



Hybrid Approach for Dataflow Analysis of MPI Programs

Sriram Ananthakrishnan*, Greg Bronevetsky+, Ganesh Gopalakrishnan*

*University of Utah +Lawrence Livermore National Laboratory



Objective

- To improve precision of dataflow analysis on MPI programs
- Detect communication pattern statically

Dataflow Analysis

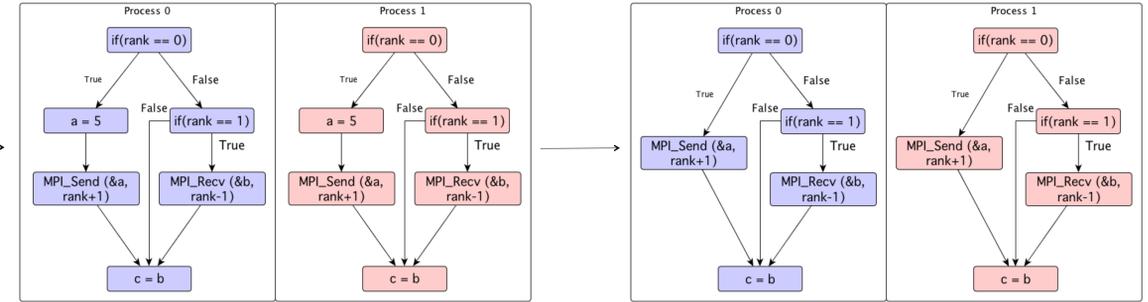
- Static compiler analysis technique to discover program properties
- Maintains an abstract program state for each control-flow graph node
- Abstract states are modified based on the semantics of the statement
- Program execution by the analysis is a sequence of abstract states

```

if(rank == 0) {
  a = 5;
  MPI_Send(&a, rank+1, ...);
}
else if(rank == 1) {
  MPI_Recv(&b, rank-1, ...);
}
c = b;

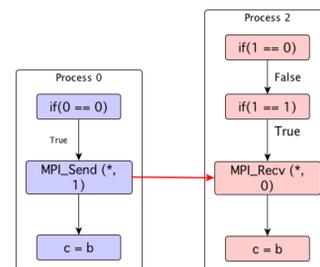
```

Source Code

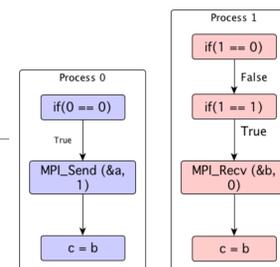


CFG

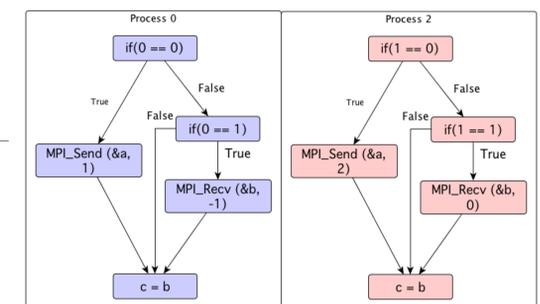
Slicing Analysis



Dynamic Send Receive Matching



Dead Path Elimination Analysis



Constant Propagation Analysis

Dataflow Analysis for MPI

- Dataflow analyses to improve performance or debugging of MPI applications require precise communication information
- Existing techniques over-approximates communication behavior or employ complex matching algorithm to determine communication statically

Our Approach

- Key insight: Communication is determined by values of rank and size of the MPI communicator
- Treat these variables concrete to determine values of target expressions in MPI function calls statically
- To solve message matching, replicate CFGs and connect send and receive operation in CFGs by concretely executing them

Details

- Each analysis instance is a composition of multiple analyses
- Each analysis instance is also a MPI process
- Slicing reduces the program to contain only statements that affect communication
- Constant propagation determines concrete values for target expressions in MPI communication calls
- Dead path elimination prunes out unreachable paths
- Communication invariance determines that communication is not input or message dependent
- Dynamic send receive matching concretely executes the operations exchanging CFG information to establish the communication edges

Fuse

- Our analysis composition is based on Fuse framework implemented in ROSE compiler
- Fuse allows transparent composition of multiple analyses requiring no knowledge of API or abstractions implemented by other analyses to exchange information

References

- Compositional Dataflow via Abstract Transition Systems (Fuse), Bronevetsky et al, LLNL TR
- Dataflow Analysis for MPI Programs, Strout et al, ICPP'06
- Communication sensitive static dataflow for MPI – Bronevetsky, CGO'09