

Converting System-Level Checkpoints of HPC Applications for their Simulation and Verification

Millian Poquet

Advisor: Martin Quinson

millian.poquet@inria.fr



Context and Motivation

Big picture

- Target: Distributed HPC apps (MPI)
- Goal: Improve study of performance/correctness

HPC apps execution

- Many resources
- Much time (weeks...)
- Resilient models checkpoints

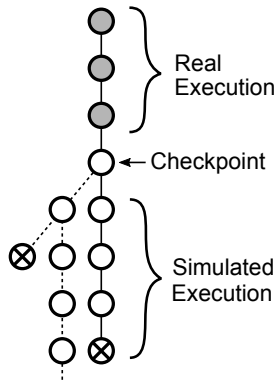
Proposition: Simulation from Checkpoints

Simulation

- Fast and cheap
- Deterministic (Heisenbugs), clairvoyance...

Start from checkpoint

- Only study desired part
- MC: huge exploration space cut



Outline

1 Introduction

2 Software Involved

3 Main Difficulty

4 Conclusion

Software Overview

SimGrid: Distributed System Simulator

- Model checking
- Very credible
 - Validated performance models
 - Tested implementation
 - Sustained effort since ≈ 2002
- LOC: $\approx 150k$ C/C++



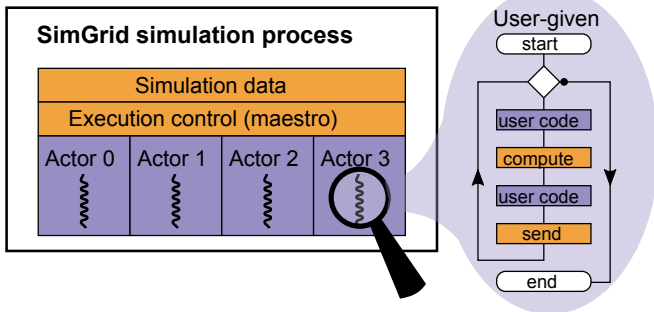
DMTCP: Distributed MultiThreaded CheckPointing

- Checkpoint/restart any distributed app
- User-space
- Sustained effort since ≈ 2007
- LOC: $\approx 40k$ C/C++, assembly

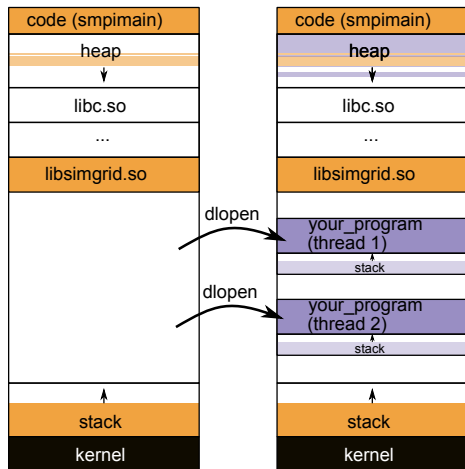
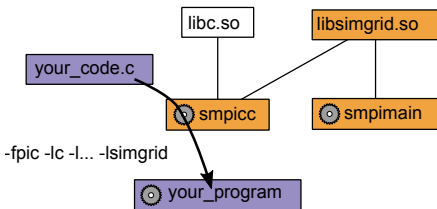
SimGrid: Execution Overview

Essentially a library. Architected as an OS.

- 1 address space (**kernel** + **user code**)
- mutual exclusion on actors' execution
- *maestro* dictates who run



SimGrid: SMPI Execution



Initial state

User code loaded




DMTCP: Overview

Essentially a set of programs + some internal libs

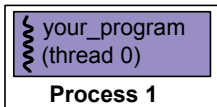
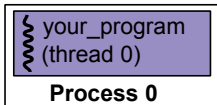
Three main operations

- Initial launch
- Do checkpoint
- Restart from checkpoint

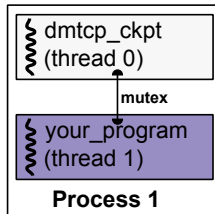
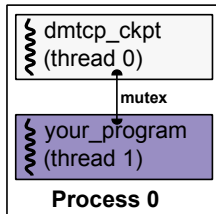
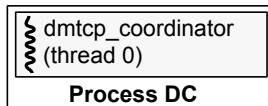
DMTCP: Launch

 your_program your_runner dmtcp_launch

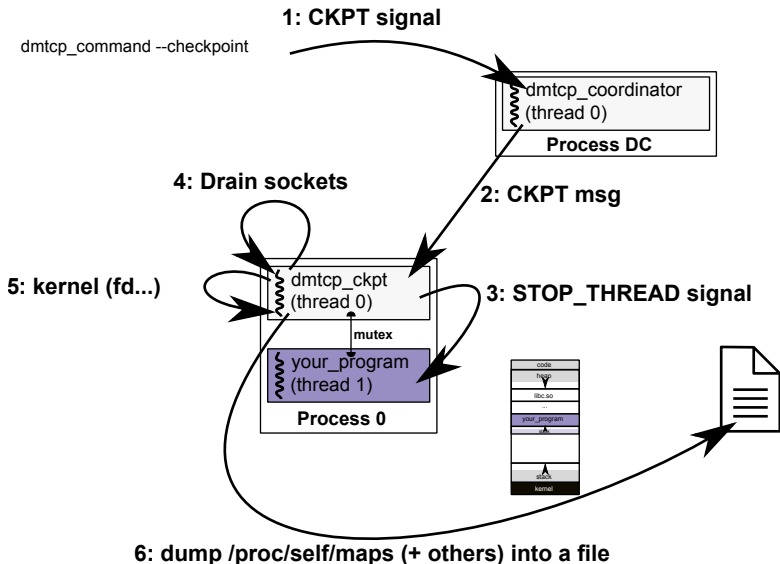
your_runner your_program [...]






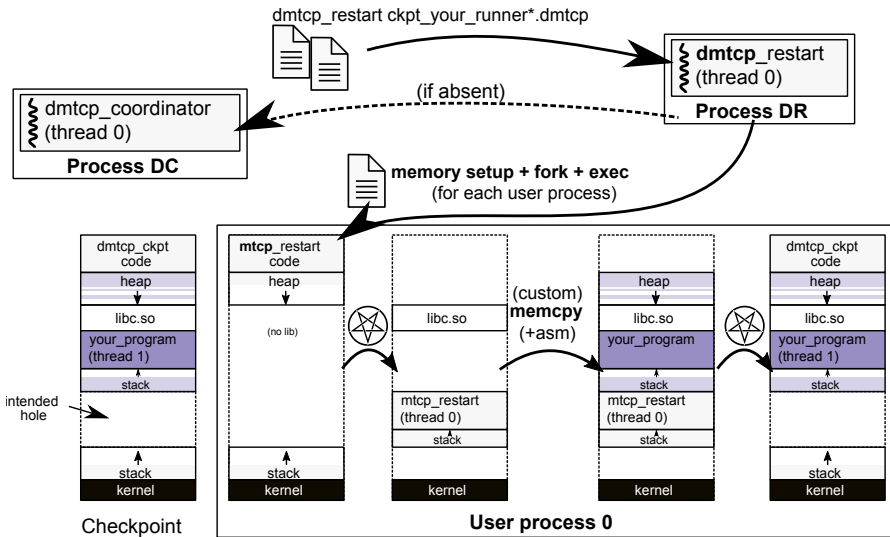
dmtcp_launch your_runner your_program [...]

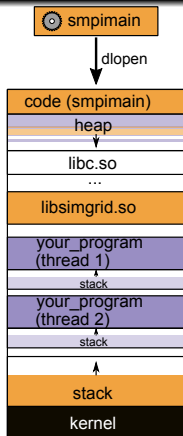
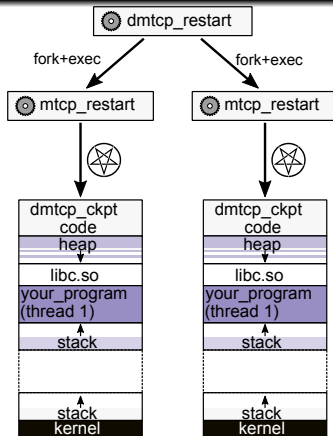


DMTCP: Checkpoint



(-fpic)

 your_program mtcp_restart dmtcp_restart

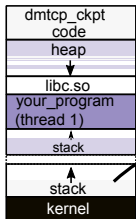
**SMPI's One Process Architecture****DMTCP's Distributed Architecture**

How to match them?

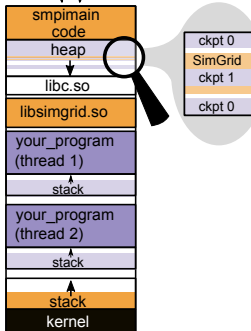
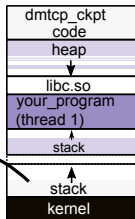
- 1 Somehow load checkpoints into a single SMPI process
- 2 Somehow use SimGrid in a DMTCP-restarted execution

Approach 1: *One Process* Architecture

Checkpoint 0



Checkpoint 1



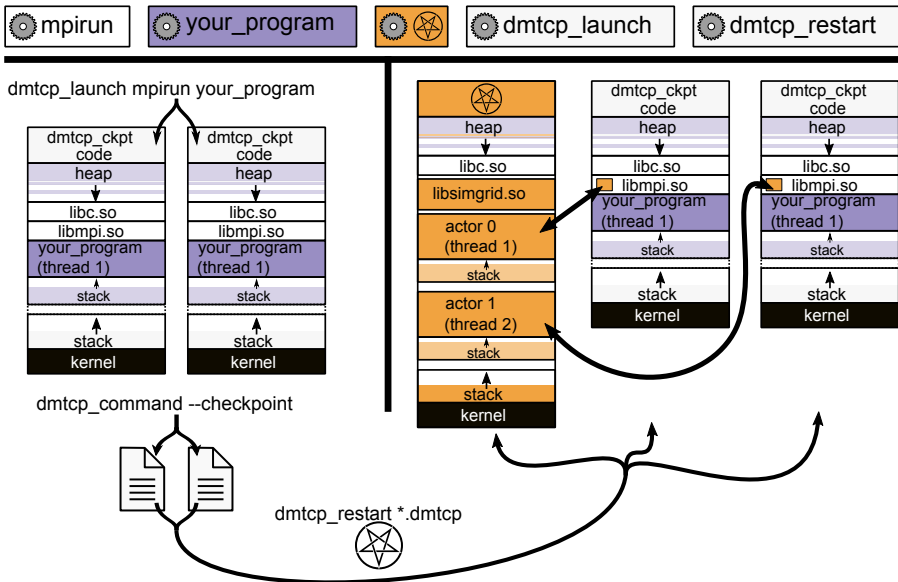
Black magic

- Malloc (code, stack)
- Fix heap collisions ☹️ ☹️
- Fix libc collisions ☹️
- Hack MPI implementation ☹️
- Fix kernel state (fd...) ☹️

So...

- Reimplement/improve DMTCP
- Fix heap collisions ☹️ ☹️

Approach 2: *Several Processes* Architecture



DMTCP plugins

A plugin is responsible for modelling an external subsystem, and then creating a semantically equivalent construct at the time of restart.
(Gene Cooperman)

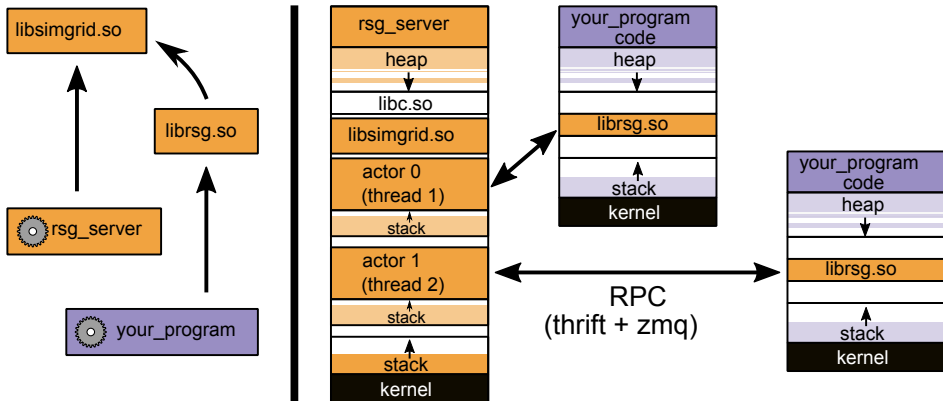
How to apply this for MPI?

- Do not checkpoint MPI implementation internals (lib memory, misc. processes)
- Store anything that may alter internal state

In brief

- 1 During execution: Store sequence of *troublesome* MPI routines
- 2 At checkpoint time: Flush network buffers
- 3 At restart time: Execute *troublesome* MPI routines in order

Remote SimGrid



OpenMPI



In brief

- Joined effort since ≈ 2004
- **very** modular
- LOC: $\approx 600k$ C + misc.

Plan

- Network layer: RSG
- App launch layer: +rsg_server
(+ clients ENV)

Conclusion

Problem: Incompatible memory loading model

- Approach 1: Load ckpts from SMPI
- Approach 2: Distributed SG + restart injection

Distributed arch seems more reasonable

- Separate parts are contributions by themselves
- Better SoC → maintainability

Big picture

- 1 Distributed simulated MPI implem (RSG + OpenMPI)
- 2 Restart checkpoint on another MPI implem (DMTCP plugin)
- 3 Extend SimGrid's MC