Piggybacking

- Allow a user/tool to add a piggyback value to every MPI message
- Use PMPI interface to intercept send calls
- How to add a piggyback message?
  - New API calls e.g.:
    - MPI_Send_oink(..., pb_buf, pb_dt, pb_count)
    - MPI_Recv_oink(..., pb_buf, pb_dt, pb_count)
  - Use MPI datatypes

Using MPI Datatypes

- Steps:
  - Create a struct of:
    - piggyback message (pb_buf, pb_dtype, pb_count)
    - user message (user_buf, user_dtype, user_count)
  - Send as a normal message
  - Free datatype
- What's the overhead of creating and freeing the struct?
- Optimization? create proto datatype

MPI Datatype Example (thanks Torsten)

```
MPI_Aint offsets[2];
int counts[2];
MPI_Datatype types[2];
MPI_Datatype mine;
offsets[0] = pb_buf;       offsets[1] = user_buf;
types[0]   = pb_dtype;     types[1]   = user_dtype;
counts[0] = pb_count;      counts[1] = user_count;
PMPI_Type_create_struct(2, counts, offsets, types, &mine);
PMPI_Type_commit(&mine);
PMPI_Send(MPI_BOTTOM, 1, mine, dest, tag, comm);
PMPI_Type_free(&mine);
```

Proto Datatype Method

- “Pre-create” as much as we can before hand
- Create a two element struct
  - Specify piggyback dtype and count, but not buffer
  - Don't specify user buffer, dtype or count
- For each message to be piggybacked, complete the datatype
  - Fill in
    - bp_buf
    - user_buf
    - user_dtype
    - user_count

Proto Datatype Example (pt 1)

Once to set up datatype:

MPI_Aint offsets[2];

int counts[2];

MPI_Datatype types[2];

MPI_Datatype mine;

offsets[0] = **MPI_UNKNOWN**; offsets[1] = **MPI_UNKNOWN**;

types[0] = pb_dtype; types[1] = **MPI_TYPE_UNKNOWN**;

counts[0] = pb_count; counts[1] = **MPI_CNT_UNKNOWN**;

MPI_Type_create_pstruct(2, counts, offsets, types, &mine);

MPI_Type_commit(&mine);

Proto Datatype Example (pt 2)

For each message to be piggybacked:

MPI_Datatype mine_complete;

offsets[0] = pb_buf;

offsets[1] = user_buf;

types[1] = user_dtype;

counts[1] = user_count;

PMPI_Type_complete_pstruct(2, counts, offsets, types, mine, &mine_complete);

PMPI_Send(MPI_BOTTOM, 1, mine_complete, dest, tag, comm);

PMPI_Type_free(&mine_complete);

Results for Datatype Method

- Create struct with *n* fields and send. Data sent is 32 bytes in each case (latency in usec)

Num Fields	MPICH2	Open MPI
contig	0.660	0.949
2	1.944	1.520
4	2.055	1.692
8	2.475	1.975
16	3.254	2.676
32	4.743	3.802

- Overhead for piggybacking by creating a struct for every message is **1.28us** for MPICH2 and **0.57us** for Open MPI