CONVERSE :
An Interoperable Framework for Parallel Programming

Sanjay Kale, Milind Bhandarkar,

Narain Jagathesan, Sanjeev Krishnan

Parallel Programming Laboratory

Department of Computer Science

University of Illinois, Urbana.

Motivation

- many different parallel programming paradigms
  - SPMD
  - Message driven objects (MDO)
  - Threads and Shared Memory
  - Data parallel
  - Functional
  - Logic
- each is suited for different types of applications
- each provides different run-time primitives
  - SPMD : tag based messages
  - MDO : asynchronous remote method invocation

Motivation

- a single “best” paradigm does not exist
- different paradigms have different advantages

We should be able to :

- express different modules in a large application in different paradigms
- combine pre-written modules from different paradigms

INTEROPERABILITY HELPS MODULARITY AND REUSE

CONVERSE Design

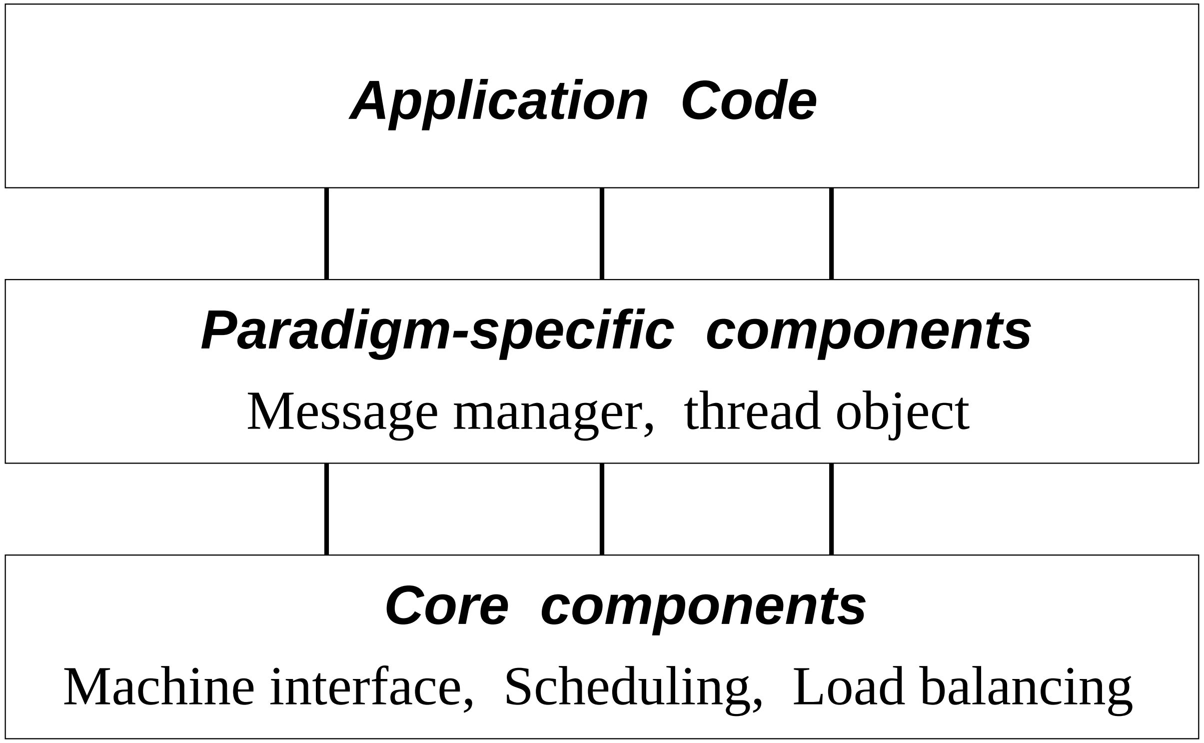
Desirables :

- Completeness : should support most paradigms
- Need based cost : each paradigm should incur cost only for its own features
- Efficiency : should be comparable to native (non-interoperable) implementation

Component based framework :

- each component is specified by concrete interface
- each component can be implemented in different ways

CONVERSE Architecture



CONVERSE Machine Interface

Support messaging for SPMD, objects, threads

- Handler invoked on message delivery : no tag matching/ordering overhead - handler function/id sent with message - optional "GrabMessageBuffer" for preventing copying where possible
- Synchronous send : message buffer reusable after send
- Asynchronous send : no waiting for message to leave processor, no copying
- Atomic terminal I/O, timers, etc

CONVERSE Machine Interface

Extended Machine Interface : generic implementations, can be specialized for individual machines.

- Global pointers (Create / Get / Put)
- Interrupt (active) messages
- General Gather / Scatter
- Processor Groups
- Parallel I/O

Control Paradigms

Focus on SPMD, MDO, Threads

Two characteristics :

- Concurrency within a process (processor)
  - No concurrency : SPMD
  - Concurrency via objects / threads
- Control regime
  - Explicit, static : SPMD
  - Implicit, adaptive : MDO, threads

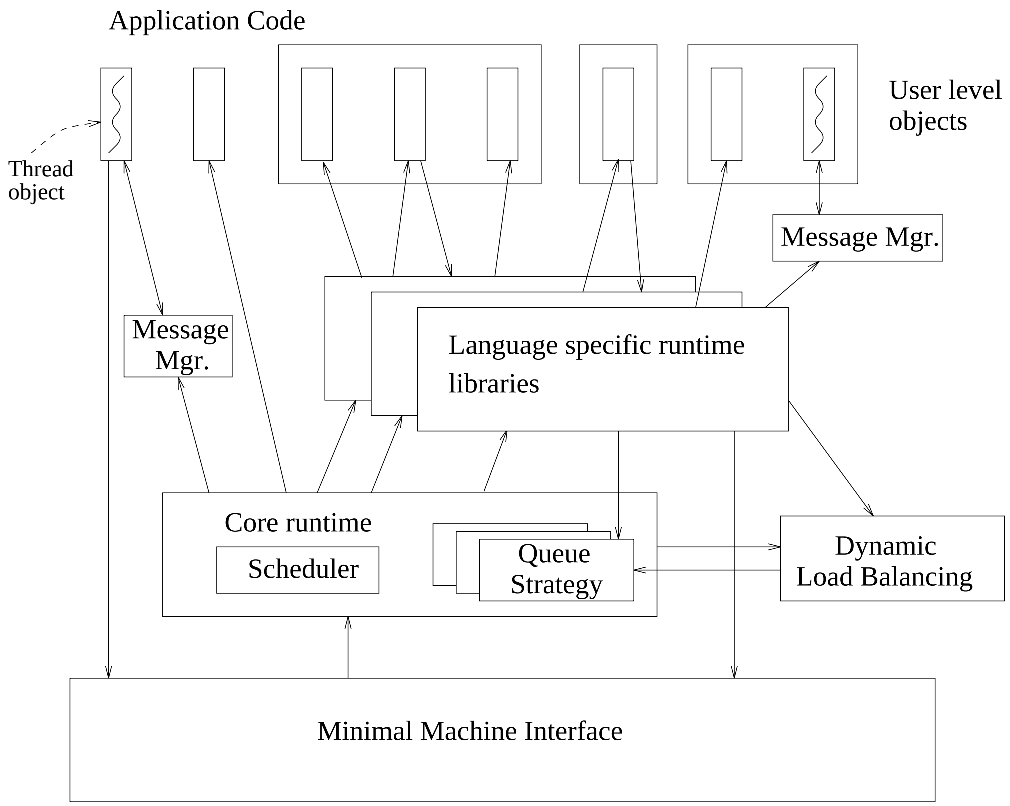
CONVERSE Scheduler

For implicit control regimes

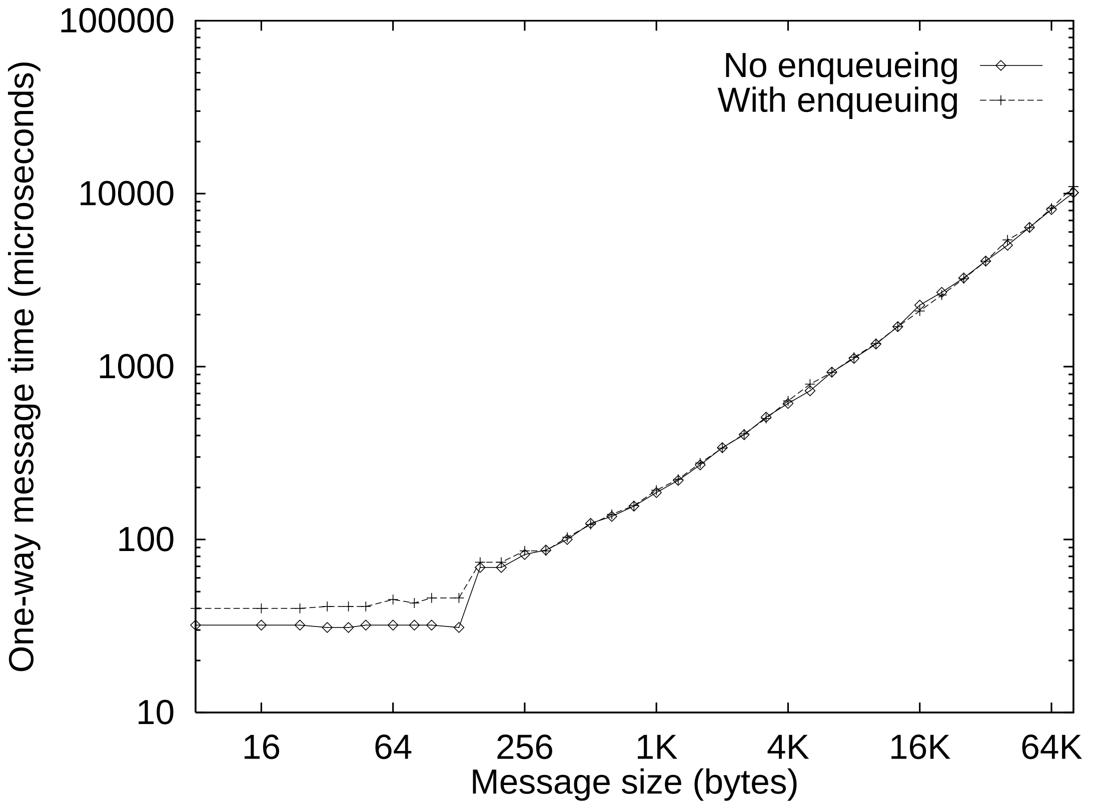
- Used by MDO / Thread based languages when they need it
- Generalized handler-based messages
- Scheduler maintains a single message queue
- Plug in different queue implementations e.g. fifo, prioritized

```
Scheduler()
{    while ( not done ) {
         msg = GetMsg() ;  // From network or queue
         (HandlerOf(msg))(msg) ; }
}
```

CONVERSE Architecture



CONVERSE Messaging, Scheduling performance



Related work

- Interoperability within same paradigm: HPF-MPI
- Nexus:
- PORTS: Portable Run-Time System (working group)
- Chant: message passing threads
- tPVM : threaded PVM
- Messaging layers: FM, Active messages, ...
- HPC++ : data parallel, threads, objects