6th Annual MVAPICH User Group (MUG) Meeting

清華大學
Tsinghua University

Combining Static and Dynamic Analysis for Top-down Communication Trace Compression

Jidong Zhai

Tsinghua University

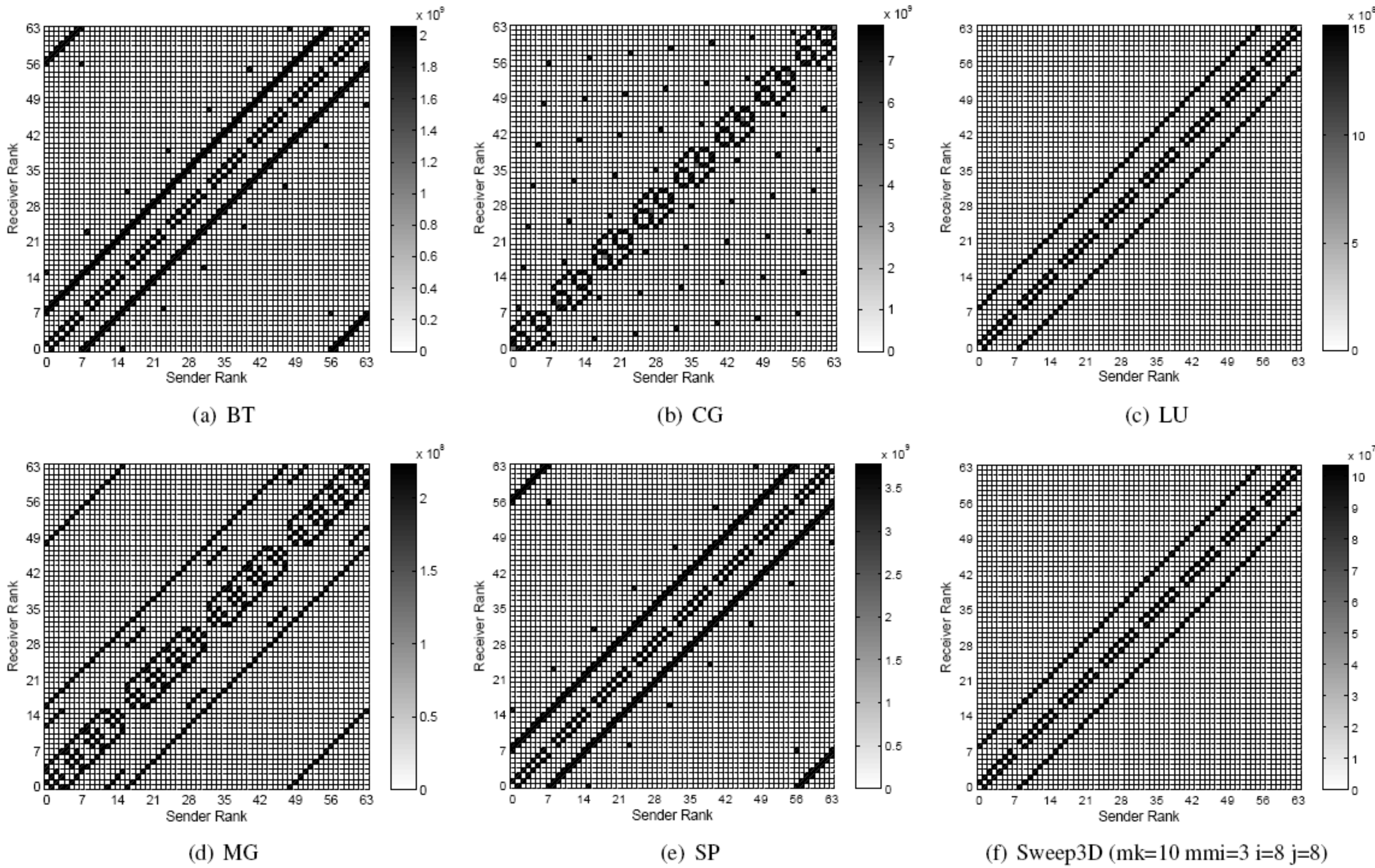
Communication Traces

- **Communication Traces highly useful:**
  - Communication performance analysis
    - Bottleneck/Bug detection
    - Platform selection
    - Communication optimization
  - Design future HPC systems

ID	MPI_TYPE	SIZE	SRC	DEST	TIME
1.	MPI_Send	20B	10	16	10us
2.	MPI_Recv	10B	12	10	12us
3.	MPI_Send	20B	10	12	10us
4.	MPI_Recv	10B	11	10	11us
5.	MPI_Send	20B	10	11	12us
6.	MPI_Recv	10B	14	10	15us
7.	MPI_Send	20B	10	14	11us
8.	MPI_Recv	10B	16	10	13us
……					

Segment of MPI communication trace

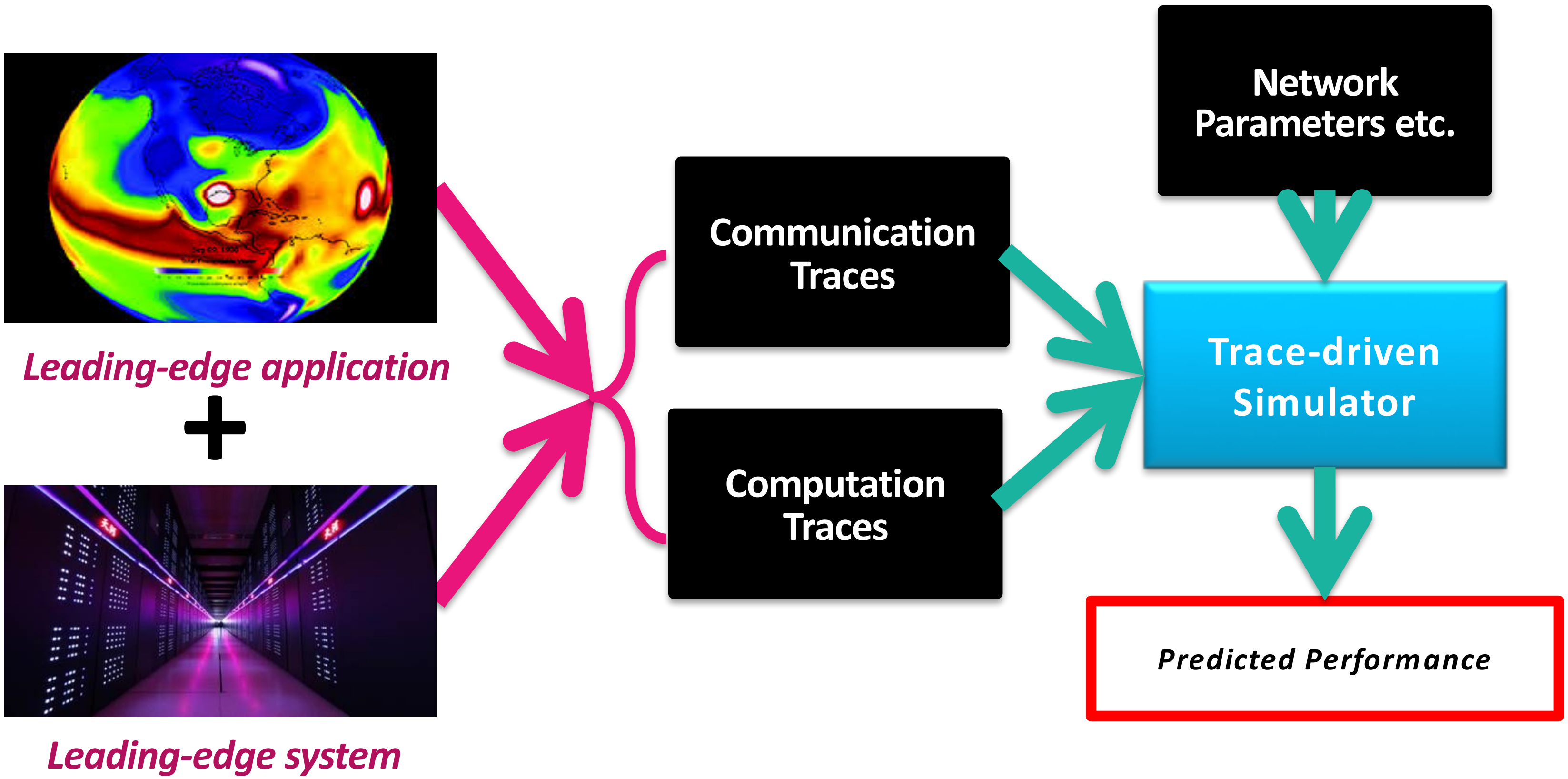
Analyzing Communication Patterns



Communication patterns extracted from MPI communication traces.

[1] Ruini Xue, Xuezheng Liu, Ming Wu, Zhenyu Guo, Wenguang Chen, Weimin Zheng, Zheng Zhang, Geoffrey M. Voelker: MPIWiz: subgroup reproducible replay of mpi applications. PPOPP 2009: 251-260.

Predicting Future HPC Systems



[1] A. Snavely, L. Carrington, N. Wolter, J. Labarta, R. Badia, and A. Purkayastha. A framework for application performance modeling and prediction. In SC’02, pages 1–17, 2002.

Tracing Communication: Challenges

- Problem: Traces → Huge
  - Caused by growing
    - Problem size
    - job scale (# of processes)
  - Example:
    - ASCI SMG2000, 64*64*32 Problem size, 22,538 Processes → **5 TB Traces**[1]
- **Huge traces bring**
  - Pressure on storage system
  - Interference with user application execution

[1] B. J. N. Wylie, M. Geimer, and F. Wolf, "Performance measurement and analysis of large-scale parallel applications on leadership computing systems," Sci. Program., vol. 16, no. 2-3, pp. 167–181, Apr. 2008.

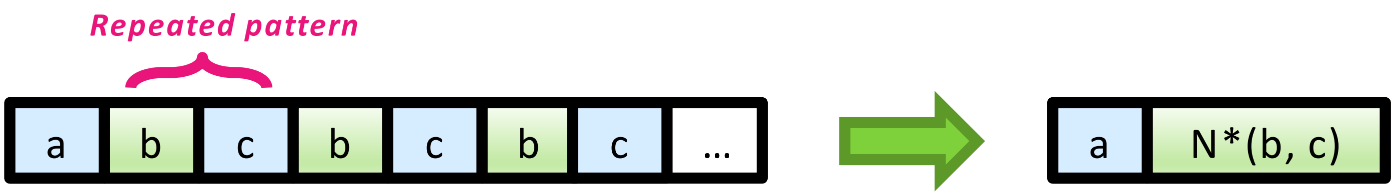
Solution: Trace Compression

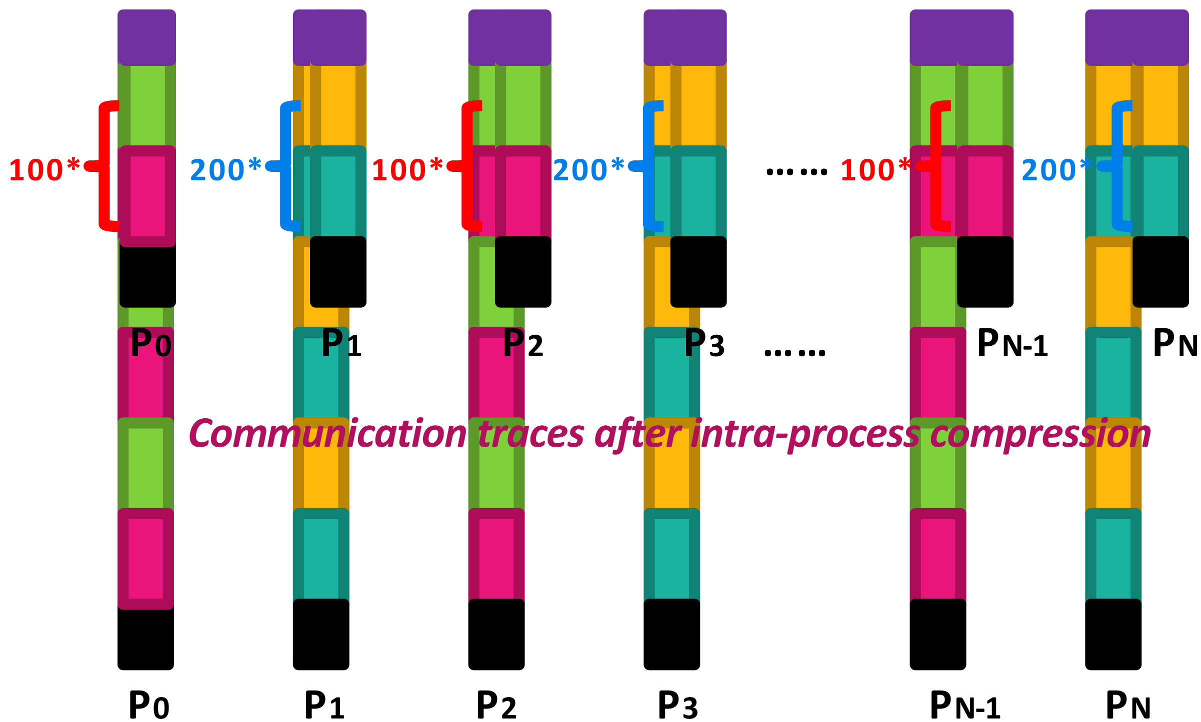
- Significant **redundancy**
  - Within a process: iterative timesteps (loops)
  - Across processes: similar behavior (SPMD)
- **Existing tools: *Bottom-up* approach**
  - ScalaTrace[1], ScalaTrace-2[2]
  - Examine traces at runtime to discover repeated patterns
  - **Two-phase**: intra-process + inter-process

[1] M. Noeth, F. Mueller, M. Schulz, and B. de Supinski, "Scalable compression and replay of communication traces in massively parallel environments," in IPDPS'07, 2007.
[2] X. Wu and F. Mueller, "Elastic and scalable tracing and accurate replay of non-deterministic events," in ICS'13, 2013.

Bottom-up 1: *Intra*-Process Compression

- **Identify repeated pattern within one process**

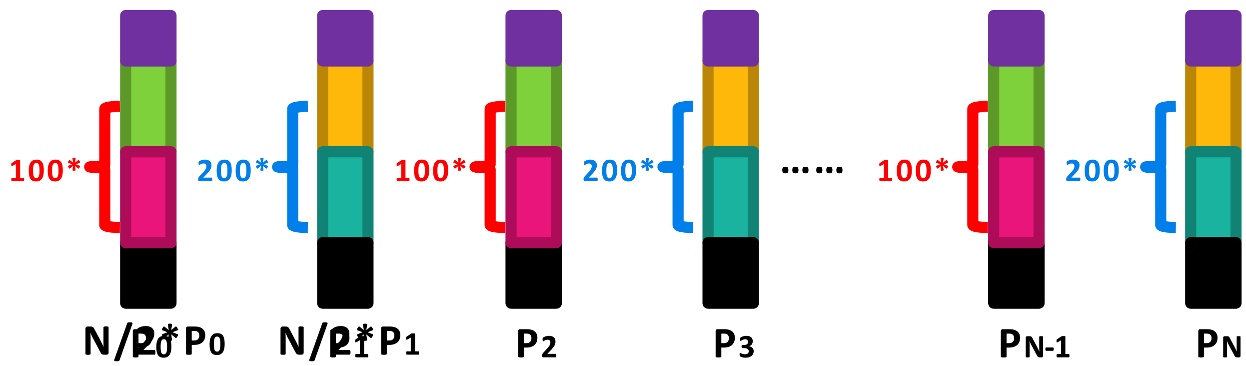




Communication traces for each process.

Bottom-up 2: *Inter*-Process Compression

- **Compare traces in horizontal dimension to identify repeated patterns**



Communication traces after intra-process compression

Limitations of Bottom-Up Compression

- **Intra-Process:**
  - Expensive with complex communication patterns
  - Nested loop structures[1]

```
for (i=0; i<10; i++)
    for (j=0; j<=i; j++) {
        if (j%2 == 0)
            MPI_Isend(…);
        else
            MPI_Irecv(…);
    }
    MPI_Waitall(…);
    MPI_Reduce(…);
}
```

MPI Program snippet

```
MPI_Isend(…)
MPI_Waitall(…)
MPI_Reduce(…)
MPI_Isend(…)
MPI_Irecv(…)
MPI_Waitall(…)
MPI_Reduce(…)
MPI_Isend(…)
MPI_Irecv(…)
MPI_Isend(…)
MPI_Waitall(…)
MPI_Reduce(…)
……
```

Communication trace segment

[1] Q. Xu, J. Subhlok, and N. Hammen, “Efficient discovery of loop nests in execution traces,” in MASCOTS’10, 2010, pp. 193–202.

Limitations of Bottom-Up Compression

- **Inter-Process:**
  - Even more expensive, not scalable
  - Complexity for two processes → $O(n^2)$

*4*MPI_Send*
*3*MPI_Recv*
*5*MPI_Bcast*
*3*MPI_Recv*
……

comm. traces of process P_0

*3*MPI_Isend*
*4*MPI_Irecv*
*5*MPI_Wait*
*3*MPI_Barrier*
……

comm. traces of process P_1

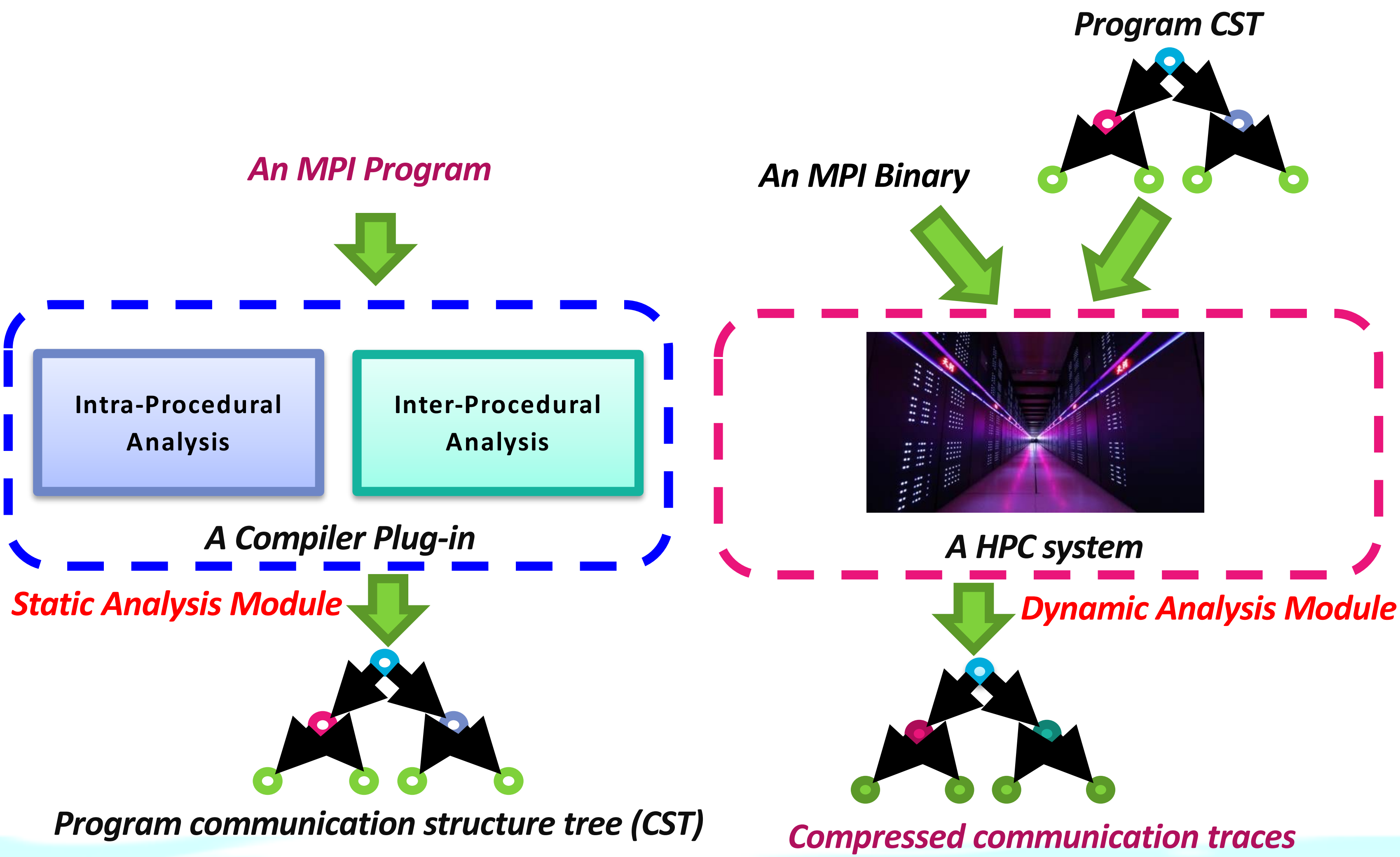
*4*MPI_Send*
*3*MPI_Recv*
*5*MPI_Bcast*
*3*MPI_Recv*
……
*3*MPI_Isend*
*4*MPI_Irecv*
*5*MPI_Wait*
*3*MPI_Barrier*
……

Compressed traces of P_0 and P_1

Our Approach: CYPRESS

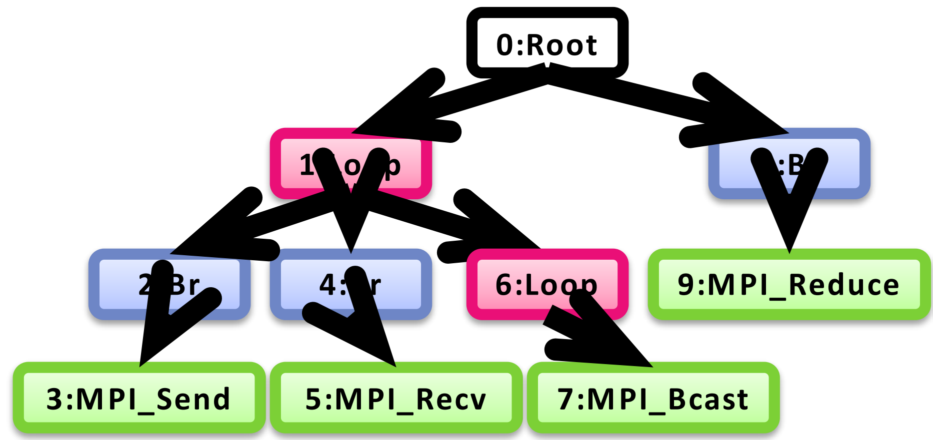
- ***Top-down*** technique
  - Combines **static** and **dynamic analysis**
  - **Static analysis phase**
    - Extract program communication structure tree
      - Loops, branches etc.
  - **Dynamic analysis phase**
    - Use this tree as a template
    - “Fill in” runtime information
      - Loop count

Overview of CYPRESS



CYPRESS: Advantages

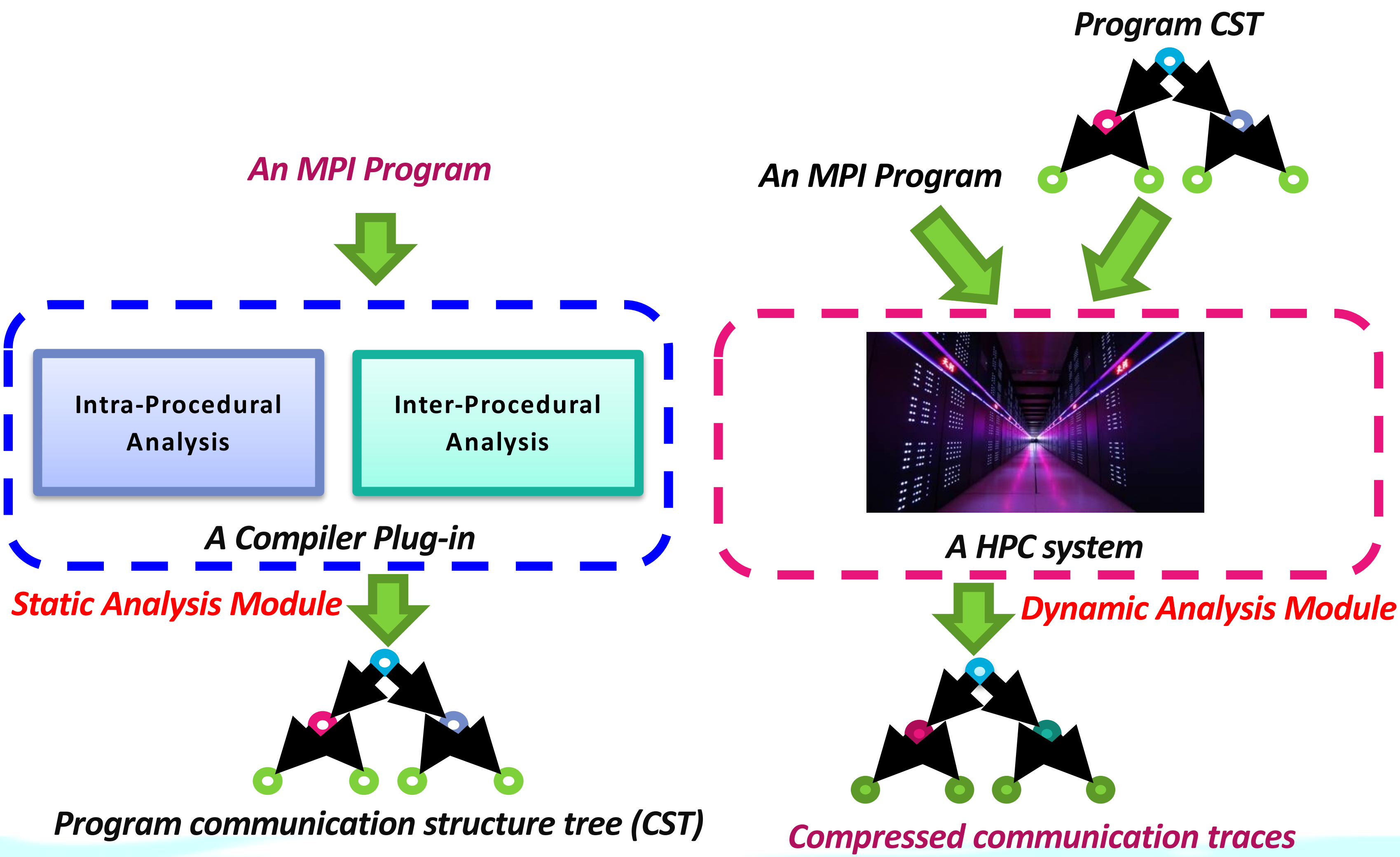
- Much faster
  - **Top-down "Big picture"**
    - Redundancy expected
    - Cut the guessing
  - **More efficient**
    - Knows "**where to stop**"
- Scalable
  - Significant improvement to Inter-process compression
  - A better worst-case complexity
  - More affordable for large-scale analysis




```
1:MPI_Isend(...)        10:MPI_Isend(...)
2:MPI_Waitall(...)      11:MPI_Waitall(...)
3:MPI_Reduce(...)       12:MPI_Reduce(...)
4:MPI_Isend(...)        13:MPI_Isend(...)
5:MPI_Irecv(...)        14:MPI_Irecv(...)
6:MPI_Waitall(...)      15:MPI_Isend(...)
7:MPI_Reduce(...)       16:MPI_Irecv(...)
8:MPI_Isend(...)        ...
9:MPI_Irecv(...)
```

?

CYPRESS: Static Analysis

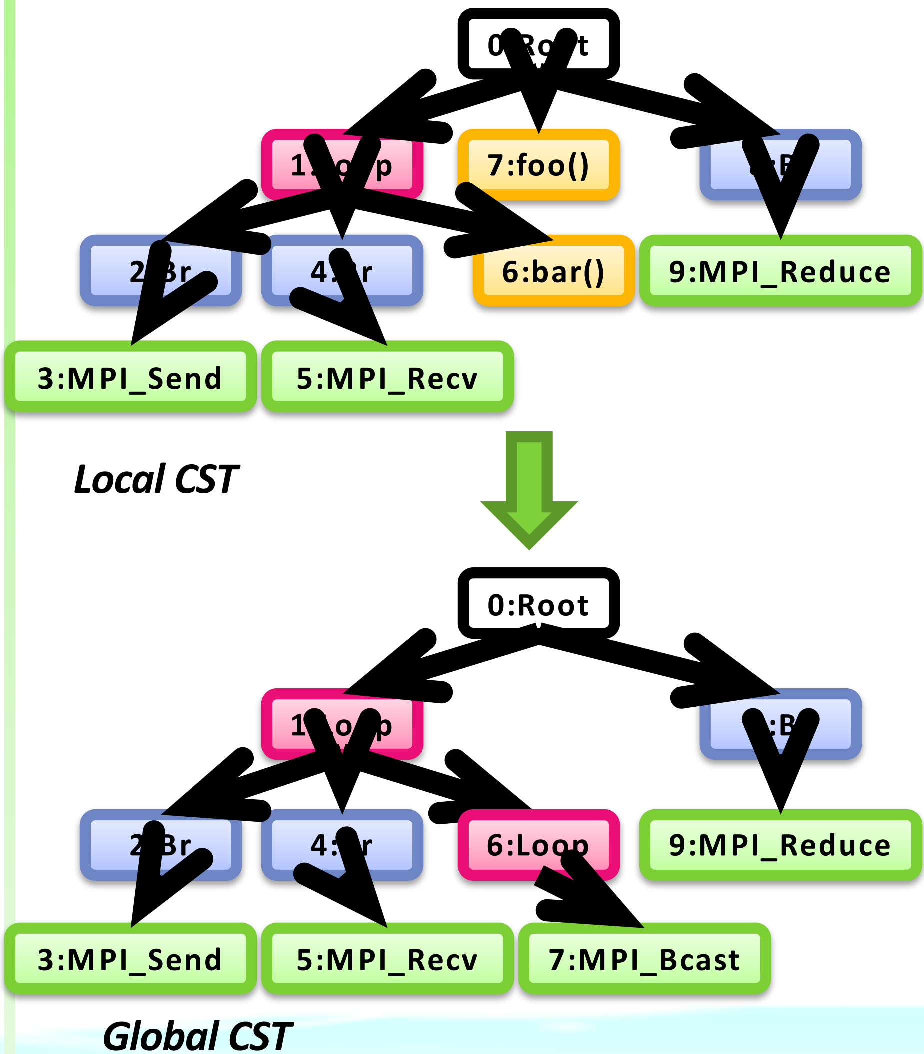


Communication Structure Tree (CST)

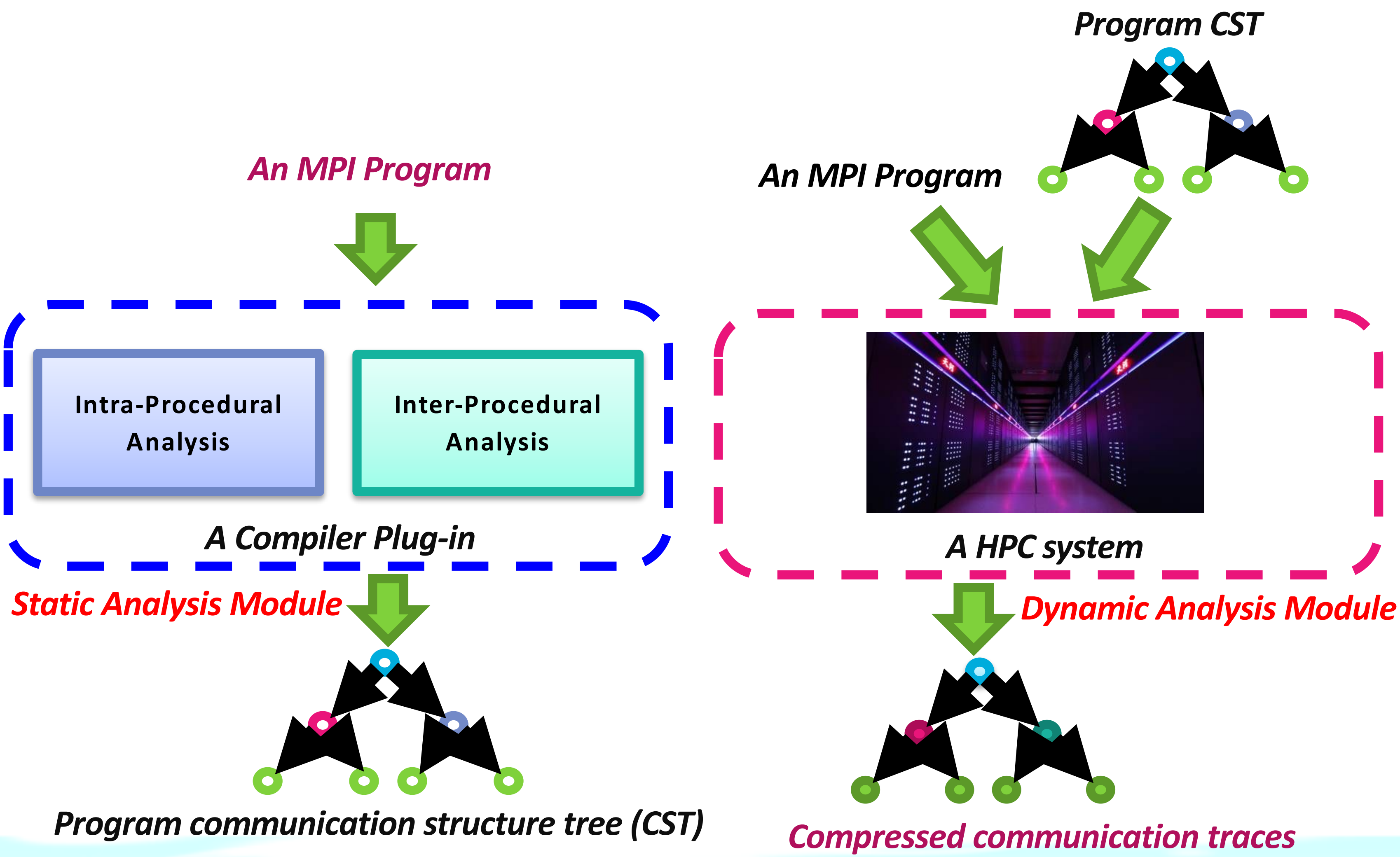
- Ordered tree
  - Program communication structure
  - Non-leaf nodes:
    - Loop
    - Branch
  - Leaf nodes:
    - MPI communication calls
      - » MPI_Send, MPI_Recv

Build Communication Structure Tree

```
int main(){
  for (i=0; i<k; i++){
    if (myid % 2 == 0)
      MPI_Send(buf, size, MPI_INT,
               myid+1, 0, MPI_COMM_WORLD);
    else
      MPI_Recv(buf, size, MPI_INT,
       myid-1, 0, MPI_COMM_WORLD, &status);
    bar();
  }
  foo();
  if (myid % 2 == 0)
     MPI_Reduce(sbuf, rbuf, 1, MPI_INT,
                     MPI_SUM, root, comm);
}
int bar(){
  for(k=0; k<n; k++)
    MPI_Bcast(buf, size, MPI_INT, root, 18
                  MPI_COMM_WORLD);
}
int foo(){
  for(j=0; j<m; j++)
     sum += j;
}
```

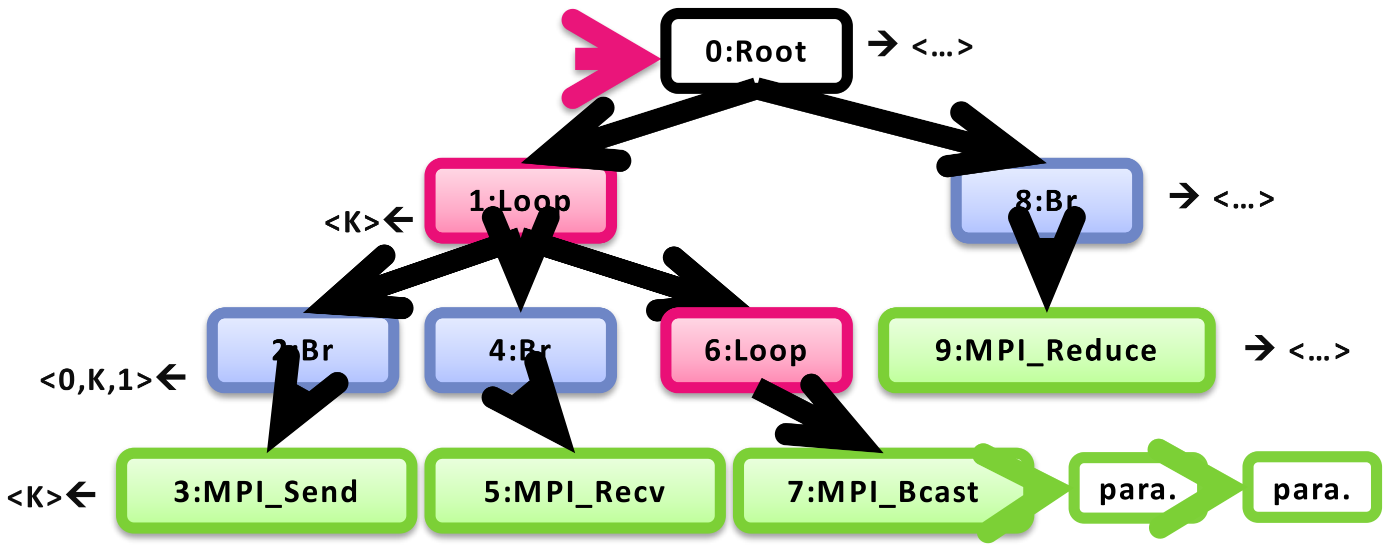


CYPRESS: Dynamic Analysis



Intra-Process Compression

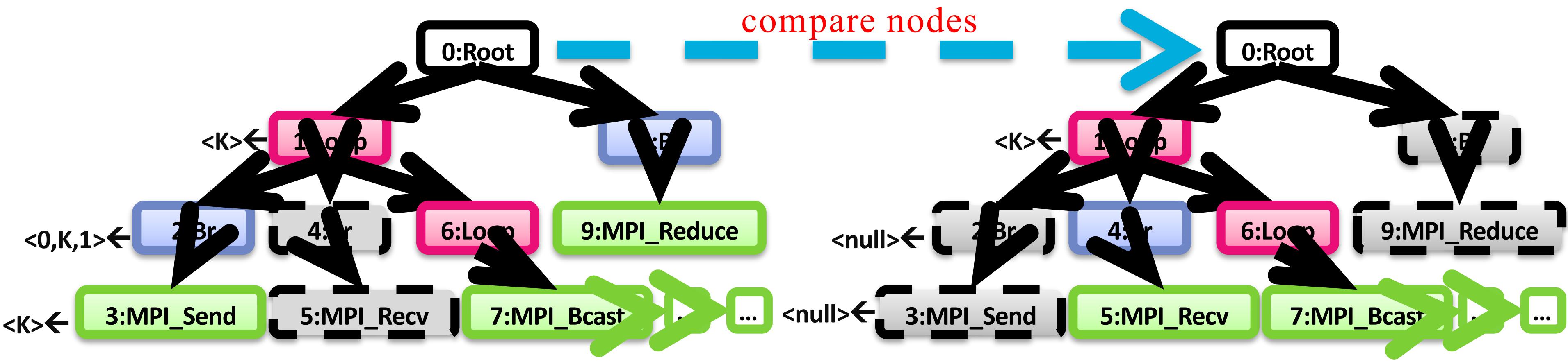
- "Fill in" dynamic information into CST
  - Record loop count and branch taken status
  - Per-node linked list for MPI event details



Fill in dynamic information into the CST.

Inter-Process Compression

- Compare corresponding nodes of the trees



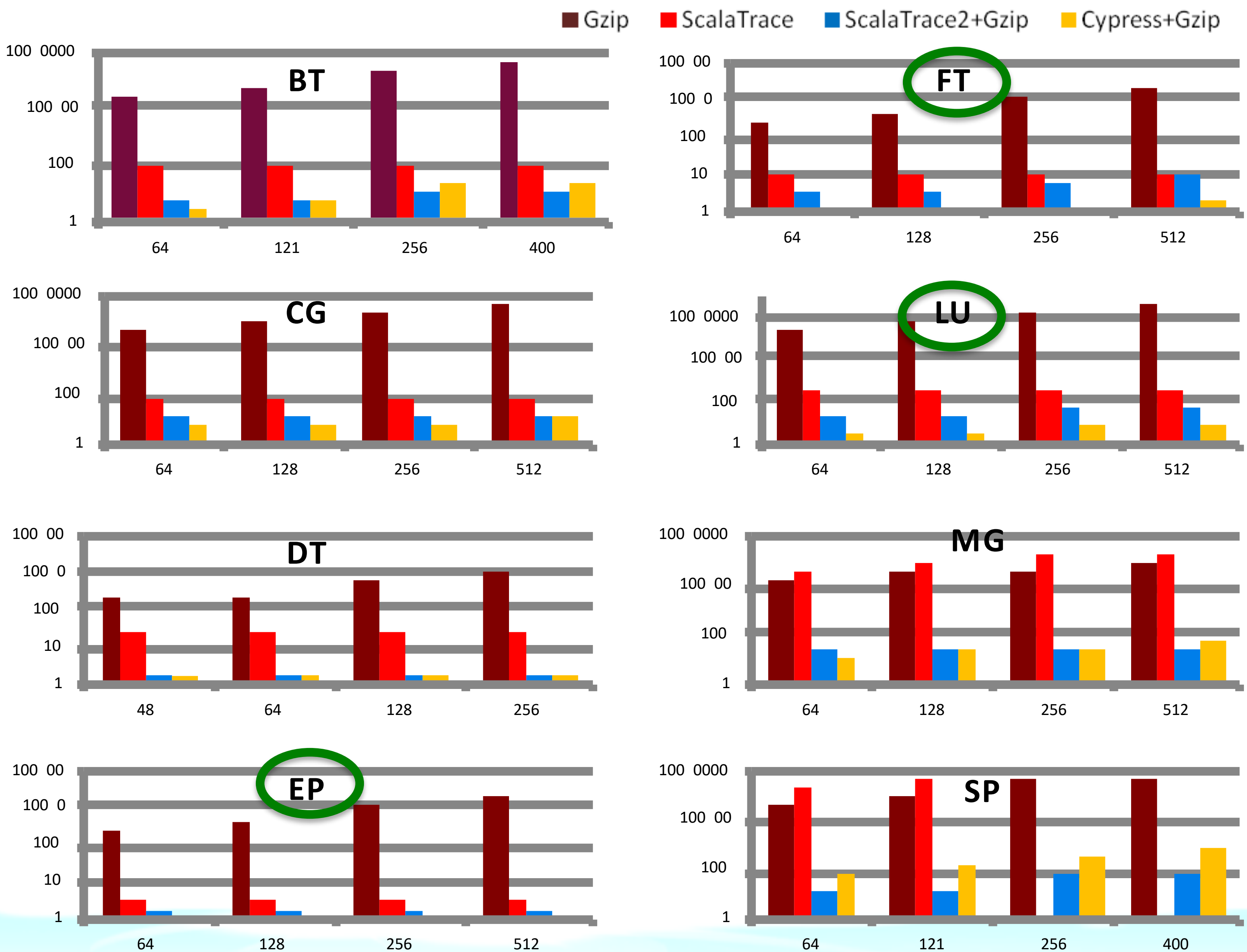
A compressed trace tree for P0

A compressed trace tree for P1

Experiments

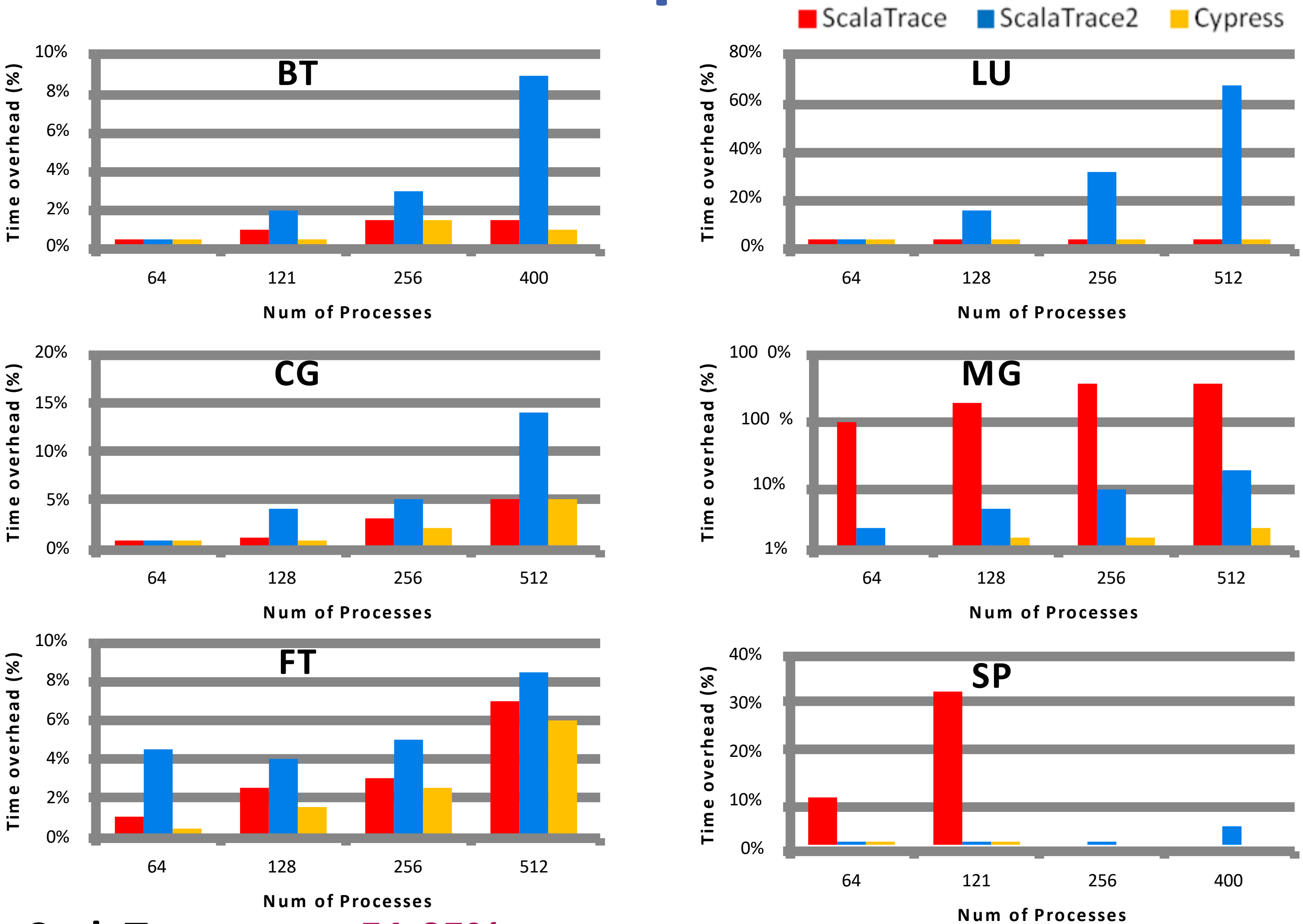
- **Benchmarks**
  - 8 NPB programs: BT, CG, DT, EP, FT, LU, MG, SP
  - Real Application: LESlie3d
- **Platform**
  - Explorer-100 HPC cluster
    - 2*Intel Xeon X5670/48GB Memory
    - Infiniband network, MVAPICH-2
- **Alternative systems**
  - Gzip (offline compression)
  - ScalaTrace (online, bottom-up)
  - ScalaTrace-2 (online, bottom-up)
- **Metrics**
  - Compression effectiveness (trace size reduction)
  - Compression overhead

Compressed Trace Size



Compressed communication trace size in log scale (KB).

Intra-Process Compression Overhead

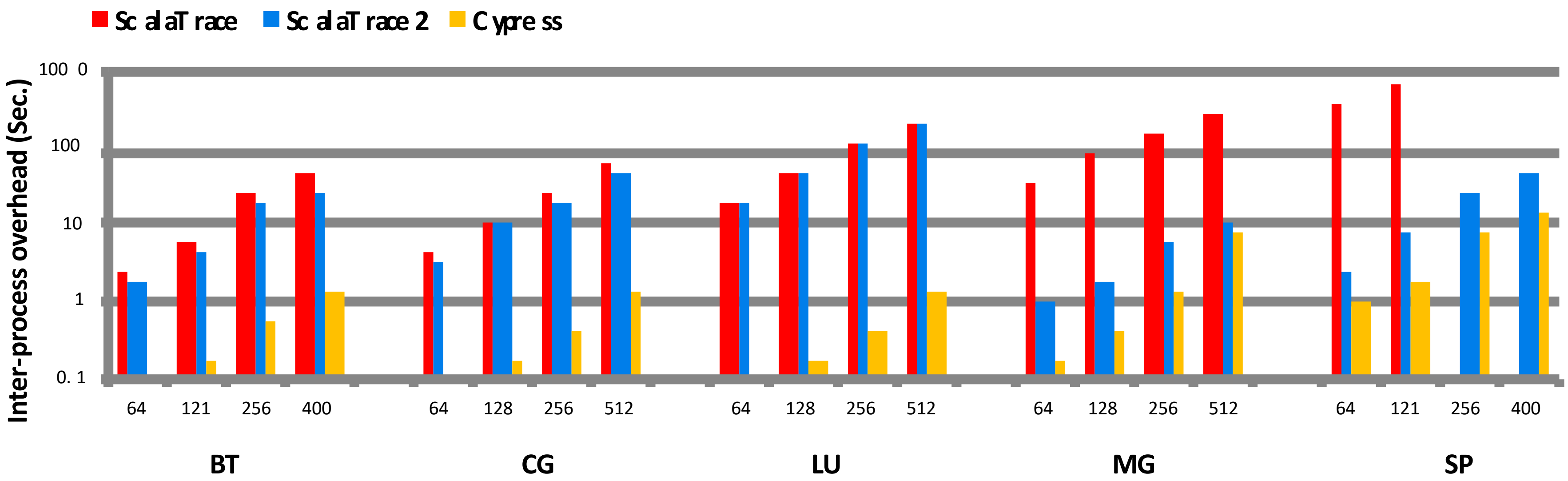


ScalaTrace: 51.05%

ScalaTrace-2: 9.1%

CYPRESS: 1.58% ↓5X over ScalaTrace-2

Inter-Process Compression Overhead



Inter-process trace compression overhead in log-scale (second)

Average inter-process compression overhead:

ScalaTrace: **170.69%**

ScalaTrace-2: **30.3%**

CYPRESS: **3.29%** **↓9X over ScalaTrace-2**

Overall Compression Overhead

- Overhead breakdown

	Dynamic Phase		Static Phase
	Intra-Process	Inter-Process	
ScalaTrace	51.05%	170.69%	N.A.
ScalaTrace-2	9.1%	30.3%	N.A.
CYPRESS	1.58%	3.29%	8.27%

Programs	BT	CG	DT	EP	FT	LU	MG	SP
Compilation Overhead (Sec.)	0.25	0.11	0.05	0.09	0.05	0.16	0.09	0.11

Extra compilation time incurred by Cypress for each program (second)

Conclusion

- CYPRESS: Top-Down trace compression
  - Leverage static information to facilitate more effective and efficient trace compression
  - Reduce compression overhead
    - Intra-process by 5X, inter-process by 9X
  - Strong lossless compression performance
- Take-away Message: static information useful
  - Provides reliable insight
  - Relatively cheap, with scale-independent cost
- Analyze memory access patterns:
  - Spindle: Informed Memory Access Monitoring. USENIX ATC 2018.

Thanks!

Contributors: Jianfei Hu, Liwei Chen, Jidong Zhai, XiongchaoTang, Xiaosong Ma, Wenguang Chen

清華大學
Tsinghua University

معهد قطر لبحوث الحوسبة
Qatar Computing Research Institute
Member of Qatar Foundation عضو في مؤسسة قطر