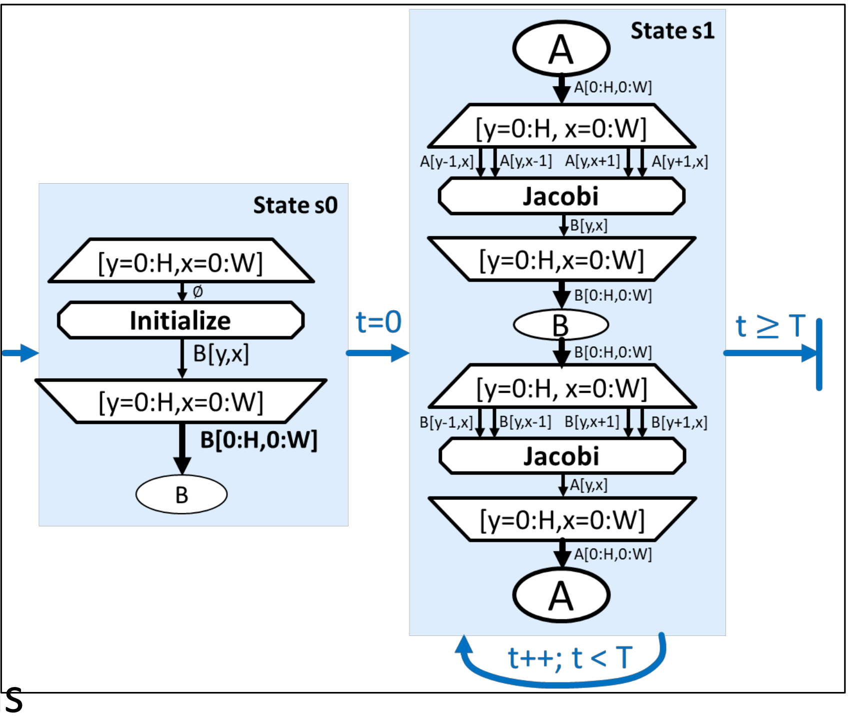
Stateful Dataflow Parallel Programming in DaCe



Meet the Nodes

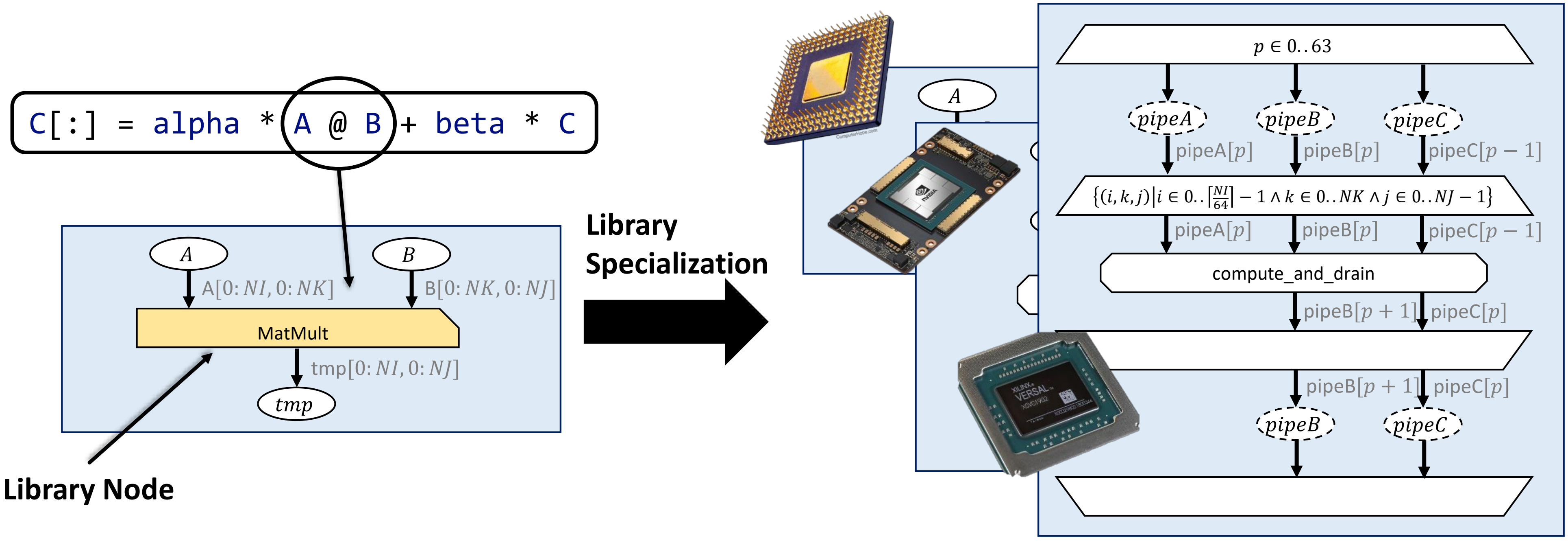


State	State machine element
Tasklet Nested SDFG	Fine-grained computatior
Array	N-dimensional data conta
Map Exit	Parametric graph abstract
Stream	Streaming data container
Library Node	Represent call to functions



Library Nodes

```
C[:] = alpha * A @ B + beta * C
```


A
B
A[0: NI, 0: NK]
B[0: NK, 0: NJ]
MatMult
tmp[0: NI, 0: NJ]
tmp
Library Node
Library Specialization
A
p ∈ 0..63
pipeA
pipeB
pipeC
pipeA[p]
pipeB[p]
pipeC[p − 1]
{(i, k, j)|i ∈ 0..⌈NI/64⌉ − 1 ∧ k ∈ 0..NK ∧ j ∈ 0..NJ − 1}
pipeA[p]
pipeB[p]
pipeC[p − 1]
compute_and_drain
pipeB[p + 1]
pipeC[p]
pipeB[p + 1]
pipeC[p]
pipeB
pipeC