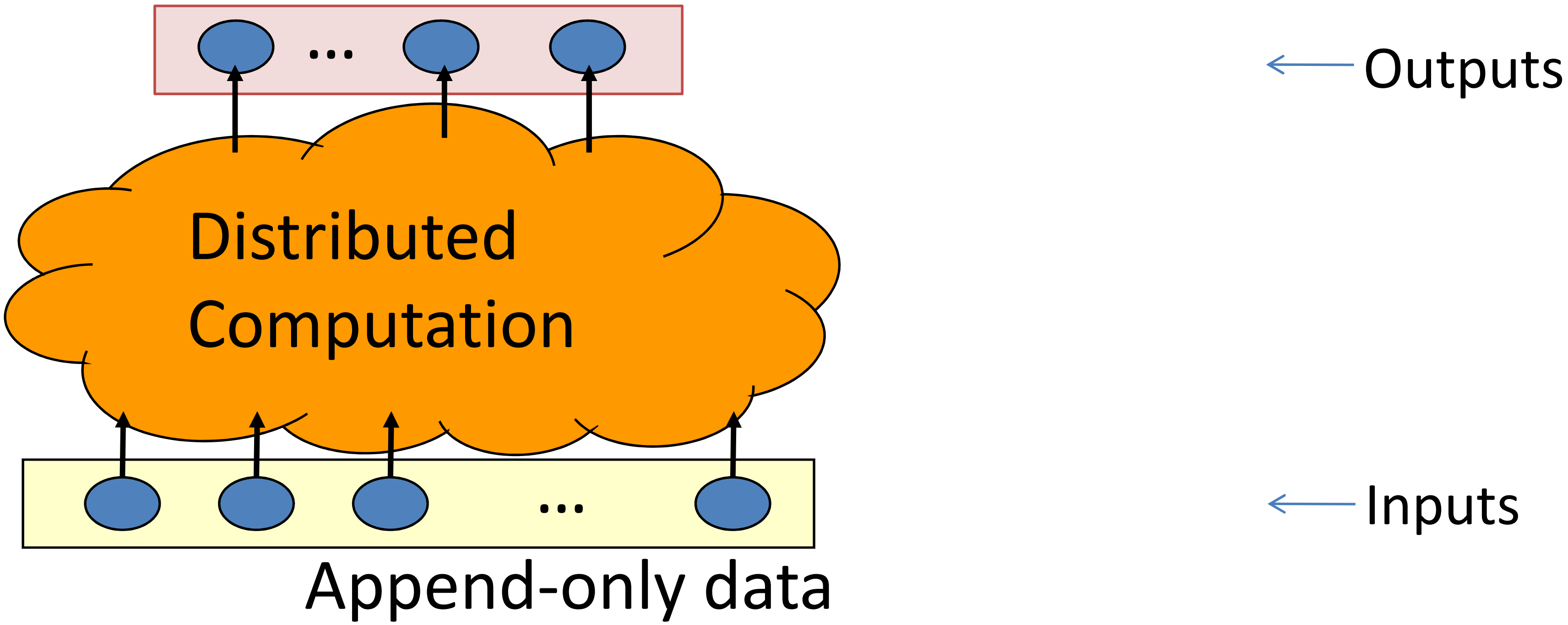
DryadInc: Reusing work in large-scale computations

Lucian Popa[*+], *Mihai Budiu*[+],
Yuan Yu[+], *Michael Isard*[+]

[+] Microsoft Research Silicon Valley
[*] UC Berkeley

Problem Statement



Outputs

…

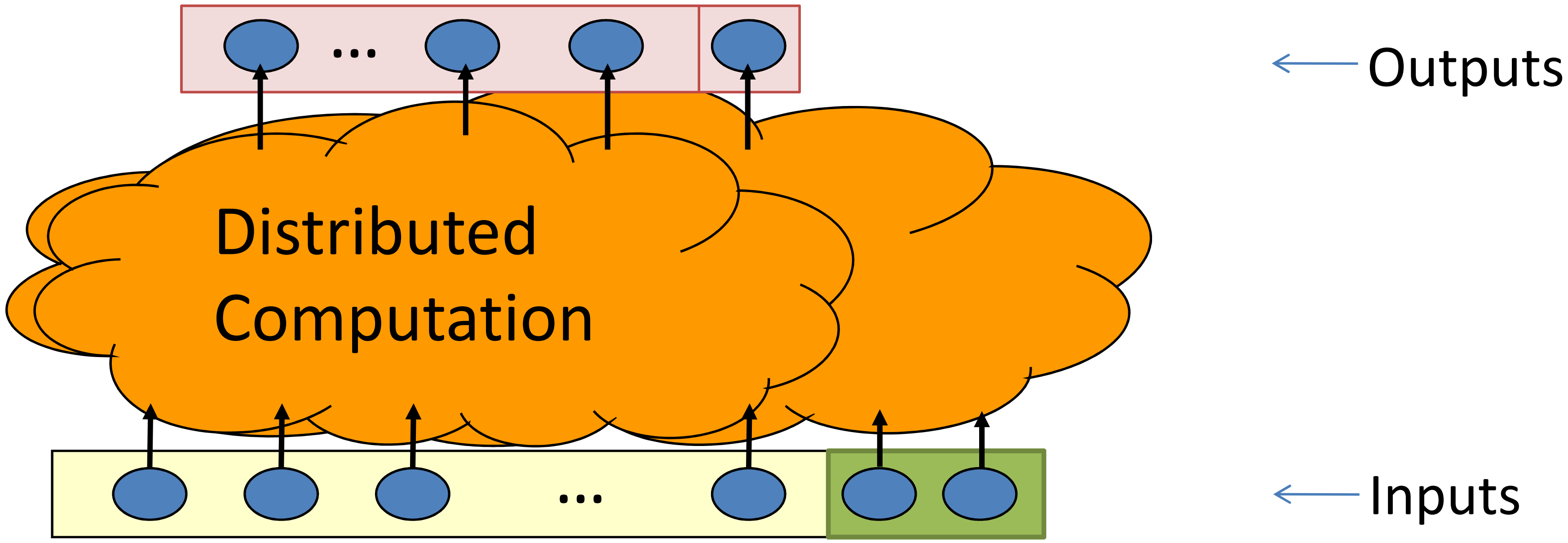
Distributed Computation

…

Inputs

Append-only data

Problem Statement



…

Distributed Computation

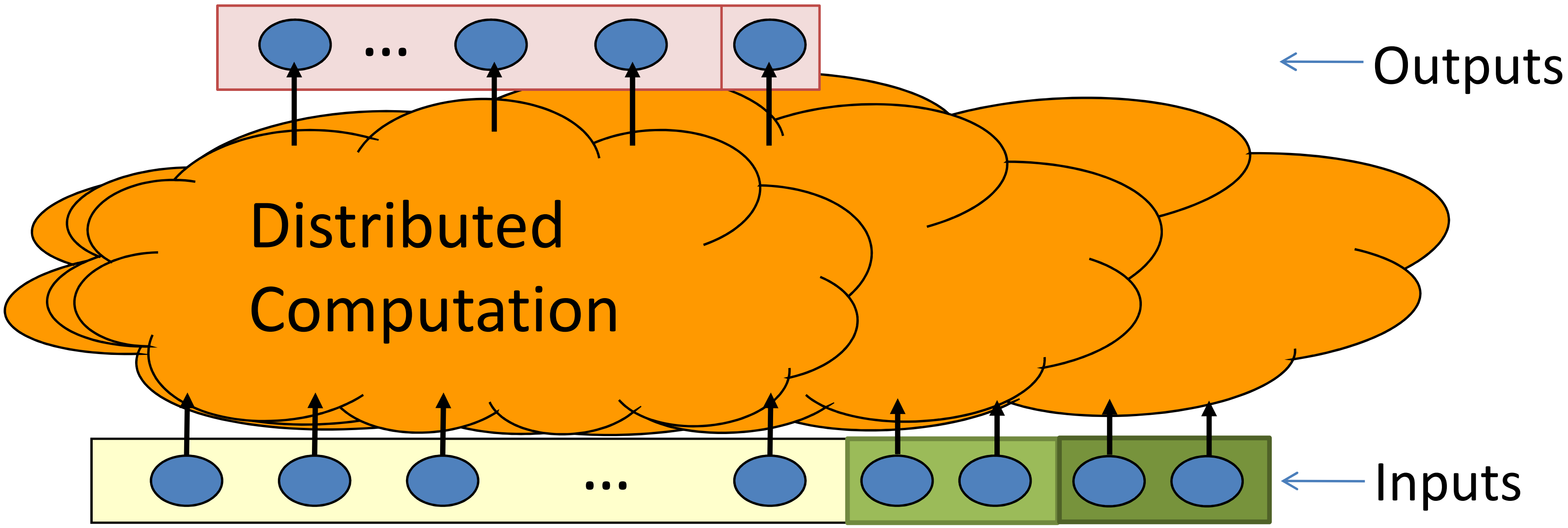
…

Outputs

Inputs

Append-only data

Problem Statement



Goal: *Reuse* (part of) prior computations to:

- Speed up the current job
- Increase cluster throughput
- Reduce energy and costs

Propose Two Approaches

1. IDE

Reuse *IDEntical computations* from the past
(like `make` or memoization)

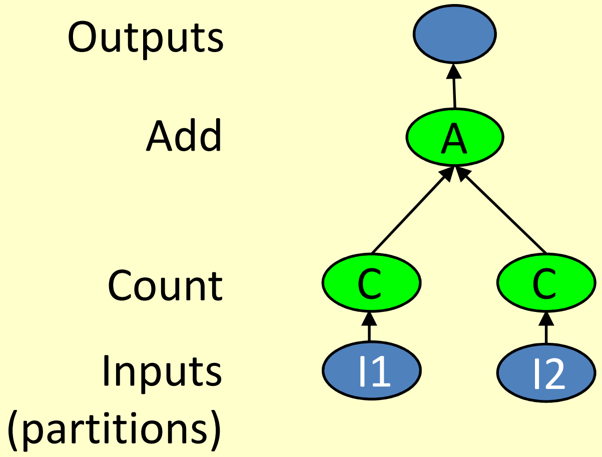
2. MER

Do only *incremental computation* on the new data
and *MERge* results with the previous ones
(like `patch`)

Context

- Implemented for **Dryad**
  - Dryad Job = Computational DAG
    - **Vertex:** arbitrary computation + inputs/outputs
    - **Edge:** data flows

Simple Example:
Record Count



Outputs

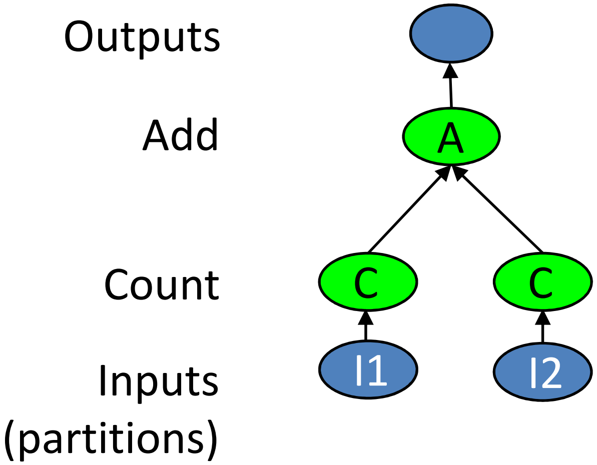
Add

Count

Inputs
(partitions)

IDE – IDEntical Computation

Record Count



Outputs

Add

Count

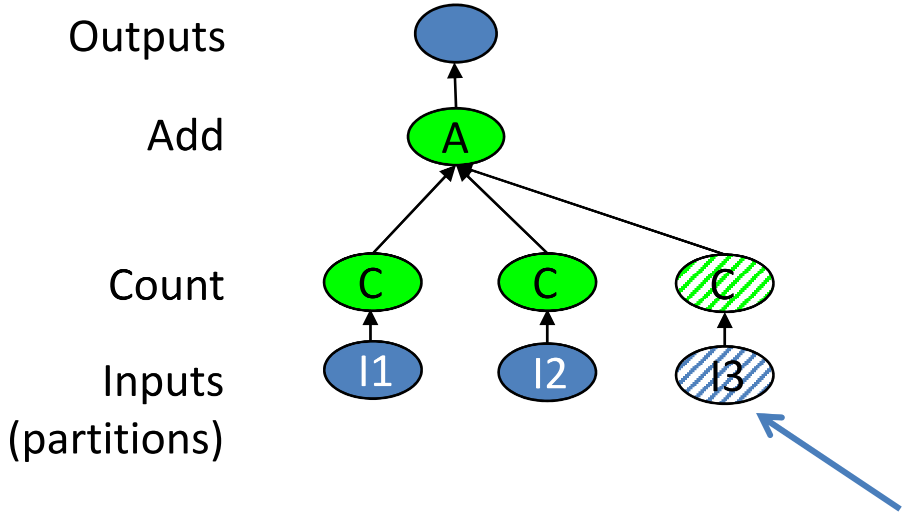
Inputs
(partitions)

First execution
DAG

IDE – IDEntical Computation

Record Count

Second execution DAG



Outputs

Add

Count

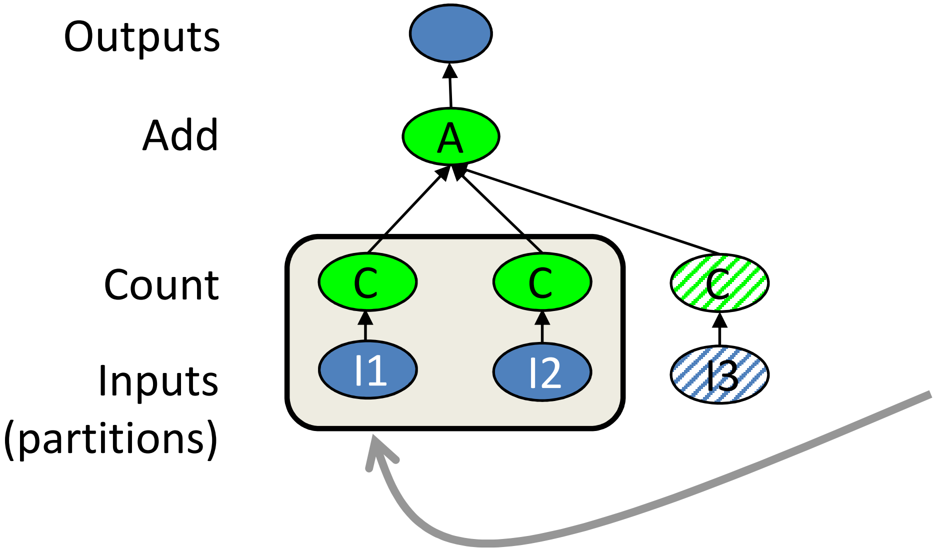
Inputs (partitions)

New Input

IDE – IDEntical Computation

Record Count

Second execution DAG

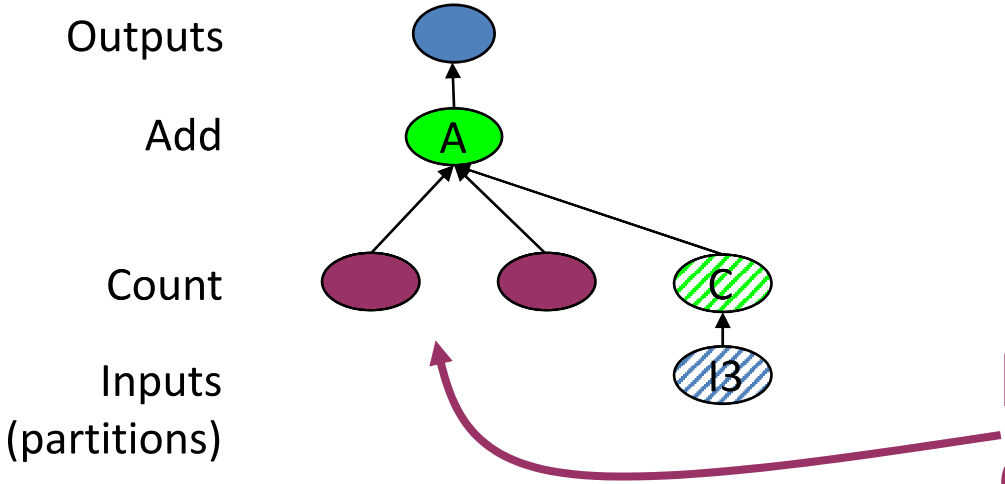


Identical subDAG

IDE – IDEntical Computation

Replace identical computational subDAG with edge data cached from previous execution

IDE Modified DAG

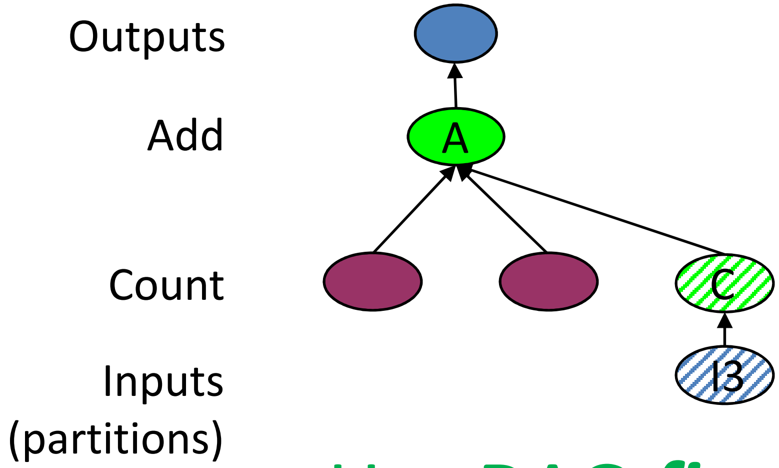


Replaced with Cached Data

IDE – IDEntical Computation

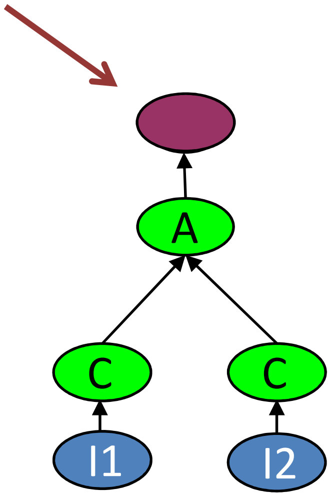
Replace identical computational subDAG with edge data cached from previous execution

IDE Modified DAG



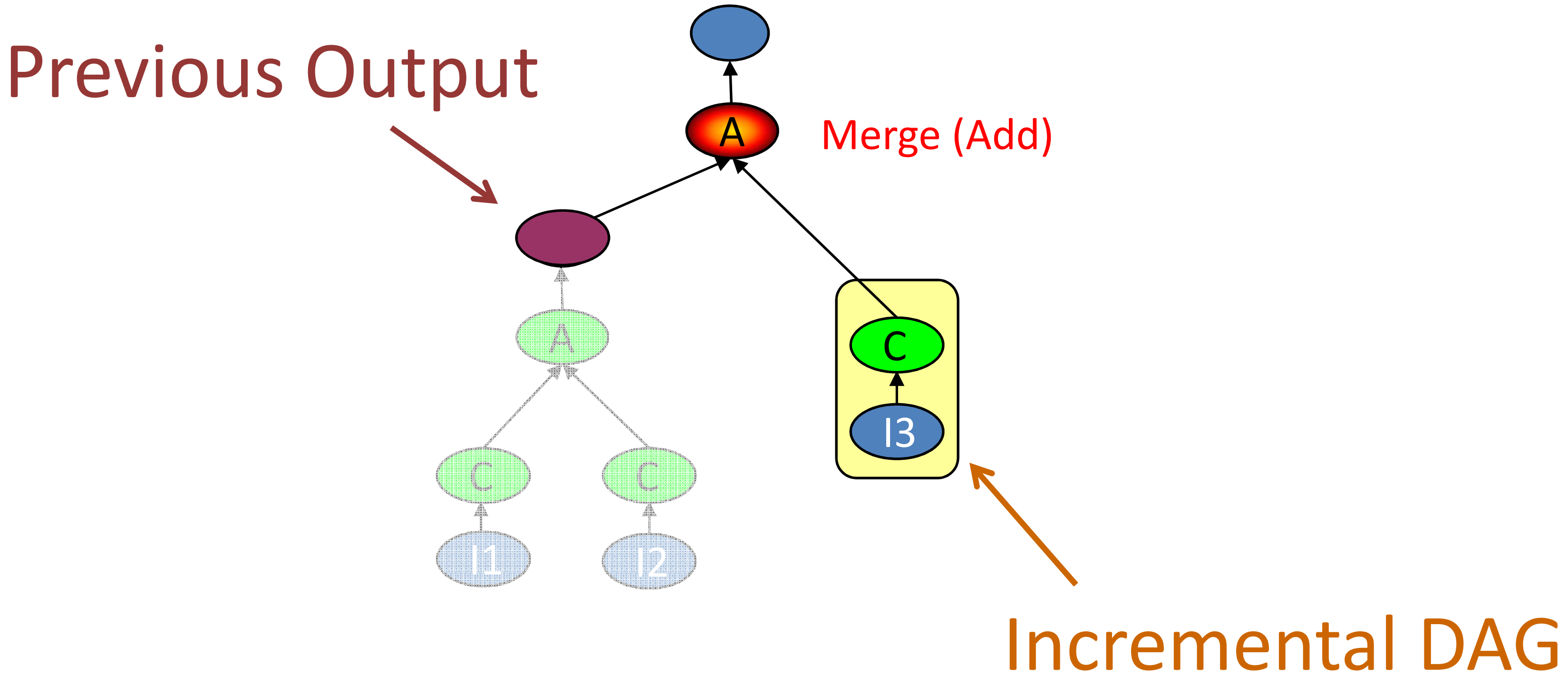
Use DAG ***fingerprints*** to determine if computations are identical

Semantic Knowledge Can Help

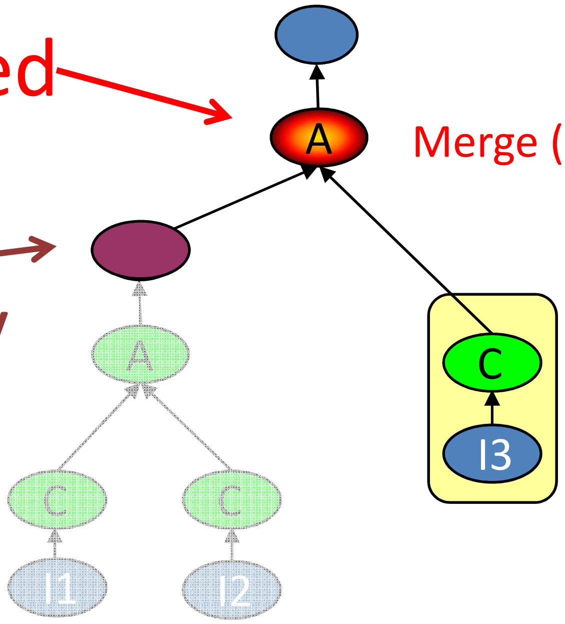


Reuse Output

Semantic Knowledge Can Help



MER – MERgeable Computation



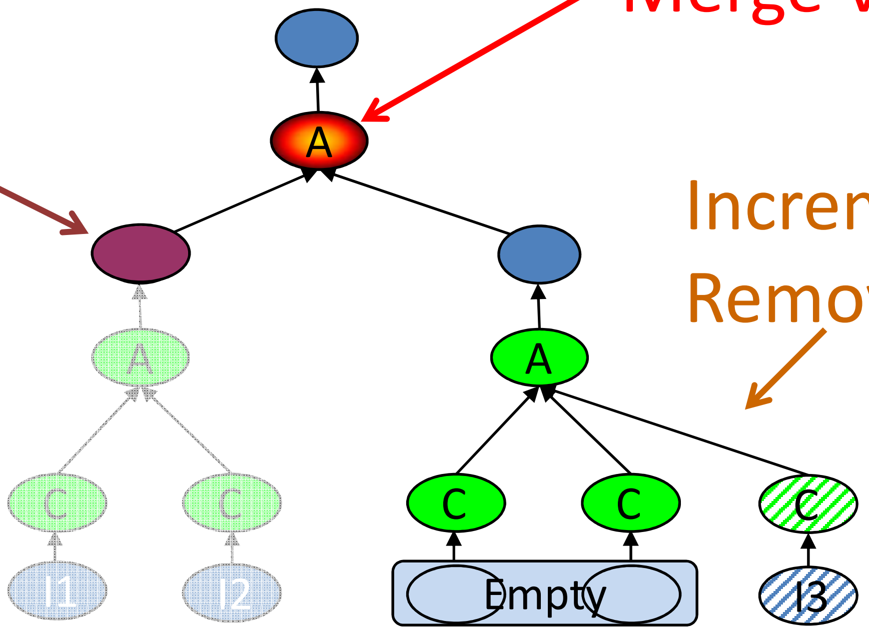
User-specified

Merge (Add)

Automatically Inferred

Automatically Built

MER – MERgeable Computation



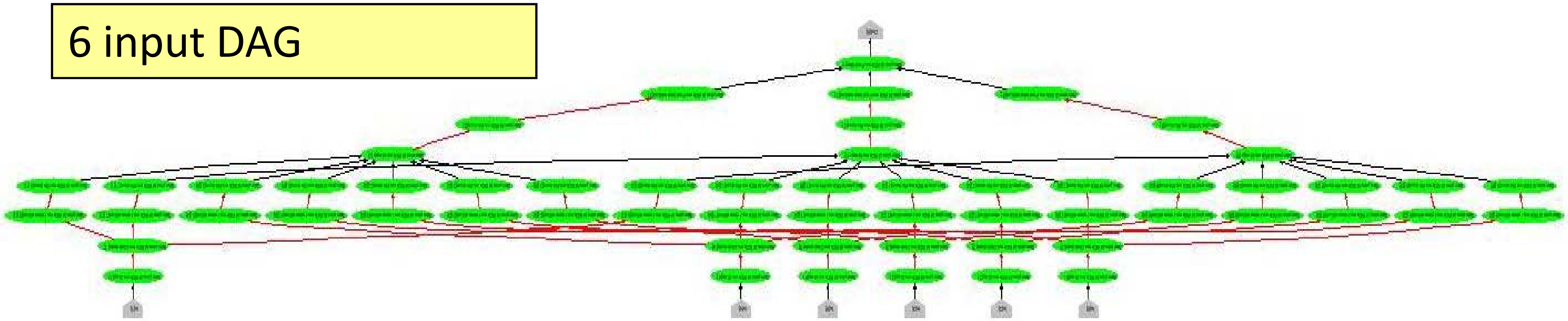
Merge Vertex

Save to Cache

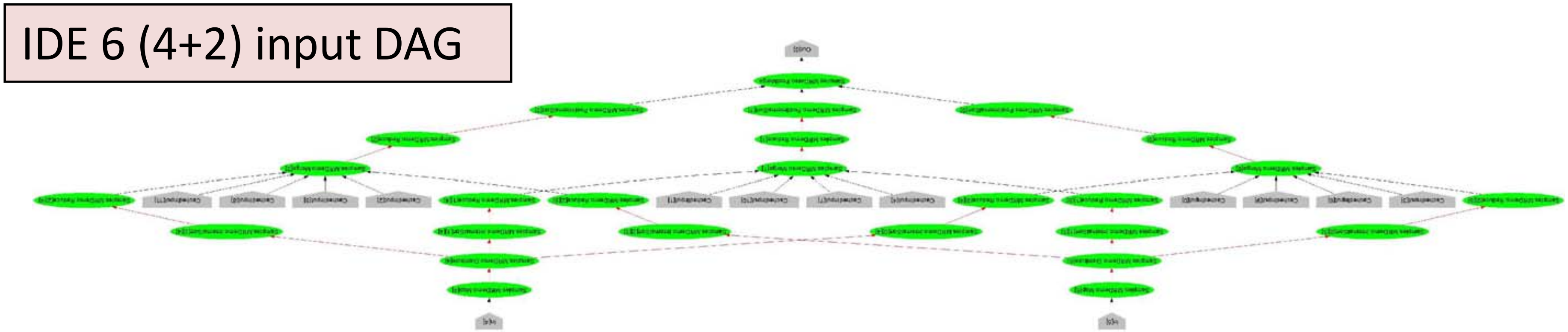
Incremental DAG – Remove Old Inputs

IDE in practice

6 input DAG

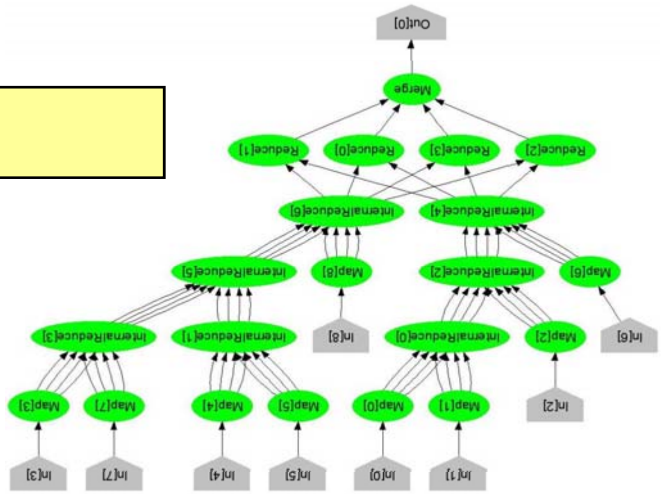


IDE 6 (4+2) input DAG

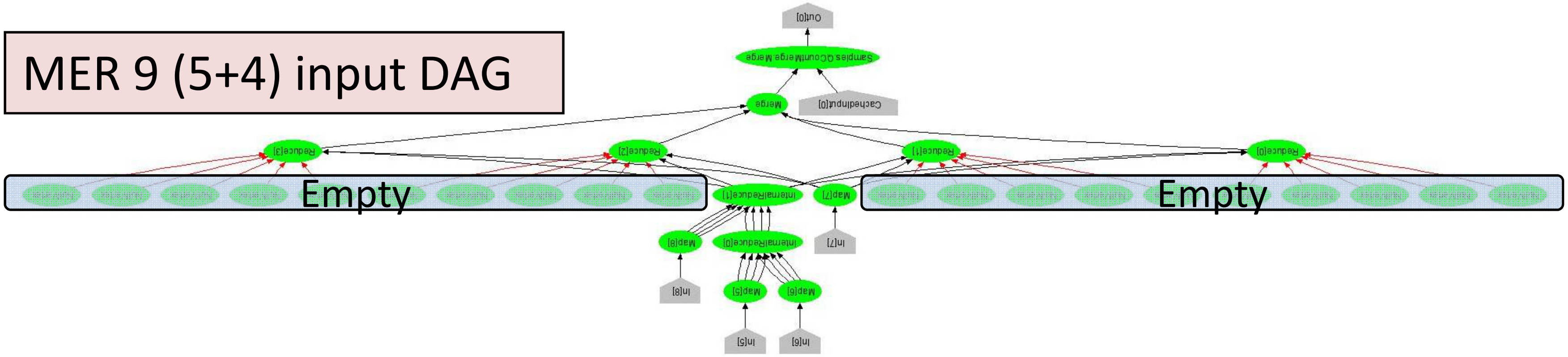


MER in practice

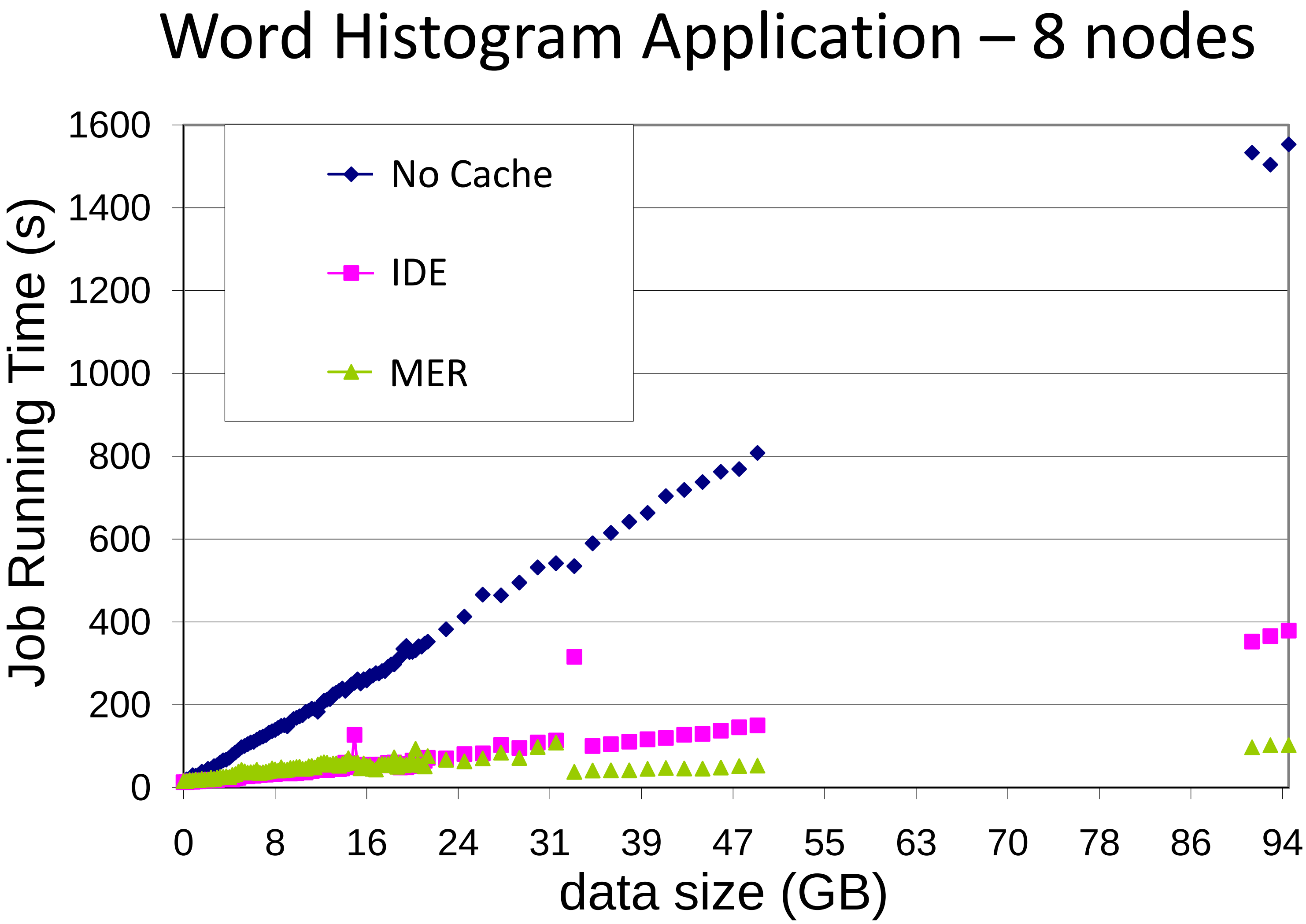
9 input DAG



MER 9 (5+4) input DAG



Evaluation – Running time



Discussion

- **MapReduce:** just a particular case
  - IDE reuses the output of Mappers
  - MER requires combined Reduce function

- **Combine IDE with MER:** benefits don’t add up
  - IDE can be used for the incremental DAG at MER

- **More semantic knowledge:** further opportunities
  - Generate merge function automatically
  - Improve incremental DAG

- **Sliding window on input data:** IDE works unchanged, MER requires “divide” besides merge

Conclusions & Questions

- **Problem:** reuse work in distributed computations on append-only data
- **Two methods:**
  - **INC** – reuse IDEntical past computations
    - No user effort
  - **MER** – MERge past results with new ones
    - Small user effort, potentially larger gains
- Implemented for **Dryad**