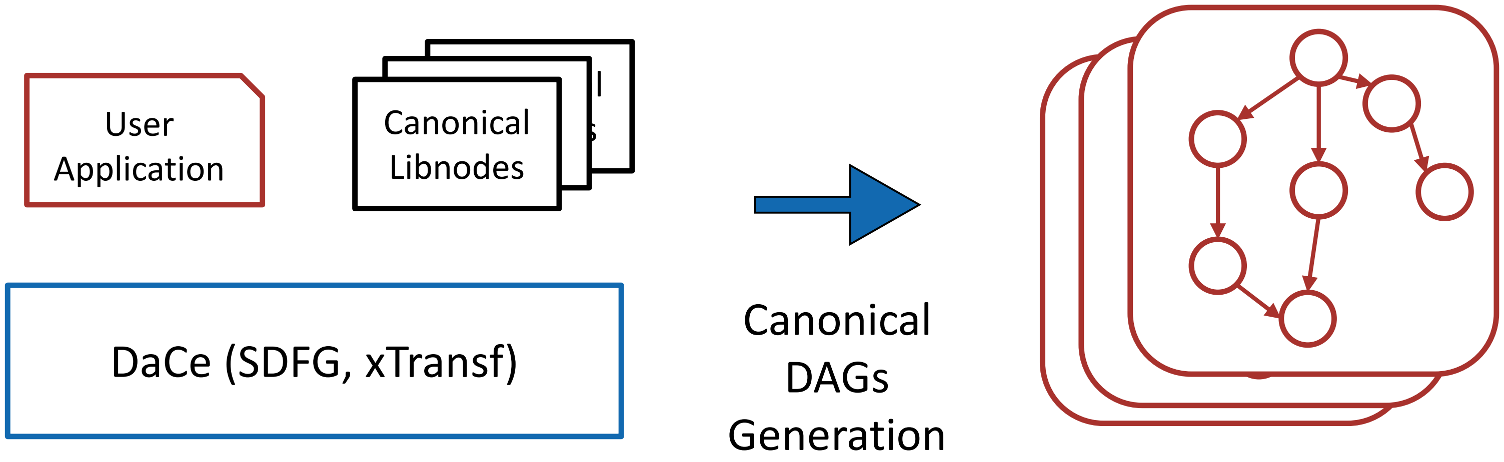
ASA: Enabling DSE

Application Space Exploration

We want to use DaCe (IR, LibNode, Transformations) to enable all of this (“data-centric and compiler approach”)



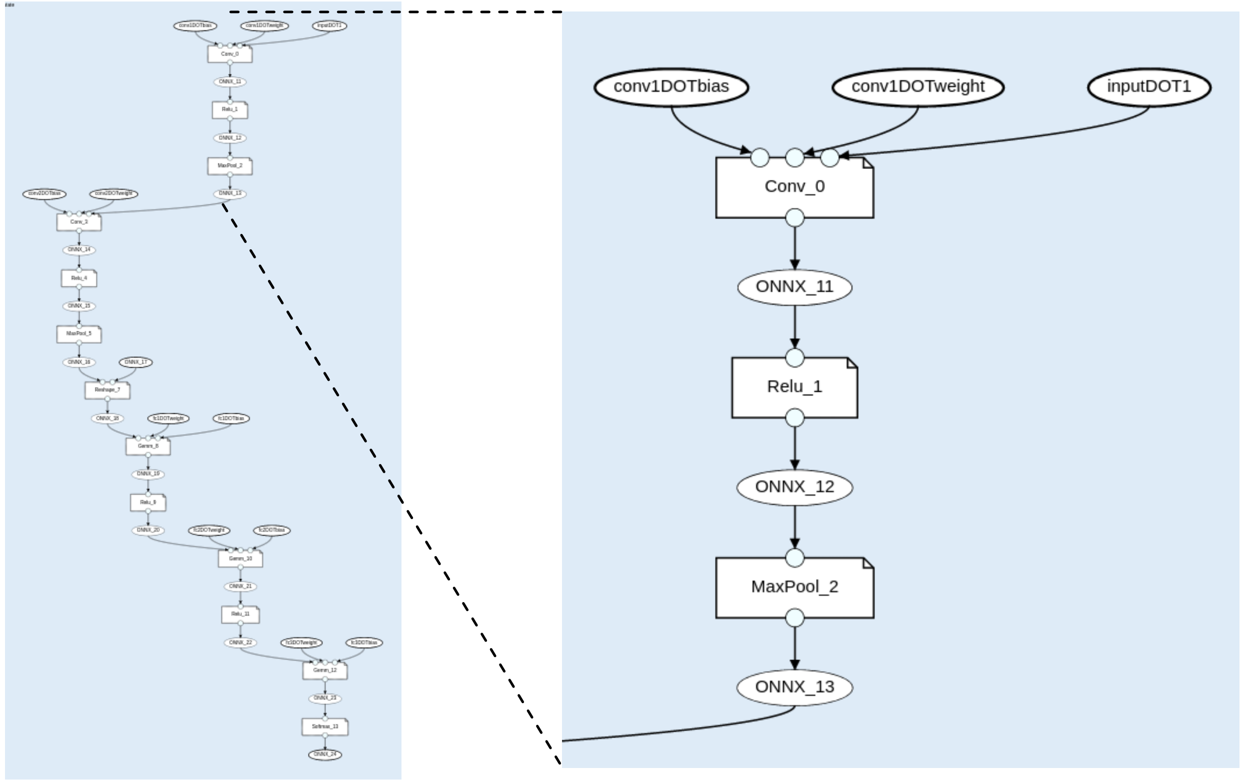
ML Workloads

For these we leverage DaCe as frontend

```
class LeNet(nn.Module):
  def __init__(self):
    super(LeNet, self).__init__()
    self.conv1 = nn.Conv2d(1, 6, 5)
    self.conv2 = nn.Conv2d(6, 16, 5)
    self.fc1 = nn.Linear(256, 120)
    self.fc2 = nn.Linear(120, 84)
    self.fc3 = nn.Linear(84, 10)

  def forward(self, x):
     x = F.max_pool2d(F.relu(self.conv1(x)), 2)
     x = F.max_pool2d(F.relu(self.conv2(x)), 2)
     x = x.view(-1, 256) x = F.relu(self.fc1(x))
     x = F.relu(self.fc2(x))
     x = self.fc3(x)
     x = F.softmax(x, dim=1)
    return x
```

DaceML

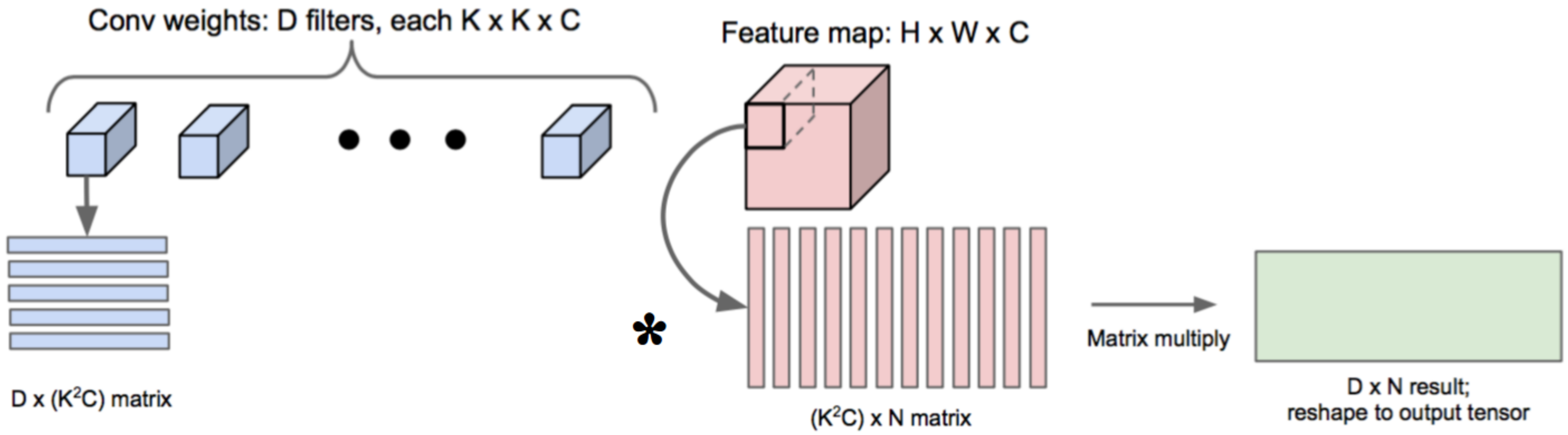


At this point, we need to create canonical expansions for the various ONNX operators

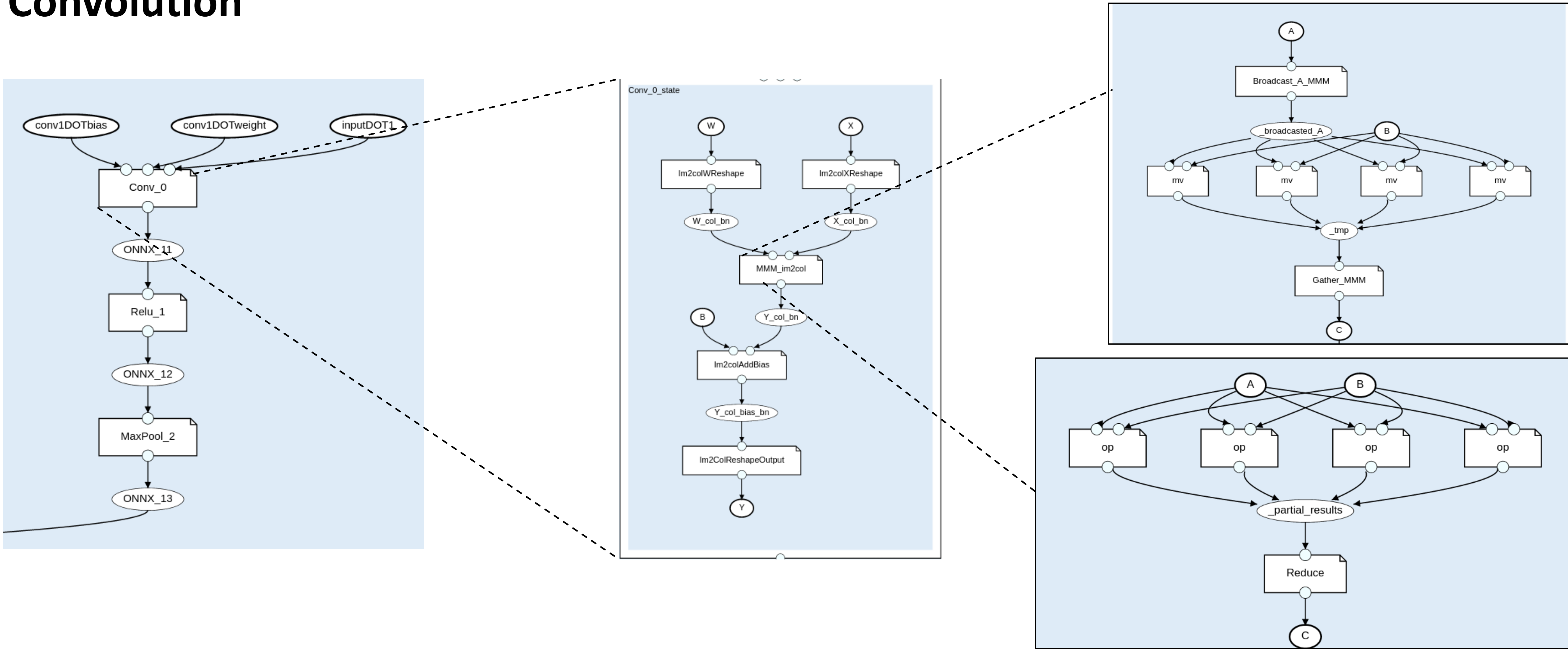
ML Workloads

For certain operators this is straightforward: e.g., all element-wise operations, such as **Relu**, **Add**, **Sub**, …

For others is a bit more complicated. Think about **Convolution**, and suppose that we want to use im2col approach

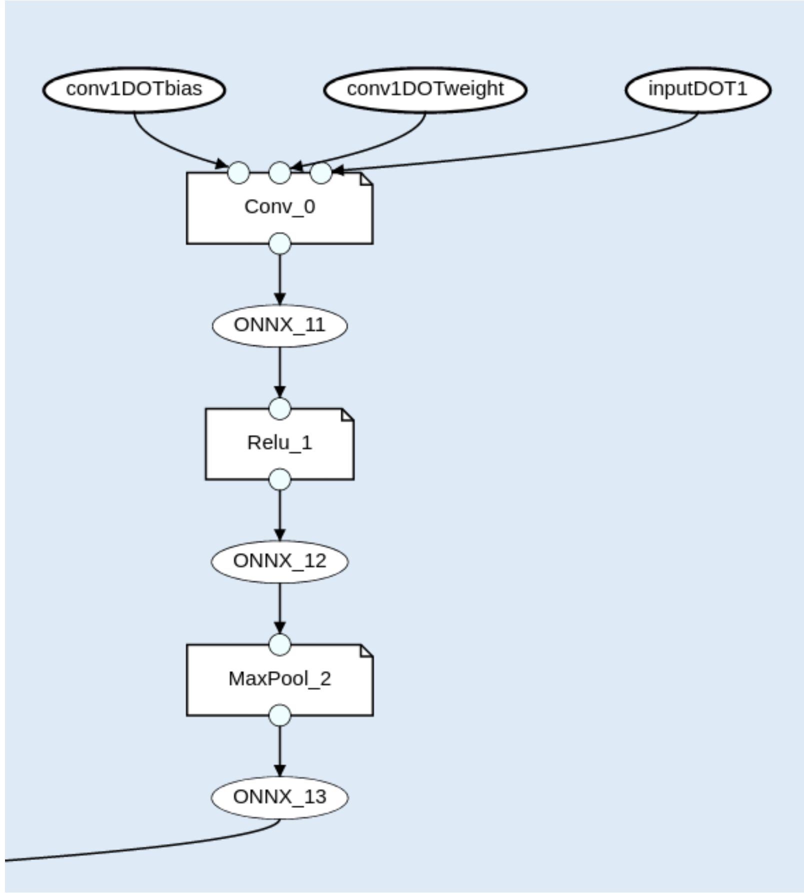


Convolution



Progressive lowering allows us to build the schedulable/analyzable Canonical DAG

Application Space Exploration



We can compare several implementations for CONV (MMM)

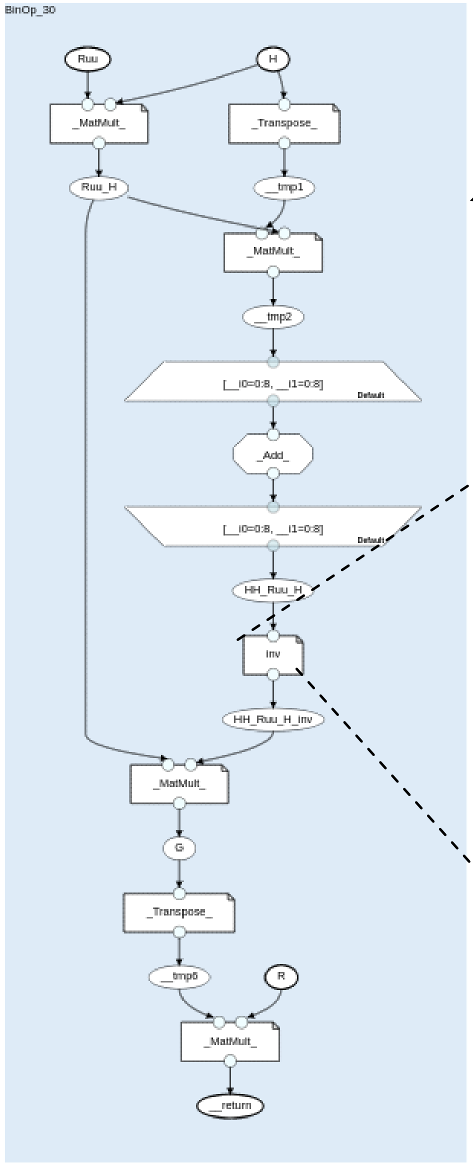
		16 PEs	128 PEs
DAG #	I/Os	Makespan	Makespan
1 (MV)	129K	145.7K	35.1K
2 (OP)	222K	194.1 K	107.7K
3 (LMV)	150K	35.7K	35.7K

Convolution with 6, 5x5 filters, over 28x28 input feature

The pair (implementation, #PEs) has impact on the makespan

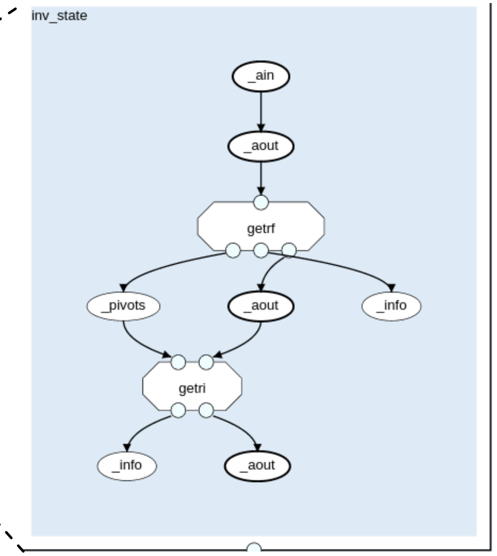
Having more convolutions with different sizes, there will be no a single winner

PUSCH-MIMO




```python
import numpy as np

@dace.program
def mimo(Ruu: dace.float32[64, 64], H: dace.float32[64, 8],
            R: dace.float32[64, 1]):
    Ruu_H = Ruu @ H
    HH_Ruu_H = np.transpose(H) @ Ruu_H + 1
    HH_Ruu_H_inv = np.linalg.inv(HH_Ruu_H)
    G = Ruu_H @ HH_Ruu_H_inv
    S = np.transpose(G) @ R
    return S
```



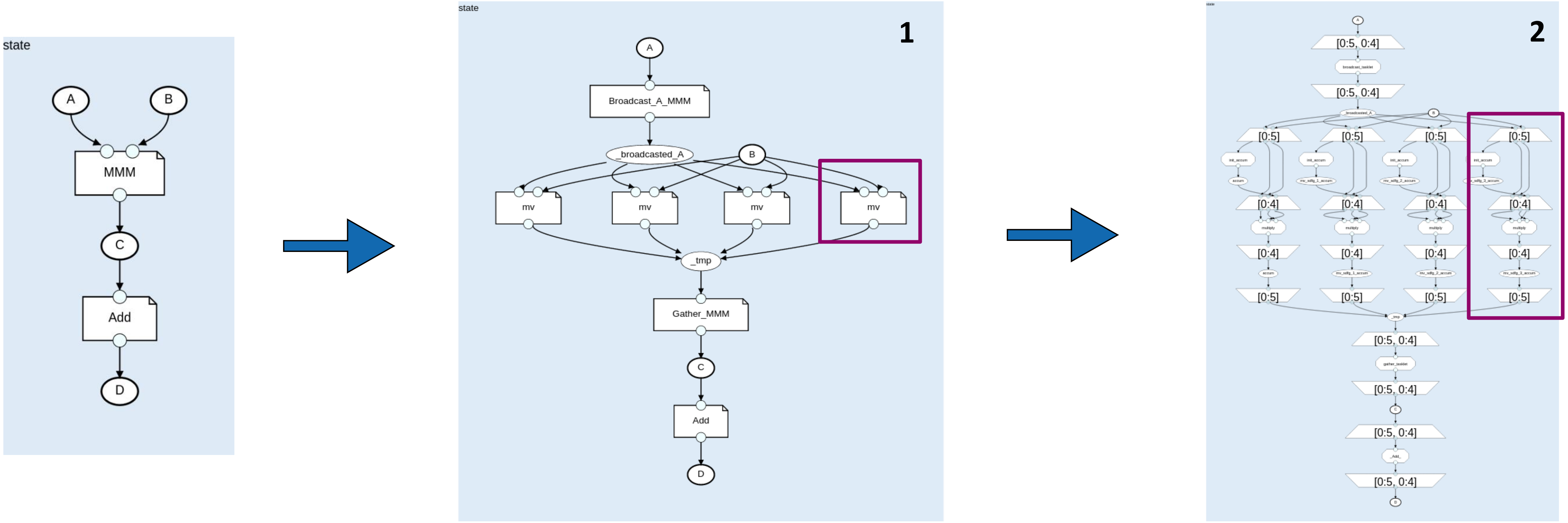
Attempt at using Lapack calls for fast evaluation (this is using LU).

Needs work on the DaCe side.

Once fixed we can perform Appl. Space Exploration as well

Q: how I can generate synthetic, but realistic, input data?

Task-Granularity

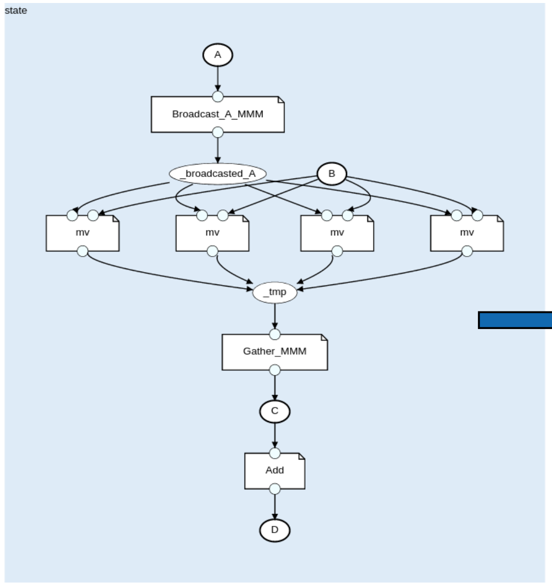


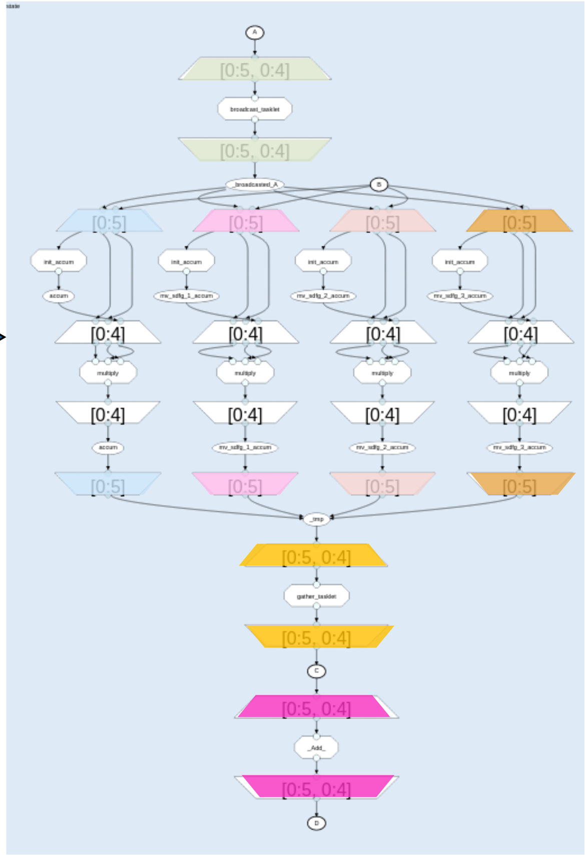
We need to "pull out" the Canonical DAG from one of these two representations

- 1 is more straightforward, but we will need anyway the fully expanded SDFG to analyze data movements
- 2 more rich, but we need to track down node-task association (no such mechanism in DaCe currently)

Task Granularity

Idea: we can reason on the fully lowered SDFG and identify as task top-level Map Scopes





This opens to the possibility of applying data-centric optimization, for example Fusion