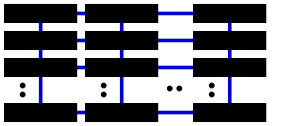
The Component-based Environment for Remote Sensing

Nathan DeBardeleben

Parallel Architecture Research Laboratory (PARL)
Clemson University, Clemson, South Carolina
http://www.parl.clemson.edu/

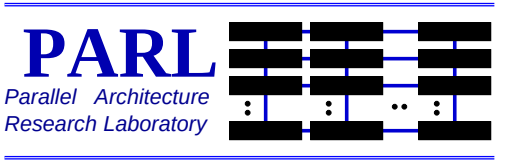
Outline

PARL
Parallel Architecture
Research Laboratory



- Introduction
- Related Work
- CERSe
- Case Studies and Results
- Conclusions

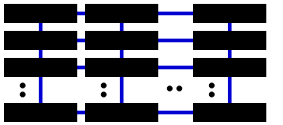
Problems Facing the RS Community



- Diversifying user base requires lower cost solutions
- Increasing temporal, spectral, and spatial resolution
- Massive data storage and processing power
- Parallel computers are difficult to program
- Required expertise in many application domains
- Re-engineering of hardware and software tools
- Rapid code development, execution, analysis, and maintenance

Proposed Solution

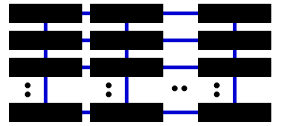
PARL
Parallel Architecture
Research Laboratory



- Provide:
  - High performance
    - Parallel computers
    - Usable without advanced parallel computing knowledge
  - Ease of use
    - Development and analysis tools
  - Extensibility
    - Code reuse

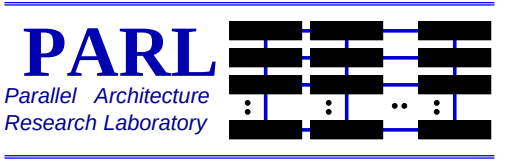
Related Work

PARL
Parallel Architecture
Research Laboratory

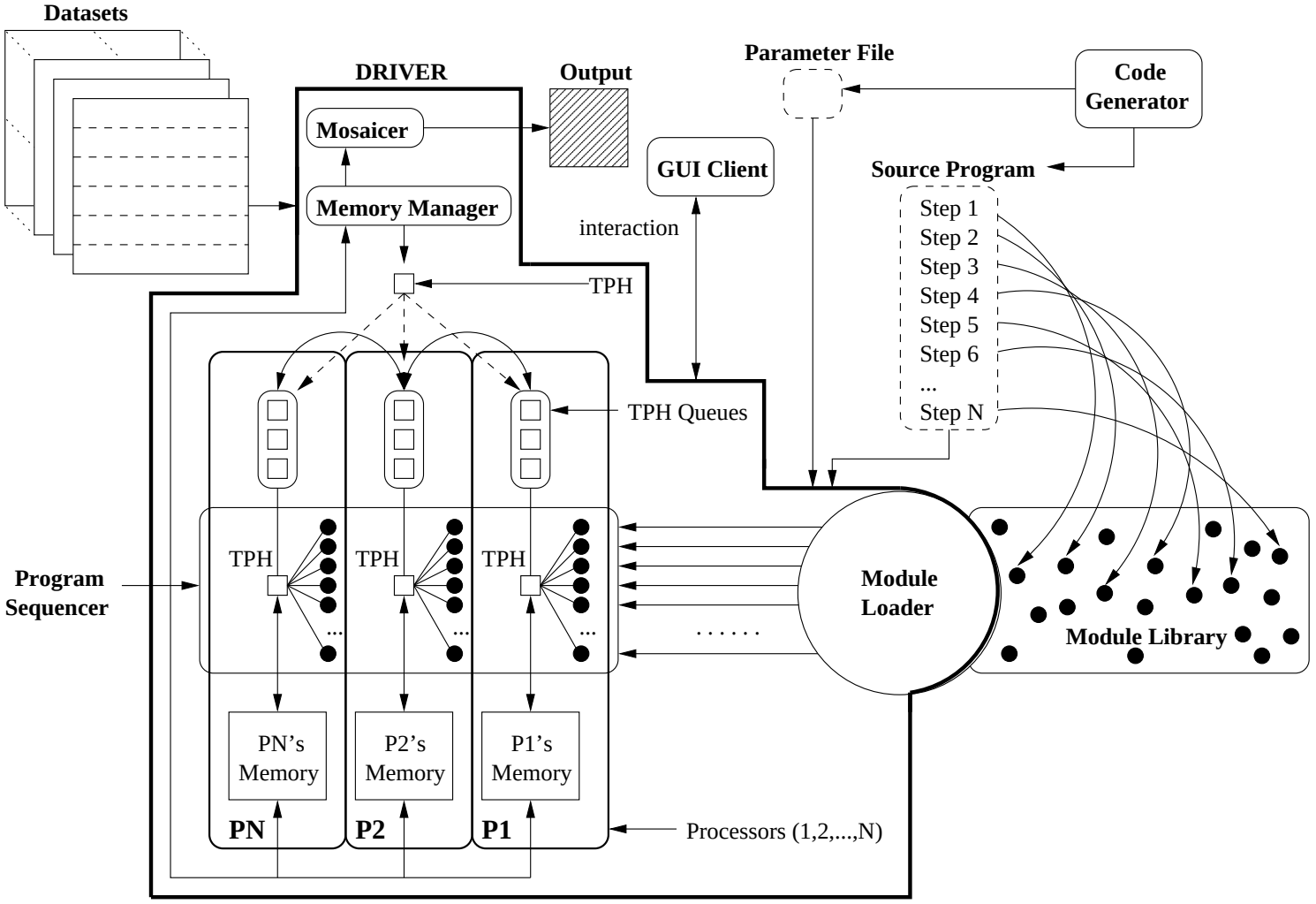


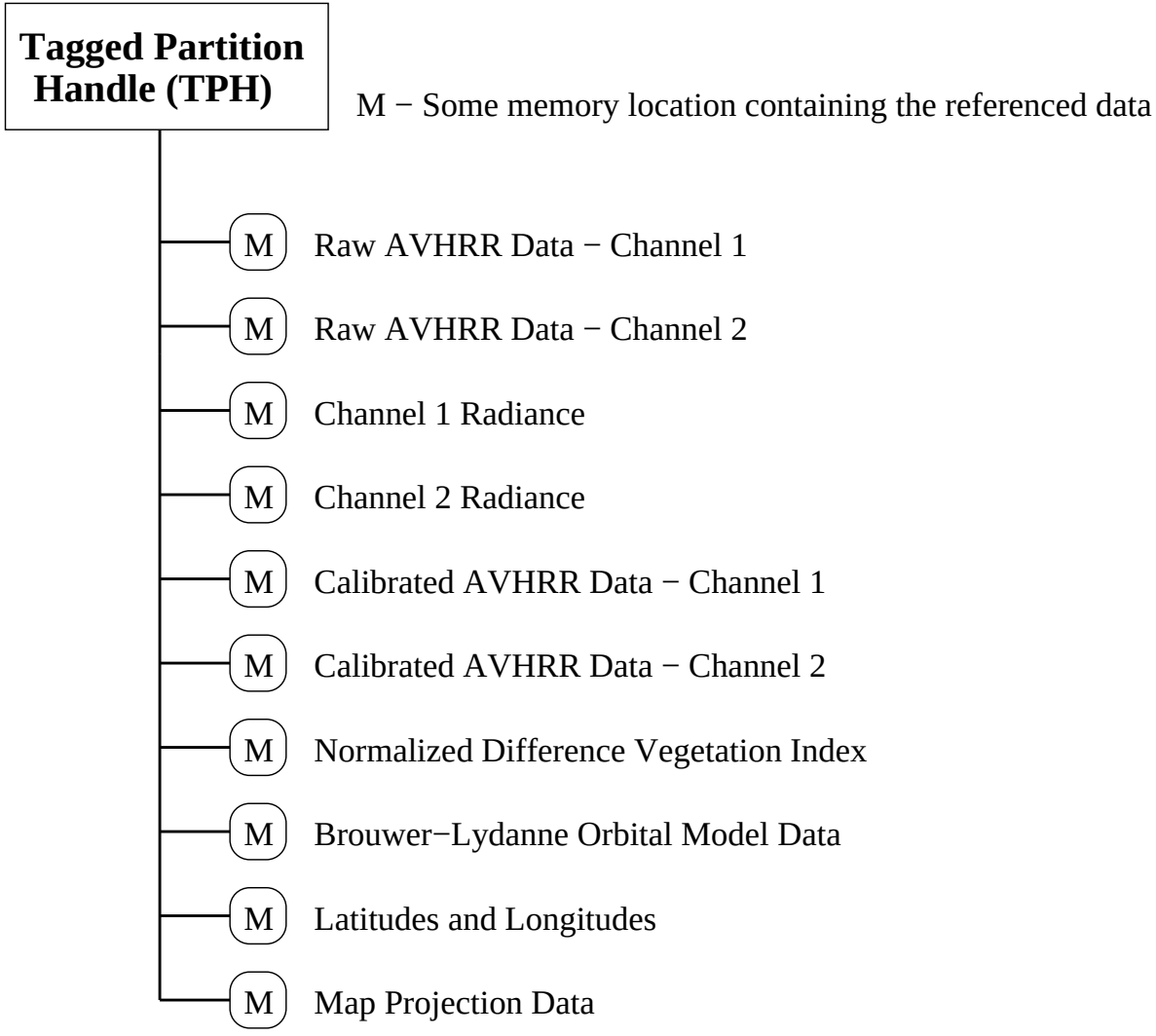
- RAC-RAT
- Khoros
- TRACEE
- Cactus
- OSSIM

CERSe



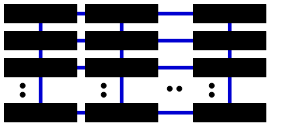
- Core is an execution engine
- Interprets a dataflow graph
- Abstracts away understanding of parallelism


Datasets
DRIVER
Output
Parameter File
Code Generator
Mosaicer
GUI Client
Source Program
Memory Manager
interaction
Step 1
Step 2
Step 3
Step 4
Step 5
Step 6
...
Step N
TPH
TPH Queues
TPH
TPH
TPH
Program Sequencer
Module Loader
Module Library
PN's Memory
P2's Memory
P1's Memory
PN
P2
P1
Processors (1,2,...,N)


Tagged Partition Handle (TPH)
M − Some memory location containing the referenced data
M
Raw AVHRR Data − Channel 1
M
Raw AVHRR Data − Channel 2
M
Channel 1 Radiance
M
Channel 2 Radiance
M
Calibrated AVHRR Data − Channel 1
M
Calibrated AVHRR Data − Channel 2
M
Normalized Difference Vegetation Index
M
Brouwer−Lydanne Orbital Model Data
M
Latitudes and Longitudes
M
Map Projection Data

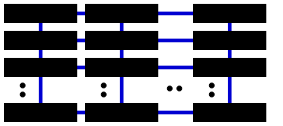
Tagged Partition Handle

PARL
Parallel Architecture
Research Laboratory



- Keeps related data in one place
- Content addressible
- Easy to move many data sets between modules
- Programmers need to know only interface

Modules



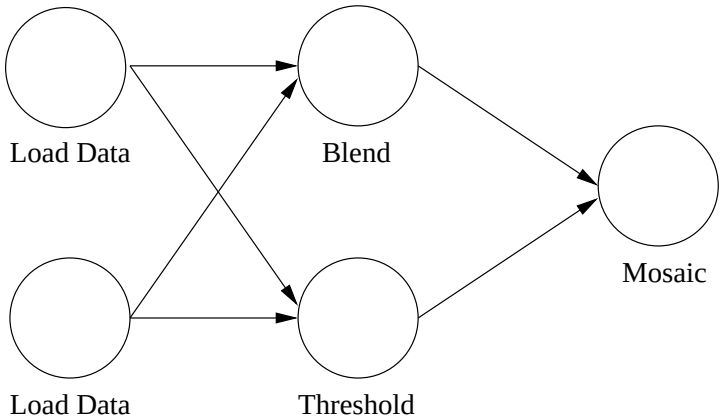
- Stand alone piece of C code
- Interconnected with other modules
- Interface defined by input, output, inout, and parameter data types
- Interface accesses TPH
- Utility functions: Count, Size, TPH

Code Generator

- Creates *source program* and *parameter file*
- Source program
  - C data structures
  - Mapping between shared library module and module name
  - Module names to execute with specific arguments

Simple Example

PARL
Parallel Architecture
Research Laboratory




```
CERSe_Program Program[] = {
  {"Load_Data",Load_Data_args_1},
  {"Load_Data",Load_Data_args_2},
  {"Blend",Blend_args_1},
  {"Threshold",Threshold_args_1},
  {"Mosaic",Mosaic_args_1},
};
```

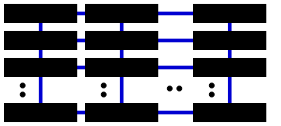
Code Generator

PARL
Parallel Architecture
Research Laboratory

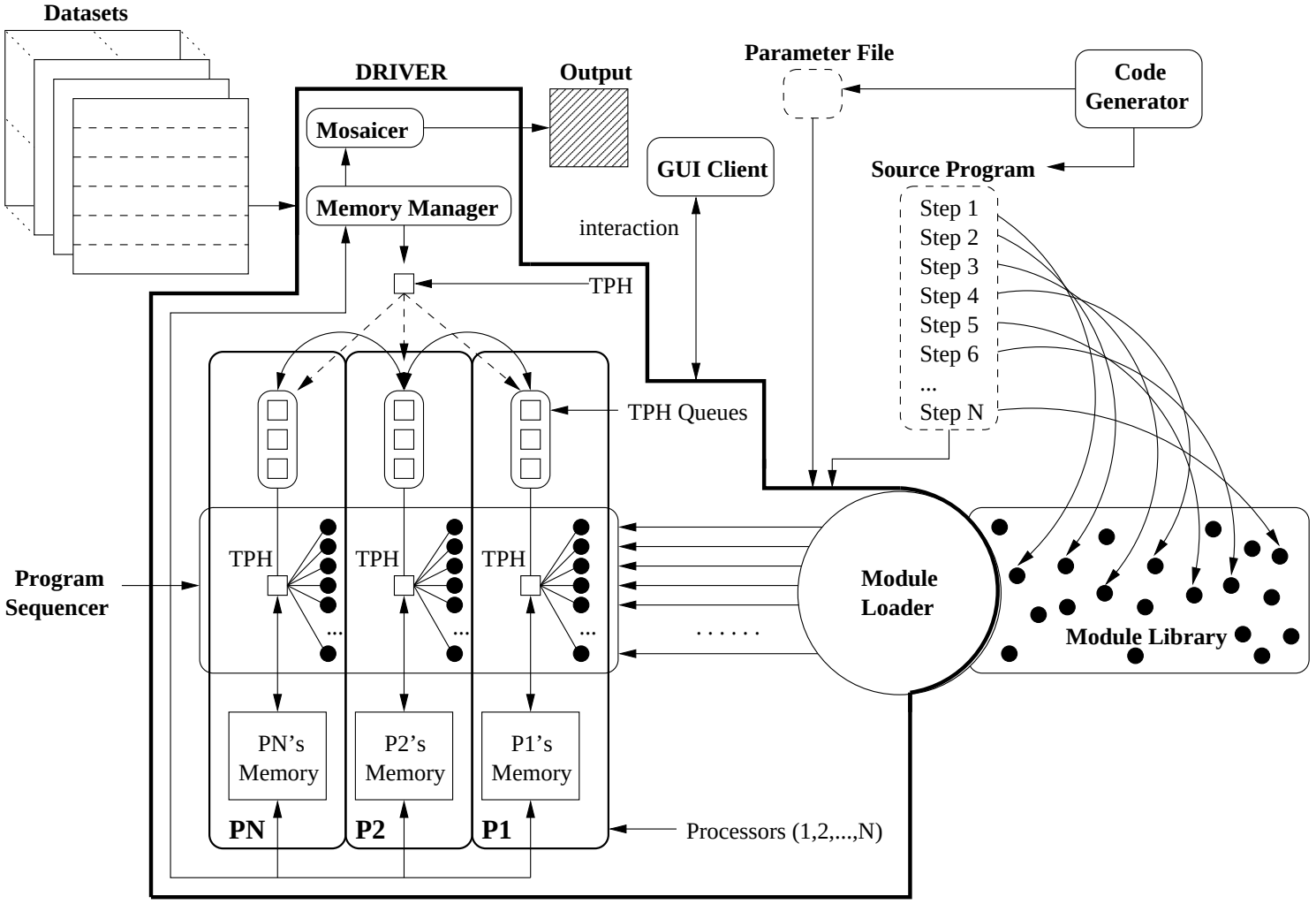
- Parameter file
  - List of datasets
  - Number of processors
  - Partitioning information
  - Working directories

The CERSe Driver

PARL
Parallel Architecture
Research Laboratory

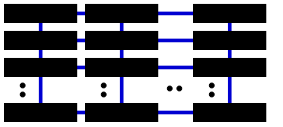


- Module Loader
- Memory Manager
- Datasets
- Program Sequencer
- Mosaicer


Datasets
DRIVER
Output
Parameter File
Code Generator
Mosaicer
GUI Client
Source Program
Memory Manager
interaction
Step 1
Step 2
Step 3
Step 4
Step 5
Step 6
...
Step N
TPH
TPH Queues
TPH
TPH
TPH
Program Sequencer
Module Loader
Module Library
PN's Memory
P2's Memory
P1's Memory
PN
P2
P1
Processors (1,2,...,N)

The CERSe Driver

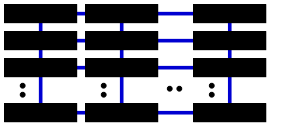
PARL
Parallel Architecture
Research Laboratory



- Out-of-Core Computation
- Parallel I/O

GUI Clients

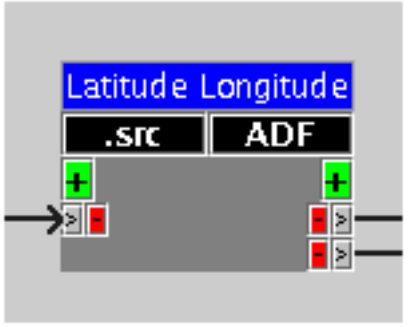
PARL
Parallel Architecture
Research Laboratory



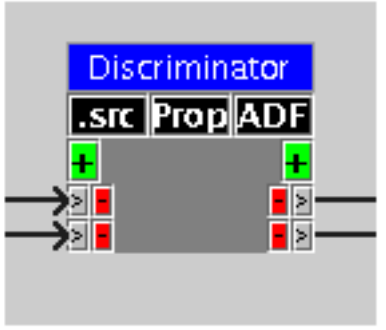
- Performance but not yet flexibility and usability
- GUI is not required
- Underlying the CERSe GUI is ADF
- Agents: graphical editor and code generator

GUI Modules

PARL
Parallel Architecture Research Laboratory



(A)



(B)

GUI Client

PARL
Parallel Architecture
Research Laboratory

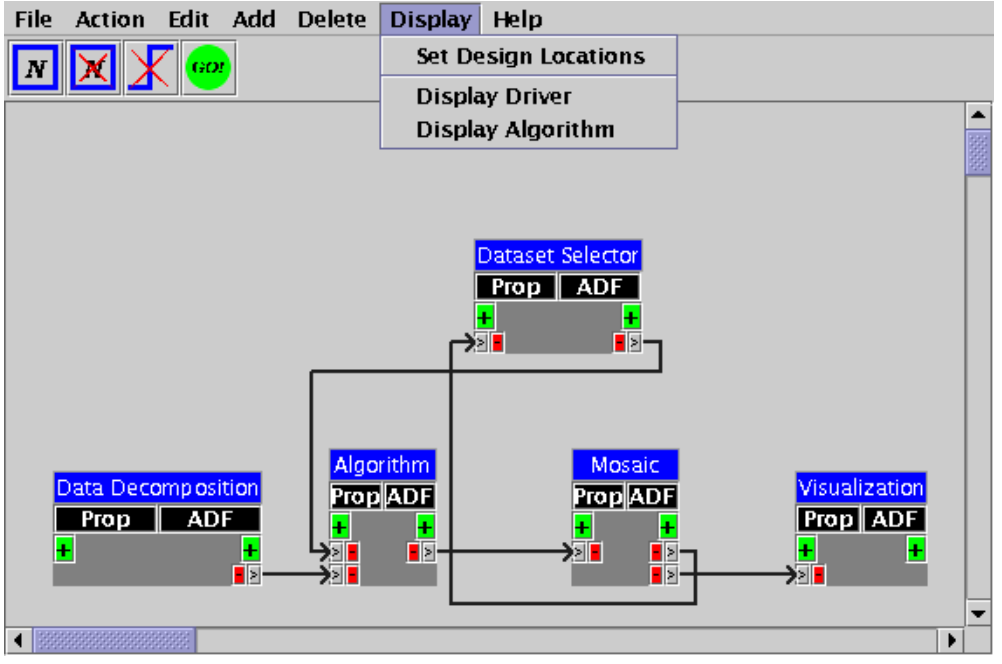
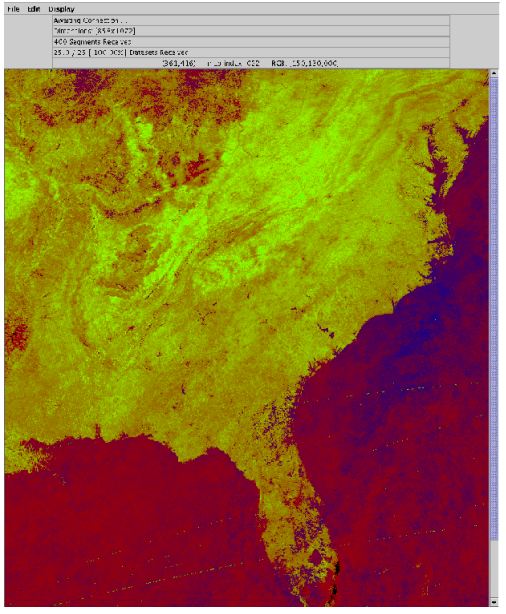


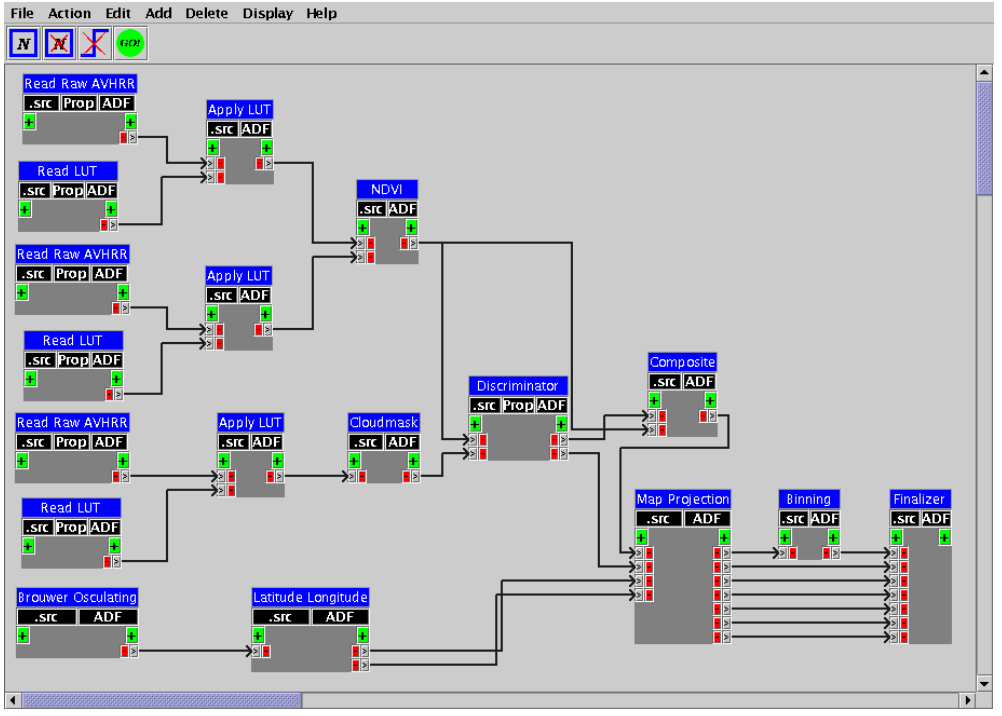
Image Viewer Server

PARL
Parallel Architecture
Research Laboratory



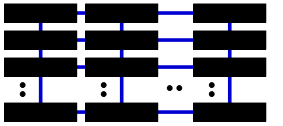
Case Study: NDVI

PARL
Parallel Architecture Research Laboratory

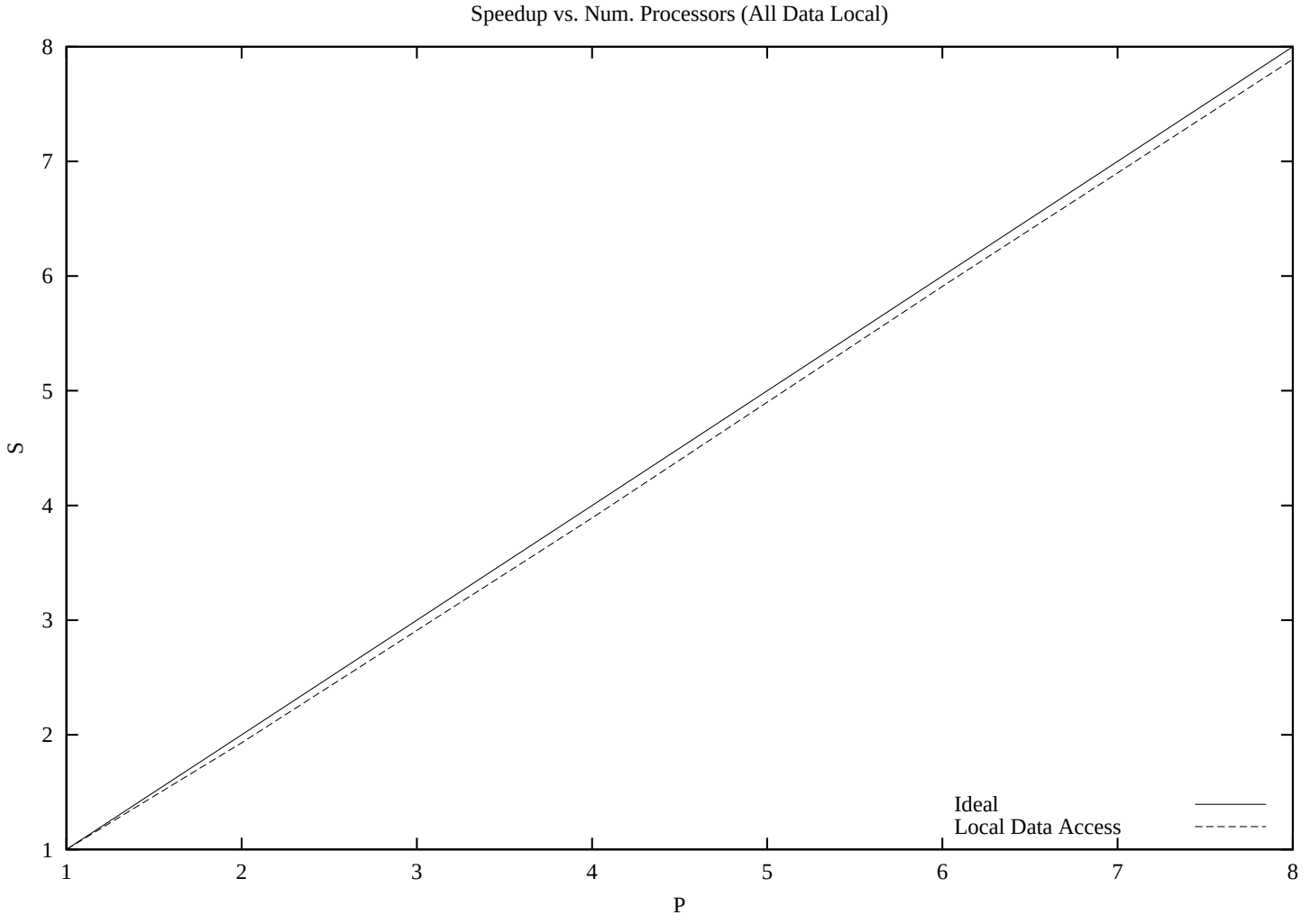


Additional Case Studies

PARL
Parallel Architecture
Research Laboratory

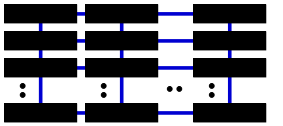


- Sea Surface Temperature
- NDVI-SST Composite
- Interactive Web-based Simulations

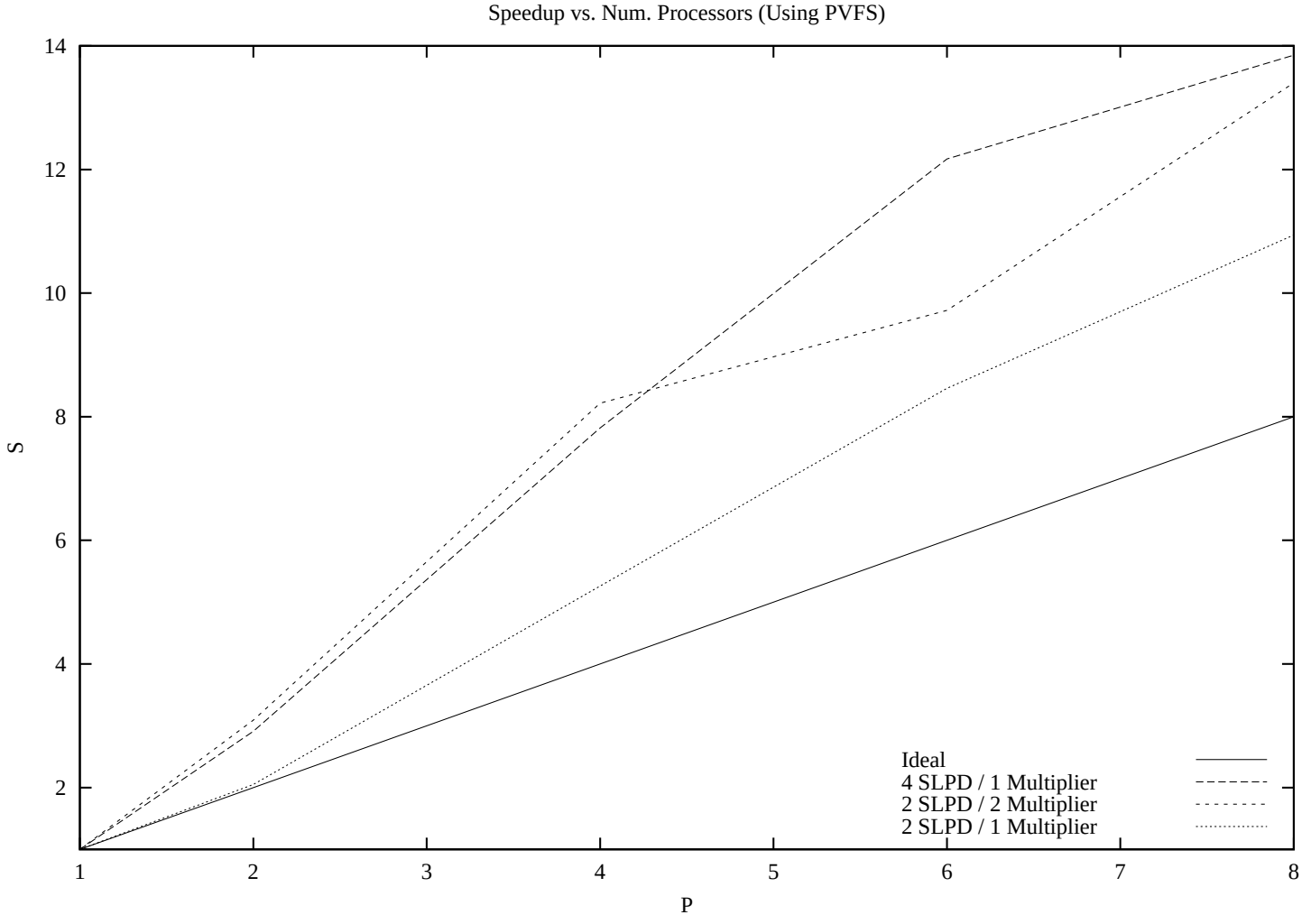

Speedup vs. Num. Processors (All Data Local)
S
P
1
2
3
4
5
6
7
8
Ideal
Local Data Access

Performance Results

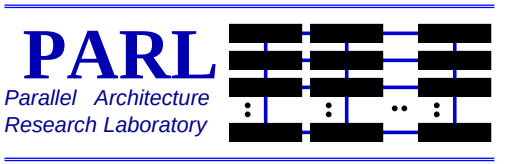
PARL
Parallel Architecture
Research Laboratory



Processors	exec. time (4,1)	exec. time(2,2)	exec. time (2,1)
1	92,752	99,135	101,643
2	31,890	32,100	49,690
4	11,849	12,055	19,321
6	7,621	10,198	12,021
8	6,248	7,390	9,292

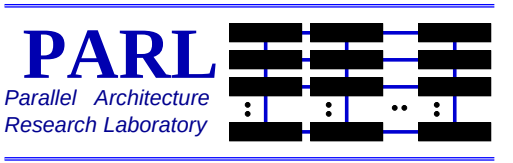

Speedup vs. Num. Processors (Using PVFS)
S
P
14
12
10
8
6
4
2
1
2
3
4
5
6
7
8
Ideal
4 SLPD / 1 Multiplier
2 SLPD / 2 Multiplier
2 SLPD / 1 Multiplier

Conclusions



- Ease of use
  - GUI - algorithm creation and data visualization
- Extensibility
  - Pluggable modules, easy module interface
- High performance
  - Automatic parallelization, nearly ideal speedup

Future Work



- Larger scalability tests
- I/O nodes seperate from compute nodes
- Additional satellite data formats
- Introspection