

Execution Environments for Parallel Applications

Master CANS 2007/2008
Departament d'Arquitectura de Computadors
Universitat Politècnica de Catalunya



Dept. d'Arquitectura de Computadors

Entornos Operativos para la Gestión de Recursos...



Outline

- Supercomputers
- OS abstractions
- Tools for performance analysis
- Extended OS interfaces
 - IRIX sysmp
 - IRIX memory placement
 - Linux interface
- Example: the CPU Manager
- Conclusions



Dept. d'Arquitectura de Computadors

Entornos Operativos para la Gestión de Recursos...



Memory Management in NUMA Systems

IRIX & Linux



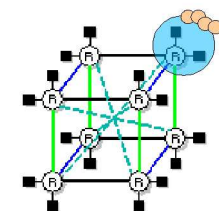
Dept. d'Arquitectura de Computadors

Entornos Operativos para la Gestión de Recursos...



Memory management interface

- IRIX
 - Memory locality domains (MLD's)
 - Policy modules
 - Initial allocation
 - Placement policy
 - Page size policy
 - Dynamic relocation
 - Migration policy
 - Replication policy
 - Paging
 - Paging policy



With Xpress Links



Dept. d'Arquitectura de Computadors

Entornos Operativos para la Gestión de Recursos...



Memory management interface

Placement policy

- Default (number of threads)
- Fixed (mld)
- FirstTouch ()
- RoundRobin (mldset)
- ThreadLocal (mldset)
- CacheColor (mld)

Migration policy

- Default ()
- Control (arguments)
- RefCnt



Memory management interface

PlacementDefault (number of threads)

- Automatic creation of an MLD foreach two threads
 - Affinity link
- Memory allocated using the affinity link MLD

Placement Fixed (mld)

- Memory allocation in the node where the mld has been placed

PlacementFirstTouch

PlacementThreadLocal (mldset)

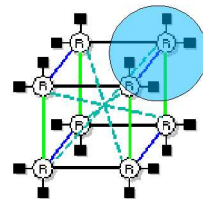
PlacementCacheColor (mld)



Memory management interface

Creating an MLD

- mld[i] = mld_create (radius, page_size)
 - Creates a memory locality domain
- Radius indicates the maximum distance of the memory
- page_size indicates the suggested page size
 - 16Kbytes to 16 Mbytes
- Returns a pmo_handle_t
 - Similar to a file descriptor



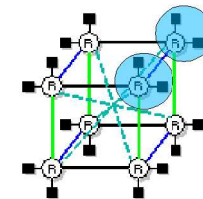
With Xpress Links



Memory management interface

Creating an MLDset

- mldset = mldset_create (mlds, number_of_mlds)
 - Creates a MLDset
 - Makes a set containing all the mlds



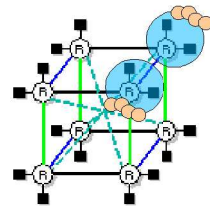
With Xpress Links



Memory management interface

Creating an MLDset

- `mldset = mldset_create (mlds, number_of_mlds)`
 - Creates a MLDset
 - Makes a set containing all the mlds



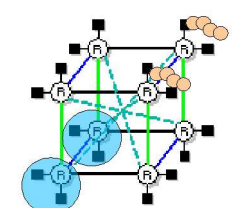
With Xpress Links



Memory management interface

Creating an MLDset

- `mldset = mldset_create (mlds, number_of_mlds)`
 - Creates a MLDset
 - Makes a set containing all the mlds



With Xpress Links



Memory management interface

Placing an MLDset

- `mldset_place (mldset, topology, affinity, size, rqmode)`
- Places the mldset according to topology and affinity
- Topologies
 - FREE
 - CUBE, CUBE_FIXED
 - PHYSNODES, affinity
- Affinity, array of affinity information foreach mld
 - `/hw/cpunum/N`
 - radius
 - ATTRACTION / REPULSION
- Size of the affinity array
- Rqmode, ADVISORY, MANDATORY



Memory management interface

Creating a policy module

- `polmod = pm_create (policy_characteristics)`
 - PlacementFixed
 - MigrationControl (corresponding mld)
- Returns a `pmo_handle_t`
- Characteristics
 - Initial allocation
 - Placement
 - Page size
 - Fallback, relative importance between placement and pagesize
 - Dynamic relocation
 - Migration
 - Replication
- Paging



Memory management interface

□ Attaching memory to a policy module

- pm_attach (polmod, address, size)
 - Connects a virtual address range to a policy module
- After memory allocation
- When physical memory is needed, it is allocated close to where the policy module is placed



Memory management interface

□ Linking processes to MLDs

- process_mldlink (mypid, mlds[threadnumber/2], RQMODE_ADVISORY)
 - The process scheduler will try to activate the process on a CPU in the node where the MLD has been placed
 - /2 because of 2 cpus (or threads) per "node"
- process_cpulink (MUSTRUN)



Linux interface

□ No specific system interface for process management

- pause / signal / kill

□ (2002) Can be easily added to last versions of the kernel

- 2.4.18-X
- 2.4.19

□ Currently, processor binding is possible

- sched_setaffinity



Linux interface

□ Extensions for binding

- Based on the cpus_allowed bit-mask
 - Maintained in a per-clone basis
- The scheduler on a cpu only can get a process allowed to run on that cpu
- Mechanism implemented initially for internal processes only
- Currently available for regular processes
 - sched_get/setaffinity (pid, len, mask);
- Works automatically for user processes



Linux interface

General purpose processor and memory placement

- Common Linux kernel mechanism to support the implementation of various placement policies
 - Including emulations of existing API's
 - Single kernel code
 - Readability
 - Performance
 - Normal and large systems



Linux interface

Layers

- cpumemmap - bottom layer
 - describes the machine resources
 - cpus
 - pid 22516 running on cpu 38 (in node 19)
 - cpus in map 80
 - 0 1 2 ... 79
 - memory
 - mems in map 40
 - 0 1 ... 39
 - cmsQueryCMM ();
 - cpus in set 4, policy 1
 - 36
 - 37
 - 38
 - 39
- cpumemset - upper layer
 - describes the resources available to the application
 - cmsQueryCMS ();
 - mems in set 1
 - mem 0 mems 2
 - 18
 - 19
 - Information may be changed for
 - Current process
 - New child processes
 - Vm_area of given virtual address
 - Kernel



Linux interface

Getting information

- numcpus();
- numnodes();
- cmsGetCpu();

Changing the mappings

- cmsSetCMM(...);
- cmsSetCMS(...);

Binding to processors

- runon (cpu);
- cpubind (cpu);

Binding to processors and memory

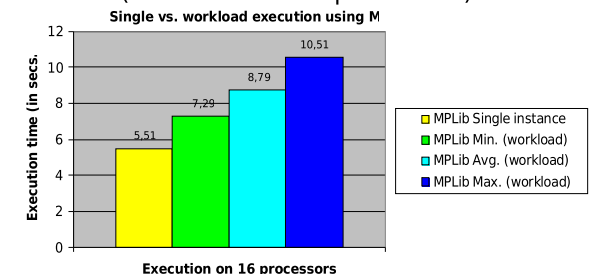
- nodebind (node);



Example: workload execution

Workload execution

- 64-processor SGI Origin2000
- SPEC95 FP SWIM (5 instances x 16 processors)



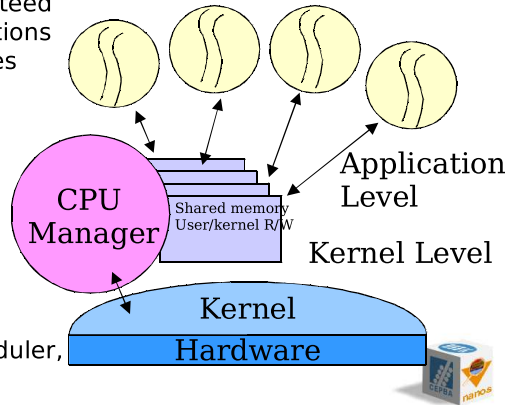
- Poor performance due to conflicts in resource sharing
- No coordination between running applications and the kernel
- No global load control



Example: workload execution

■ NANOS MP CPU Manager

- Uses the available kernel interface + shared memory
- Protection guaranteed
- Malleable applications
- IRIX native binaries



- USER-LEVEL scheduler, GLOBAL control

Dept. d'Arquitectura de Computadors

Entornos Operativos para la Gestión de Recursos...

Nanos CPU Manager

■ Supports IRIX native binaries

- **No modifications** to source code
- **No recompilation**
- **No relinking**
- Execution through interposition mechanisms

■ Malleable applications

- Adapt through the SGI MP Library sugnumthd mechanisms

■ Non-malleable applications

- Folded to the available number of processors



Dept. d'Arquitectura de Computadors

Entornos Operativos para la Gestión de Recursos...

Functionality

■ CPU Manager functionality

- Gets applications requests
- Applies a scheduling policy
- Computes the number of processors for each application
- Assigns processors
- Informs the applications

■ Contacting the CPU Manager

- Interposition mechanism allows to get control before an application executes
- First contact through a named pipe
- Dialog through shared memory
- Application tracking through /proc filesystem



Dept. d'Arquitectura de Computadors

Entornos Operativos para la Gestión de Recursos...

Implementation

■ Communication

- Read/write to shared memory
- Block/unblock application threads
- Bind application threads to physical processors
 - Improve memory affinity

■ Shared memory contents

- Requested number of processors
- Current number of processors
- Threads identifiers & status



Dept. d'Arquitectura de Computadors

Entornos Operativos para la Gestión de Recursos...

Scheduling Policies

- ❑ **Cluster**
 - ❑ Distributes processors in groups of N
 - ❑ N=1 results in Equipartition
- ❑ **Assignment steps**
 - ❑ Decides how many processors to give to each application
 - ❑ Decides which processors to assign
 - ❑ Supplies the number of processors and binds them
 - ❑ Round-robin



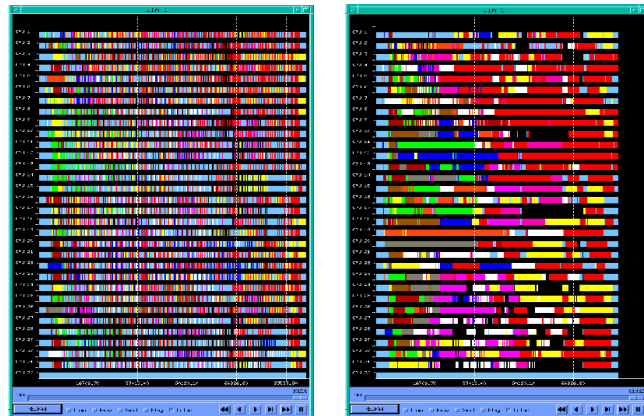
Scpus

- ❑ **Allows tracing of OS view**
 - ❑ Externally to applications
- ❑ **Searches /proc for any useful information**
 - ❑ Mapping of processes/threads to processors
- ❑ **Gets other information from the system**
 - ❑ Processor states
- ❑ **Would be useful to have the process running in each processor as an OS primitive!!!**



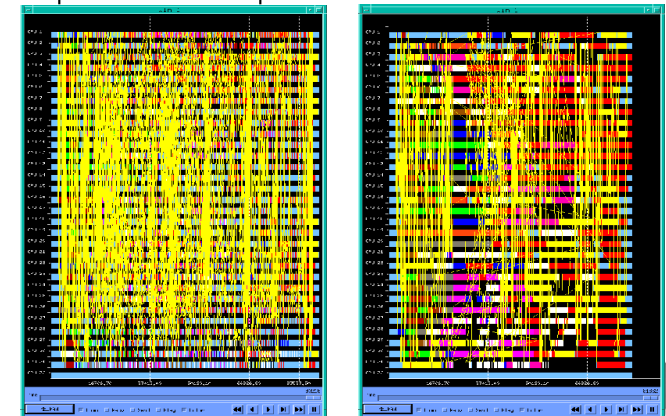
Scpus

- ❑ **Evolution of thread migrations**



Kernel level scheduling

- ❑ **Evolution of thread migrations**
 - ❑ Improvements still possible



Experiments

Results

Application	Instances	Request	Native SGI		MP CPU Manager	
			Exec. time	Std. dev.	Exec. time	Std. dev
ft.A	10	8	29.6	5.3	22.2	5.7
mg.A	10	8	35.7	5.2	22.6	2.9
cg.A	10	4	21.6	1.9	17.1	1.5

- Total execution time reduced from 400 to 264 seconds
 - CPU Manager performs better than local mechanism for load control
- Further improvements with better mapping policies
 - Bad placement for FT (see next two slides)
 - Open environment, modifications easy introduced



Dept. d'Arquitectura de Computadors

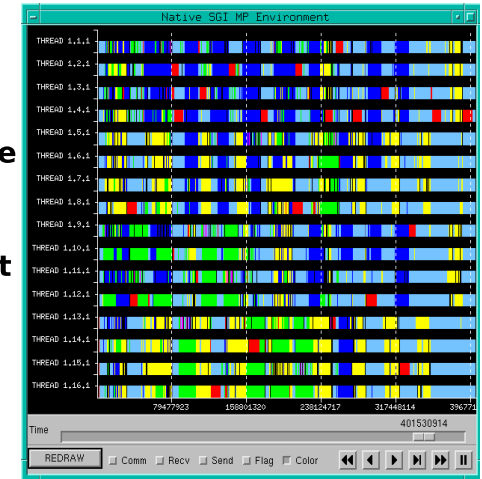
Entornos Operativos para la Gestión de Recursos...



Native SGI MP Library

- Poor coordination
- High number of movements
- Higher execution time in kernel mode
- Worse execution times and throughput

ft.A
 mg.A
 cg.A
 Kernel mode
 Kernel idle



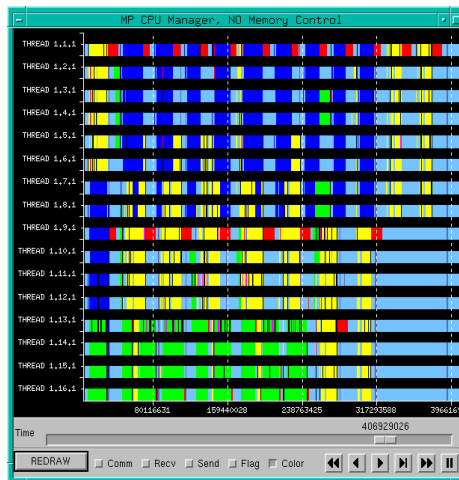
Dept. d'Arquitectura de Computadors

Entornos Operativos para la Gestión de Recursos...

CPU Manager STD Memory Control

- Better coordination
- Low number of movements
- Execution time in kernel mode still high
- High number of remote memory accesses

ft.A
 mg.A
 cg.A
 Kernel mode
 Kernel idle



Dept. d'Arquitectura de Computadors

Entornos Operativos para la Gestión de Recursos...

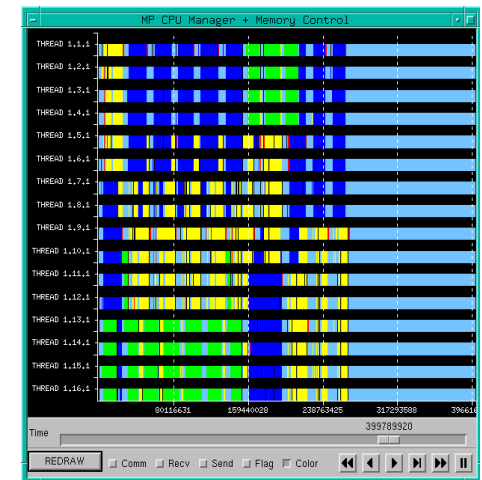
CPU Manager & Memory Control

- Better coordination
- Better processor to cpu mapping
- Consistent execution times
- Improved memory behaviour
 - Local accesses
 - Lower kernel mode execution time



Dept. d'Arquitectura de Computadors

Entornos Operativos para la Gestión de Recursos...



Conclusions

- ❑ **Diverse architectures**
- ❑ **Resource management primitives**
 - ❑ Non standard
 - ❑ Heavily bound to the architecture
- ❑ **Analysis is difficult**
 - ❑ Performance counters
 - ❑ Information from the OS
 - ❑ Visualization tools
- ❑ **Run-time systems must accomodate to each different architecture**
 - ❑ Performance issues

