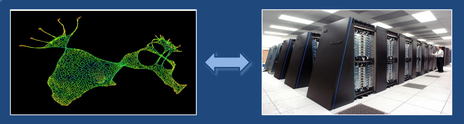
Communication-Centric Optimizations by Dynamically Detecting Collective Operations

Timo Schneider and Torsten Hoefler
University of Illinois at Urbana-Champaign



Users express collectives with p2p-messages :
- Collective not supported by the language
- Slower than hand-tuned on some machine

Tuned collectives cannot be leveraged!

```
if (thisimage() .eq. iimage) then
  ibuf(1:n) = ii(1: n)
  call sync all()
else
  call sync all()
  ii(1:n) = ibuf(1:n)[iimage]
endif
call sync_all()
```

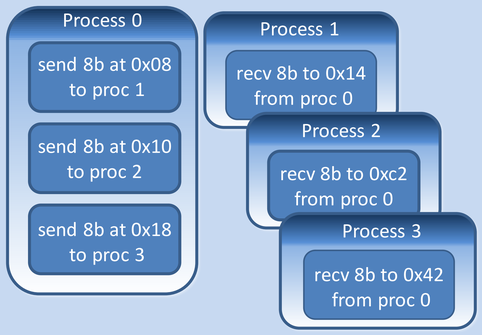
Compile

Compiler transforms this into GOAL code:
- Pattern expressed as dependency graph
- Vertices: Send- / Recv operations
- Edges: Dependencies between operations
- Optimization applied in GOAL_Compile()

```
call GOAL_Create(g)
if (thisimage() .eq. iimage) then
  ibuf(1:n) = ii(1:n)
  do dst=0, num_procs-1
    if (dst .ne. iimage) then
      GOAL_Send(g,ibuf,n*8,dst)
    endif
  end do
else
  call GOAL_Recv(g,ii,n*8,iimage)
endif
call GOAL_Compile()
```

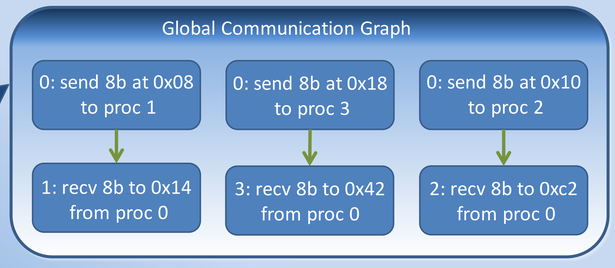
Build Local Graphs

GOAL_Compile() creates a local communication graph for each process
- At runtime, buffer addresses are available
- Note that there are no dependencies in this example

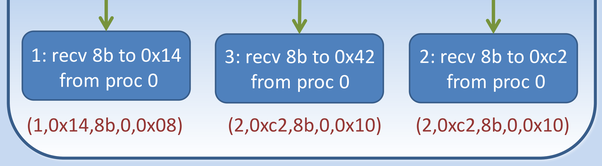


Build Global Graph

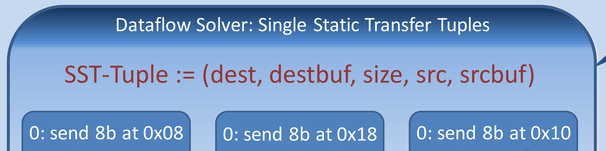
Most optimizations require knowledge of the global communication graph:
- Local graphs are gathered
- Dependencies stay intact as they are process local
- Send and receive operations are linked together (green arrows) in a matching step



Dataflow Analysis

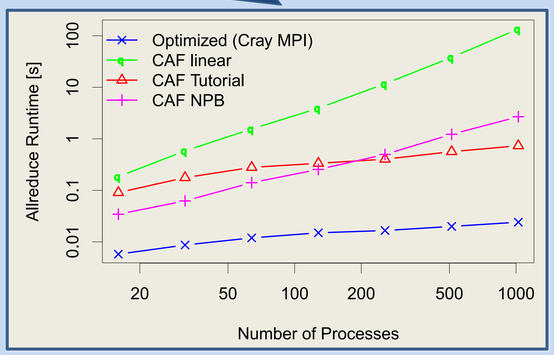


Detect Collectives

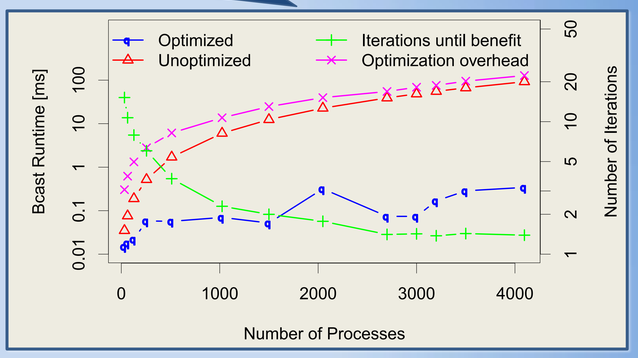


Rewrite with optimized Collectives

Performance of different published allreduce implementations in CAF
- The optimized version, where allreduce is detected and replaced by an MPI call is an order of magnitude faster

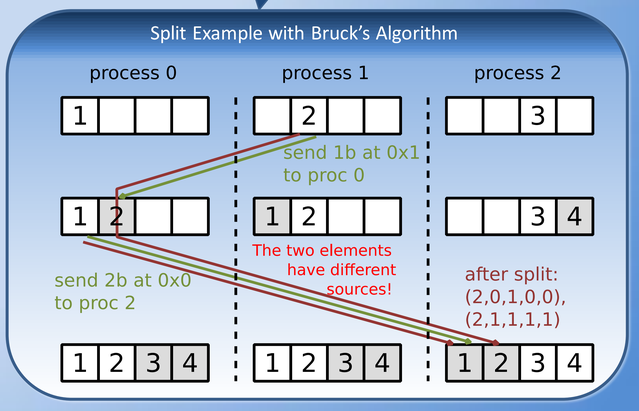


Performance benefits of our optimization:
- Runtime of the optimization is linear - building the global graph requires a Gatherv operation
- The optimization overhead is amortized after few calls of the optimized collective



A collective operation can be described by the set of SST tuples it consists of
- Can be used to match tuples to collectives
- Collectives have to be matched in the order of their expressiveness

Global graph is used solve the dataflow
- Dataflow is expressed in SST Tuples (cf. Single Static Assignment)
- SST is created by visiting the graph top to bottom
- Tuples can be split or merged



ILLINOIS UNIVERSITY OF ILLINOIS AT URBANA-CHAMPAIGN
Timo Schneider <timos@illinois.edu>
Torsten Hoefler <htor@illinois.edu>

Funded by DOE ASCR X-Stack, program manager Sonja Sachs
U.S. DEPARTMENT OF ENERGY | Office of Science