Invading instead of Wasting HPC resources

Michael Gerndt
Technische Universität München

10th VI-HPS Anniversary, Frankfurt
Invasive Resource Management

Dynamic
- Applications
- Capabilities
- Resources

Static Resource Management
Scenarios:

• Phase-based applications
• Adaptive applications
• Jobs with different input data set
• Analytics applications with data bursts
• Coupled applications with dynamically varying resource requirements
• IO intensive phases require network and IO bandwidth
• Node failures
• Urgent computing
• Power stability
• Increased scheduling opportunities
• …
InvasIC: D3 Subproject

Invasive HPC applications

Invasive resource management for HPC

Invasive X10 applications on MPSoCs
Implementation

• **MPI extensions**
  • Extend the API with adaptive operations

• **MPI Library**
  • Based on MPICH 3.2

• **Resource Manager**
  • Based on SLURM 15.08
Proposed 4 new operations as an extension to the MPI standard:

**MPI_Init_adapt(...)**
- Initializes the library in adaptive mode

**MPI_Probe_adapt(...)**
- Probes the resource manager for adaptations

**MPI_Comm_adapt_begin(...)**
- Marks the beginning of an adaptation window
- Provides inter communicator and new communicator

**MPI_Comm_adapt_commit(...)**
- Marks the end of an adaptation window
- Sets adapted MPI_COMM_WORLD

---

**Code Structure**

```c
MPI_Init_adapt(..., &status);
for (...){
    MPI_Probe_adapt(&adapt, ...);
    if (adapt){
        MPI_Comm_adapt_begin(...);
        // redistribution code
        MPI_Comm_adapt_commit(...);
    }
    // compute and MPI code
}
```
Adaptation Step 1

1: Reallocation Message

New Adapted Allocation

Preexisting Allocation

Expansion Allocation
Adaptation Step 2

1: Reallocation Message

2: Create New Processes in Expansion Nodes

New Adapted Allocation

Preexisting Allocation

Expansion Allocation
Adaptation Step 4

1: Reallocation Message

2: Create New Processes in Expansion Nodes

3: New Processes Ready

4: Notify Preexisting Processes

New Adapted Allocation

Preexisting Allocation

Expansion Allocation

SRUN

SLURMCTLD

Scheduler Plugin

Node

MPI Process

10
Adaptation Step 5

1: Reallocation Message

5: Adaptation Commit

4: Notify Preexisting Processes

3: New Processes Ready

2: Create New Processes in Expansion Nodes

New Adapted Allocation
Preexisting Allocation

Expansion Allocation
SuperMUC @ LRZ

- Allocation of set of SuperMUC nodes via batch job
- Management of the nodes via separate SLURM instance
- Distribution of resource management into
  - SLURM Scheduler
    - Selection and scheduling of invasive jobs
    - Based on resource offer
  - Invasive Scheduler
    - Invasive resource management
- Submission of new invasive jobs through sbatch command
Tsunami simulations (Mo-Hellenbrand, Bungartz)

- Static MPI
  - Integral area = 102512
  - CPU-Hour = 28.48

- Elastic MPI
  - Integral area = 57215
  - CPU-Hour = 15.89
Adaptation Visualisation

Timeline:

- RM:0
- Node:1
- Node:2
- Node:3
- Node:4

Function Summary:

- All Processes, Accumulated Exclusive Time per Function
  - Job 78 [405 s (60.54%)]
  - Job 79 [238 s (35.58%)]

- RM: 405 s (47.04%), 238 s (27.68%), 192 s (22.3%), 24 s (2.79%), 2 s (0.23%)
Periscope Tuning Framework

VI-HPS, AutoTune, READEX

- **Periscope Frontend**
  - Controls the analysis and tuning process
  - Performs a sequence of experiments
    - While the application is executing
    - Based on application phases
    - Automatically starting/restarting the application if required

- **Agent Network for scalability**
  - Leave agents responsible for a subset of the MPI processes
  - Intermediate agents aggregate performance properties

- **Online Access to the monitoring system**
  - Configuration of measurements and tuning actions
  - Retrieval of performance data
Thank You