Microsoft Windows Compute Cluster Server 2003

Evaluation

Georg Hager, Johannes Habich (RRZE)

Stefan Donath (Lehrstuhl für Systemsimulation)

Universität Erlangen-Nürnberg

ZKI AK Supercomputing 25./26.10.2007, GWDG

Outline

- Administrative issues
  - Installing the Head Node
  - Cluster Network Topology
  - RIS-Unattended Installation
  - Domain integration
  - User Environment
  - Benchmarks for Intel MPI PingPong
- Current user projects
- Evaluating Excel integration
  - LINPACK sample
  - Homebrew VBA macros for simple Jacobi benchmark
- NUMA and affinity issues

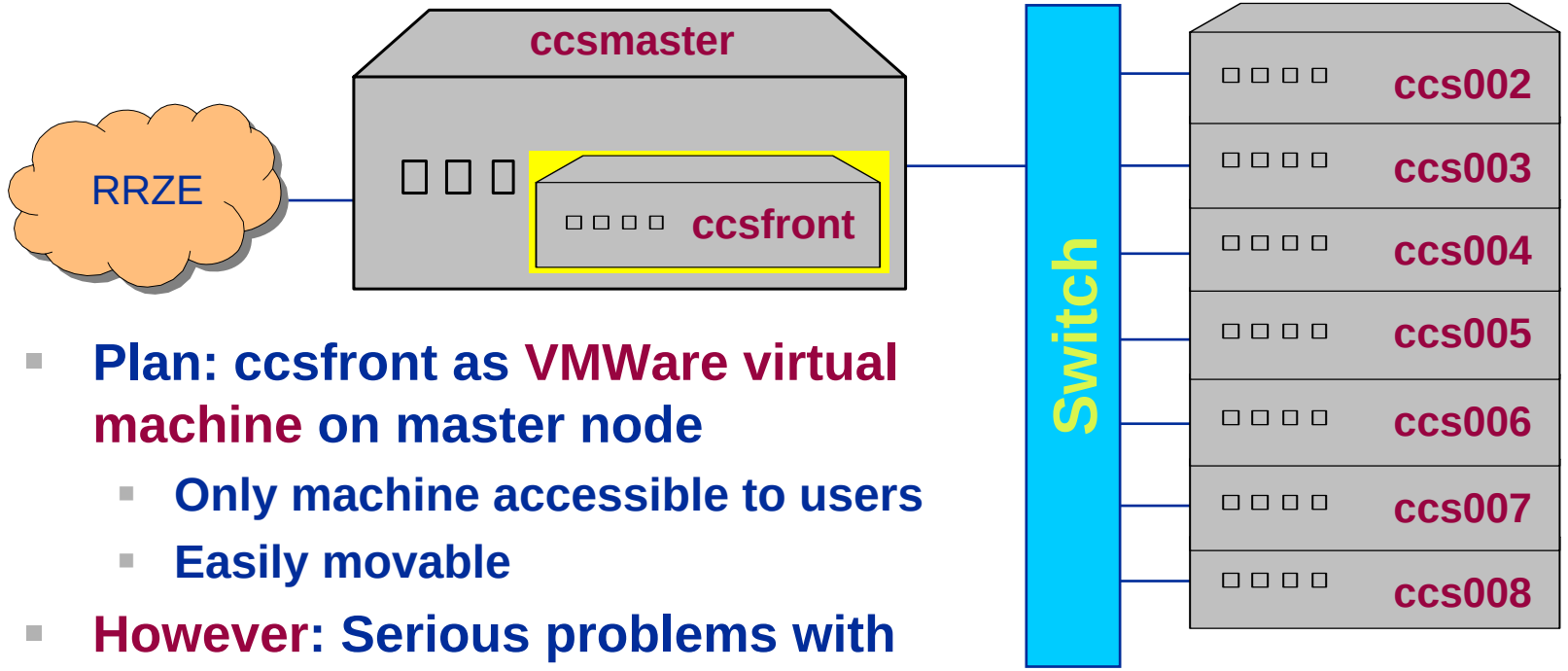
- Conclusions

Opteron Test Platform

- 7 quad Opteron nodes
  (Dual Core Dual Socket)
- 4 GB per node
  8GB on head node
- Windows 2003 Enterprise
  + Compute Cluster Pack
- Visual Studio 2005,
  Intel compilers, MKL, ACML
- Star-CD
- Gbit Ethernet

- Access via RDP or ssh (sshd from Cygwin)
  - GUI tool for job control: Cluster Job Manager
  - CLI: `job.cmd` script

Cluster Layout



- Plan: ccsfront as VMWare virtual machine on master node
  - Only machine accessible to users
  - Easily movable
- However: Serious problems with Visual Studio and Intel compilers inside VM
  - Same procedures work on ccsmaster
  - Being investigated
- For now, people work on ccsmaster

Installing the Head Node

- Preinstalled headnode and cluster nodes from transtec with evaluation version of WinCCS2003
- Preliminary network connection bandwidth evaluation with Intel IMB benchmarks
- No SUA support → Clean installation of Win2003 Enterprise R2 x64

- Installation of Intel Fortran and C compiler with Visual Studio 2005 integration
  - Intel C++ Compiler 9.1 and 10.0
  - Intel Fortran Compiler 9.1
  - Intel MKL 9.0

Cluster Network Topology

- Separate private and public 1GE networks available
- DHCP Server could not separate the two scopes to two physical network adapters
- DHCP Server is reconfigured without warning for Remote Installation Services (RIS) to install nodes unattended

→ only private network was used, with NAT translation for outside communication

RIS Unattended Installation

- Creating the RIS (Remote Installation Server) Image from Win2003 installation CD
- By-hand inlining of R2 necessary packets and settings from Win2003 installation CD 2
- Adjustment of wrong paths inside the configuration files
- First attempt to install RIS caused a complete DHCP breakdown, as RIS changes complete DHCP configuration and launches without proper rights for standard scope 192.0.0.x

- After that, RIS installed compute nodes flawlessly

RIS Unattended Installation

- Domain Administrator rights are necessary for creating automatically new computer accounts inside domain
- RIS deploy wizard warns user that password is visible in plain text during deployment
  → huge security risk during each setup!

- Suggestions:
  - No plain text passwords
  - Easy wizard creation of standard Win2003 images
  - DHCP Server with multiple instances for each network interface
  - Failure messages should be more elaborate

Domain Integration

- Headnode and cluster nodes were integrated into UNI-Erlangen ADS
- Problem of supplying a working directory which is both available as a network share via Windows Explorer and available to the jobs running on the cluster
  - User works on mounted network share
  - Job works on complete UNC path leading to same network share
  - → One common path for both, but UNC for users is not a preferred choice
  - → Automatic mapping of this location at job start and user login
  - → Problems of some Windows software to work with UNC (CMD.exe , partly Visual Studio 2005)

Intel MPI Ping Pong benchmark

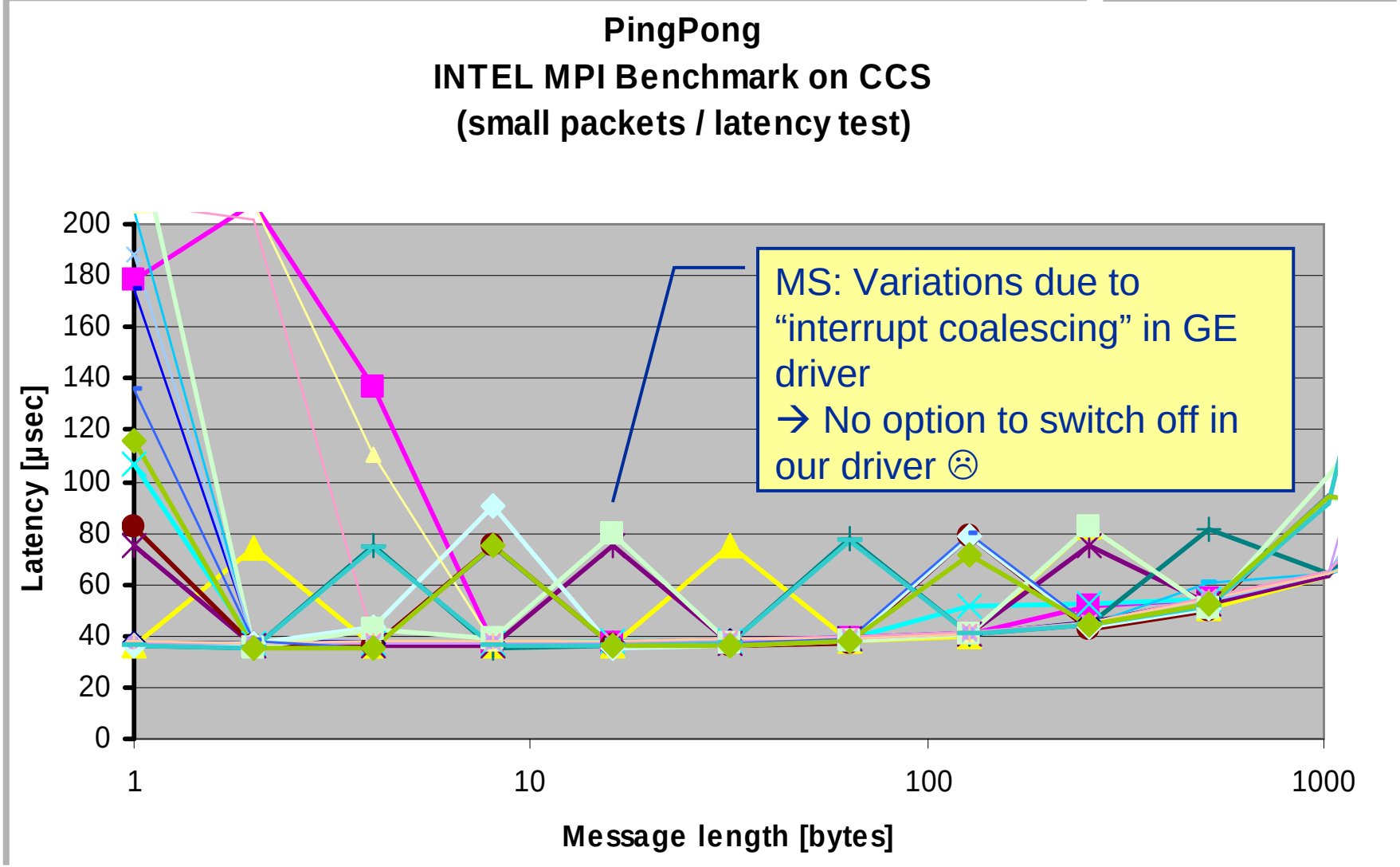
- Compiling inside SUA and Visual Studio 2005
- Issuing jobs with standard MPIEXEC causes 4 threads to run on each node → internode communication measured
- Hence:
  "Fun and interesting ways to run MPI Jobs on CCS"
  from `http://windowshpc.net/`

  Now:
  `pernode.bat` script hacks `%CCP_NODES%` system variable and only one task is executed per compute node

  → Measuring real internode connection bandwidth and latency

  → Desirable: flexible specification for „processors per node" at job submission and runtime

Intel MPI Ping Pong benchmark

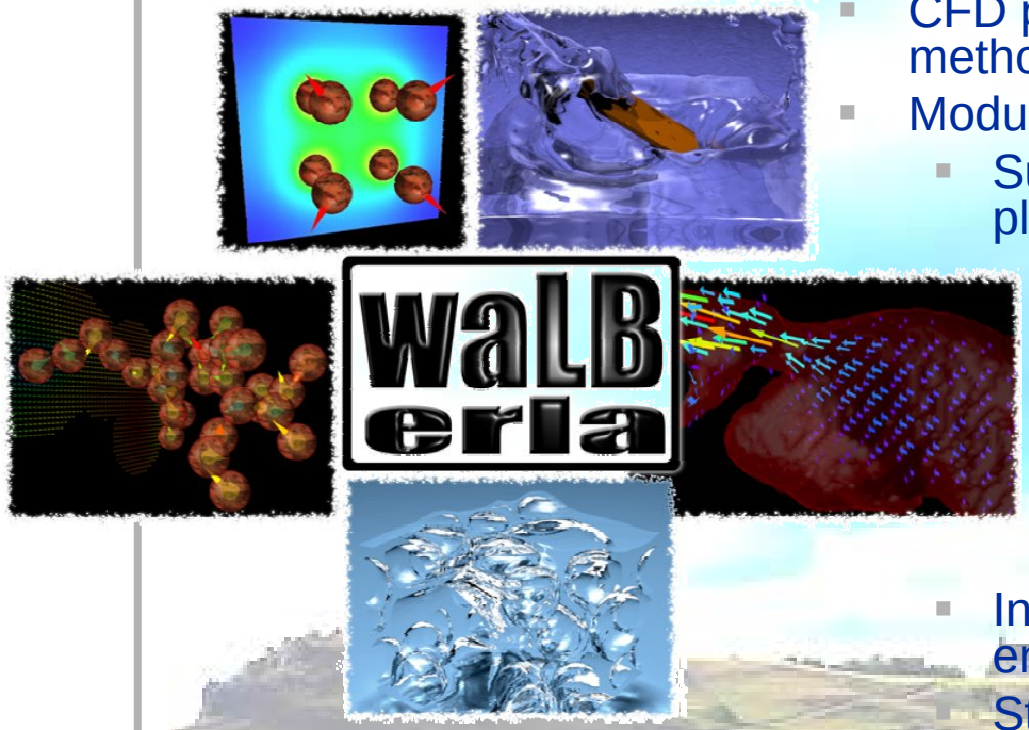


Projects on the WinCCS Cluster

- Crack propagation (FAU, Materials Science)
  - F90/OpenMP, C++ (OpenMP to come)
- VirtualFluids (TU Braunschweig)
  - C++/MPI
- Star-CD (FAU, Fluid Mechanics)
- waLBerla (FAU, System Simulation)
  - C++/MPI

waLBerla project @

Widely applicable lattice Boltzmann from Erlangen



- CFD project based on lattice Boltzmann method
- Modular software concept
  - Supports various applications, currently planned:
    - Blood flow in aneurysms
    - Moving particles and agglomerates
    - Free surfaces to simulate foams, fuel cells, a.m.m.
    - Charged colloids
    - Arbitrary combinations of above
  - Integration in efficient massive-parallel environment
  - Standardized input and output routines
  - User-friendly interface
  - Platform independency with CMAKE

waLBerla project @

Widely applicable lattice Boltzmann from Erlangen

- **Porting issues concerning CMake:**
  - CMake has to be configured to find MPI
  - Not possible to specify Cluster Debugger Configurations via CMake (overwrites settings when project is built)
- **Visual Studio & Queues**
  - Not possible to automatically submit and debug parallel job via Visual Studio
- **Debugging issues**
  - MPI Cluster Debugger: Configuration pain to run jobs on remote sites
  - Remote Debugger not able to connect to queued jobs

Goals

- **WCCS as a development environment**
  - **Visual Studio**
  - **Parallel debugging**
  - **Different compilers**
  - **Some issues (Intel project system, parallel debugging), but ok in general**
- **Coupling of CCS cluster to MS Excel by use of VBA**
  - **Job construct, submit**
  - **Result retrieval**
  - **Visualization**
- **Behaviour of Windows on a ccNUMA architecture**
  - **Locality & affinity issues**
  - **Buffer cache**

Excel-CCS coupling

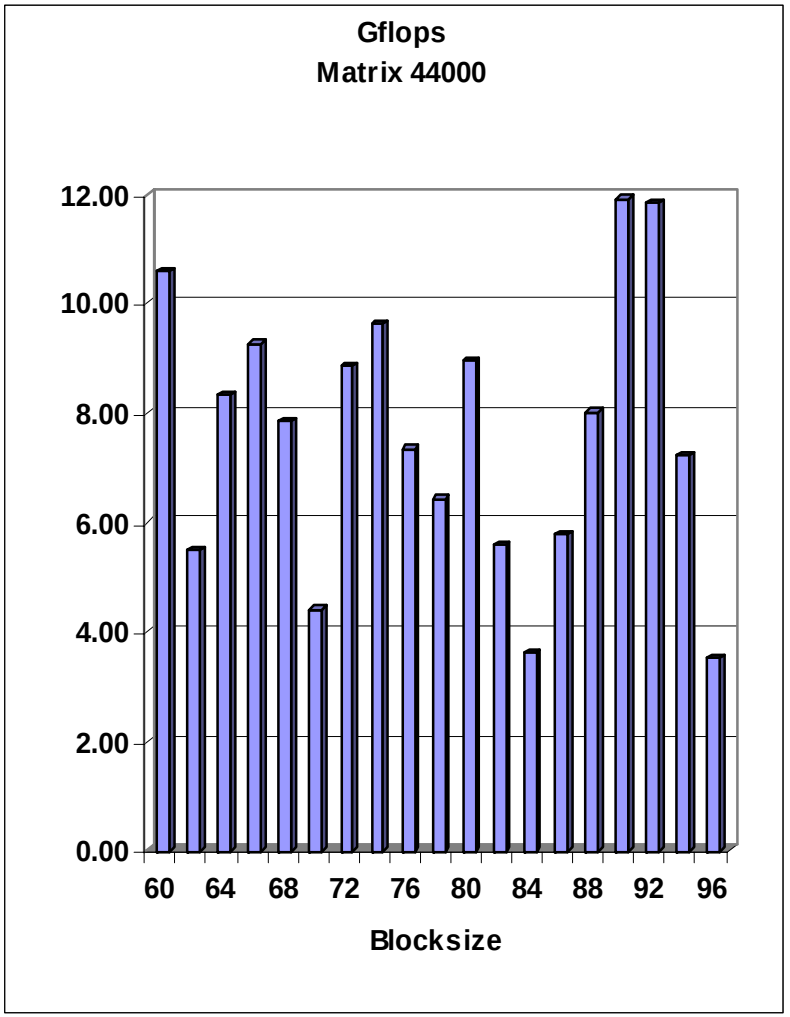
- **CCS API available for C++ and VBA**
  - **Documentation issues for VBA, but many examples online:**

    `http://www.microsoft.com/technet/scriptcenter/scripts/ccs/job/default.mspx?mfr=true`

  - **API is part of Office Professional only**
  - **Example Excel Sheet and VBA macros for LINPACK parameter scans provided**
- **Toy project: Make Excel sheet for simple heat equation solver with graphical output of performance numbers**

Evaluating Excel Integration (LINPACK Eample)

- Taking precompiled binaries offered by MS with ACML
- Tuning LINPACK parameters as suggested in: “Hands-On Lab – Building HPC LINPACK Tool”
- Issuing jobs directly to Job Manager
- Querying jobs
- Results are instantly plotted in Excel

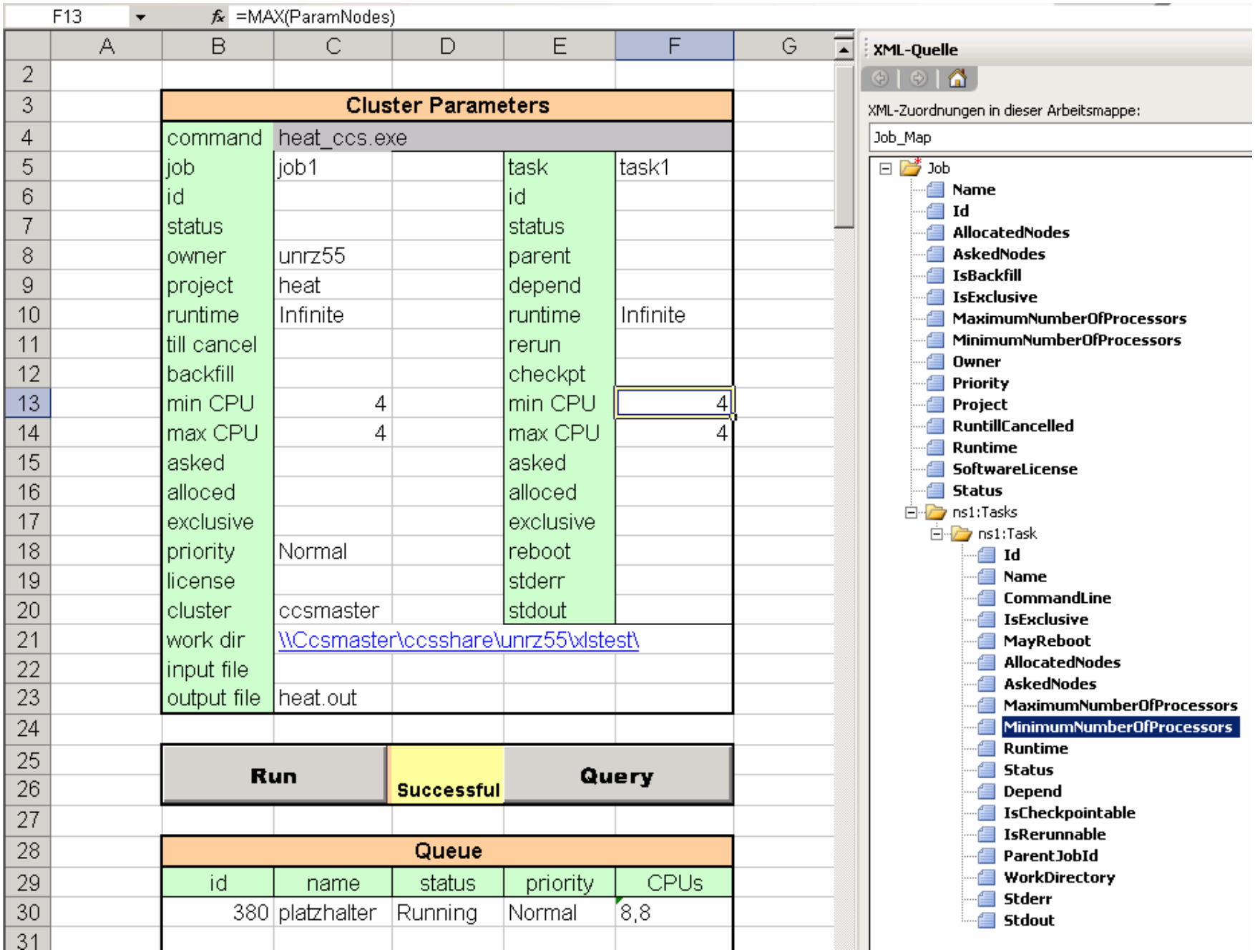


Excel-CCS coupling

- Principles of operation
  - Provide Excel worksheet with necessary parameters
    - Binary name, working dir
    - Number of CPUs, walltime limit
    - Input parameters for application
  - Position active elements (buttons,…) linked to VBA macros
  - VBA communicates with CCS using XML
    - First time generation of XML structure from
      - XML Schema
      - template file (saved from Job Manager app.)
      - scratch
    - Link entries to worksheet cells
  - Use VBA-CCS API to construct and submit jobs
    - Many options possible
  - Collect and parse output data and fill cells
    - Simple visualization possible using Excel graphs

job.xml template

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<Job xmlns:ns1="http://www.microsoft.com/ComputeCluster/"
   Name="job1"
   MaximumNumberOfProcessors="4"
   MinimumNumberOfProcessors="4"
   Owner="unrz55" Priority="Normal" Project="Jacobi"
   Runtime="Infinite">
   <ns1:Tasks>
      <ns1:Task Name="task1"
         CommandLine="echo 10 10 5 | heat_ccs.exe"
         MaximumNumberOfProcessors="4"
         MinimumNumberOfProcessors="4"
         Runtime="Infinite"
         WorkDirectory="\\Ccsmaster\ccsshare\unrz55\xlstest\"
         Stderr="3700-3700-50-err.out"
         Stdout="3700-3700-50-heat.out“ />
      </ns1:Tasks>
</Job>
```


F13
=MAX(ParamNodes)
Cluster Parameters
command heat_ccs.exe
job job1 task task1
id id
status status
owner unrz55 parent
project heat depend
runtime Infinite runtime Infinite
till cancel rerun
backfill checkpt
min CPU 4 min CPU 4
max CPU 4 max CPU 4
asked asked
alloced alloced
exclusive exclusive
priority Normal reboot
license stderr
cluster ccsmaster stdout
work dir \\Ccsmaster\ccsshare\unrz55\xlstest\
input file
output file heat.out
Run
Successful
Query
Queue
id name status priority CPUs
380 platzhalter Running Normal 8,8
XML-Quelle
XML-Zuordnungen in dieser Arbeitsmappe:
Job_Map
Job
Name
Id
AllocatedNodes
AskedNodes
IsBackfill
IsExclusive
MaximumNumberOfProcessors
MinimumNumberOfProcessors
Owner
Priority
Project
RuntillCancelled
Runtime
SoftwareLicense
Status
ns1:Tasks
ns1:Task
Id
Name
CommandLine
IsExclusive
MayReboot
AllocatedNodes
AskedNodes
MaximumNumberOfProcessors
MinimumNumberOfProcessors
Runtime
Status
Depend
IsCheckpointable
IsRerunnable
ParentJobId
WorkDirectory
Stderr
Stdout

Submitting a Job with VBA in Excel

```
' connect to cluster, defined by xls cell “Cluster”
Set objCluster =
    CreateObject("Microsoft.ComputeCluster.Cluster")
objCluster.Connect (Range("Cluster").Value)

' Job object from XML description
Set Job = objCluster.CreateJobFromXml(strXML)

' obtain user credentials
Set WshNetwork = CreateObject("WScript.Network")
UserName = WshNetwork.UserDomain & "\" &
    WshNetwork.UserName

' submit job to queue
ID = objCluster.QueueJob((Job), UserName, "", False, 0)
```

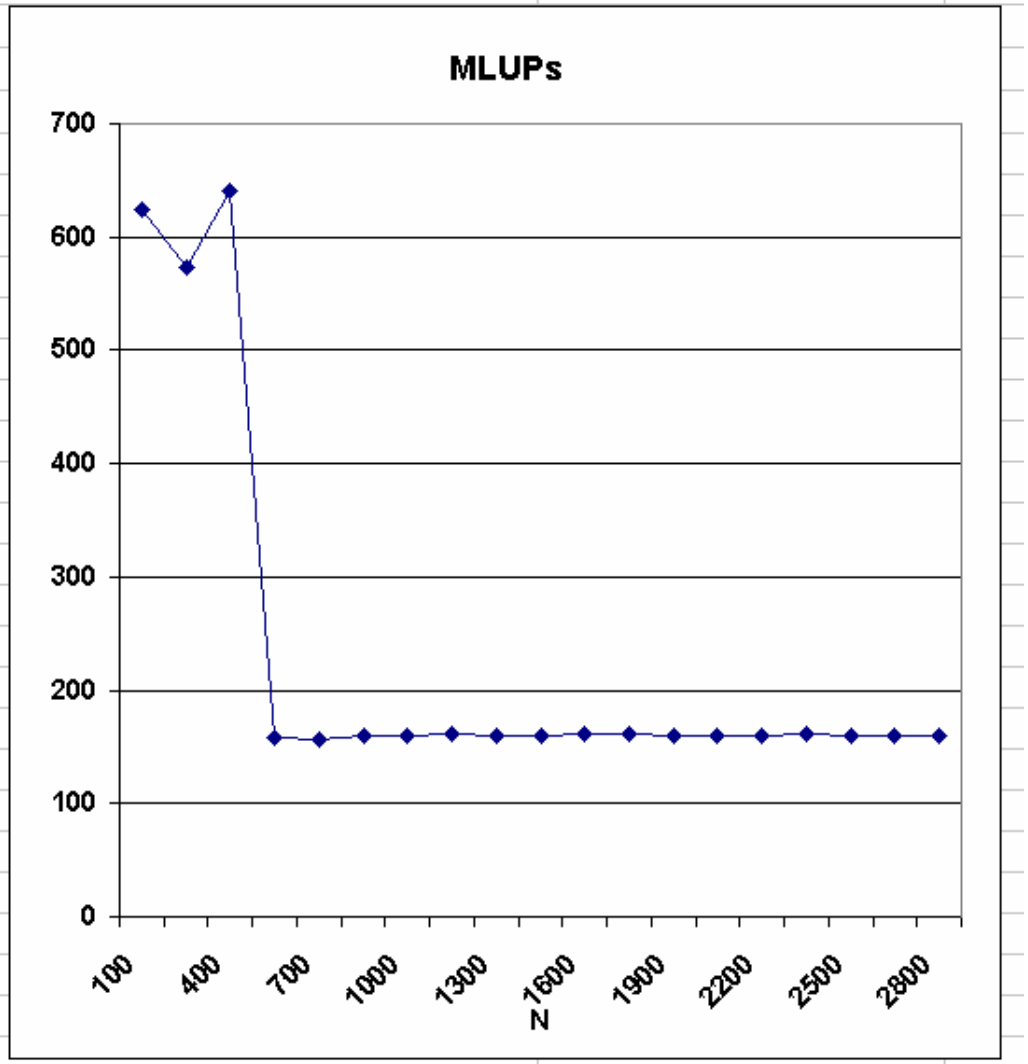
Many other CCS-API and general VBA functions available

- **Status query, job cancel etc.**
- **VBA: Regexp package …**

Result retrieval e.g. by VBScript Regexp Package

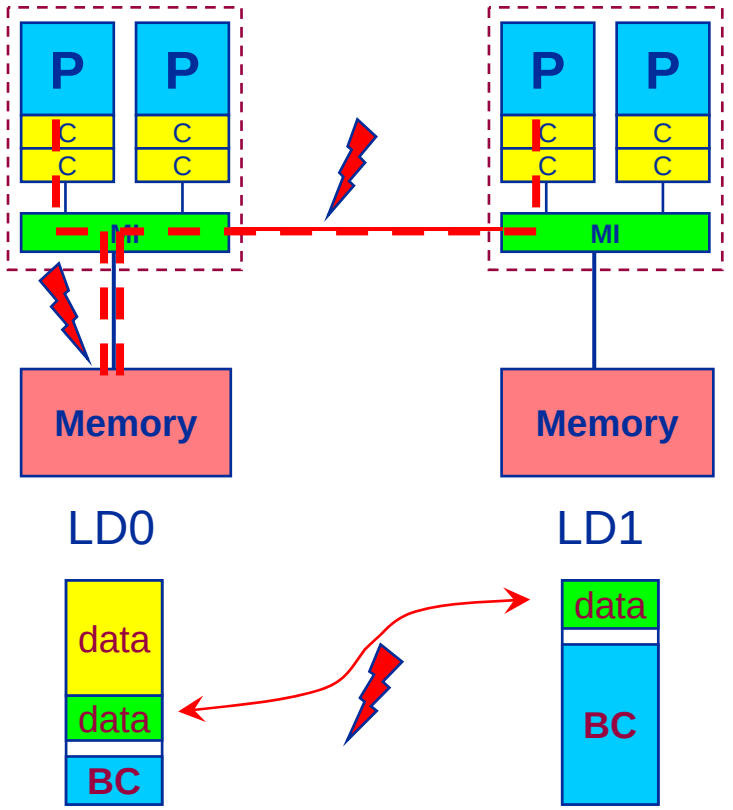
Execute		Retreive

Results			
NX	NY	#THR	MLUPs
100	100	4	625,00
250	250	4	573,39
400	400	4	640,00
550	550	4	158,17
700	700	4	156,88
850	850	4	159,32
1000	1000	4	160,00
1150	1150	4	161,28
1300	1300	4	159,13
1450	1450	4	160,25
1600	1600	4	160,60
1750	1750	4	160,68
1900	1900	4	160,44
2050	2050	4	160,03
2200	2200	4	159,63
2350	2350	4	160,63
2500	2500	4	160,01
2650	2650	4	160,55
2800	2800	4	159,80

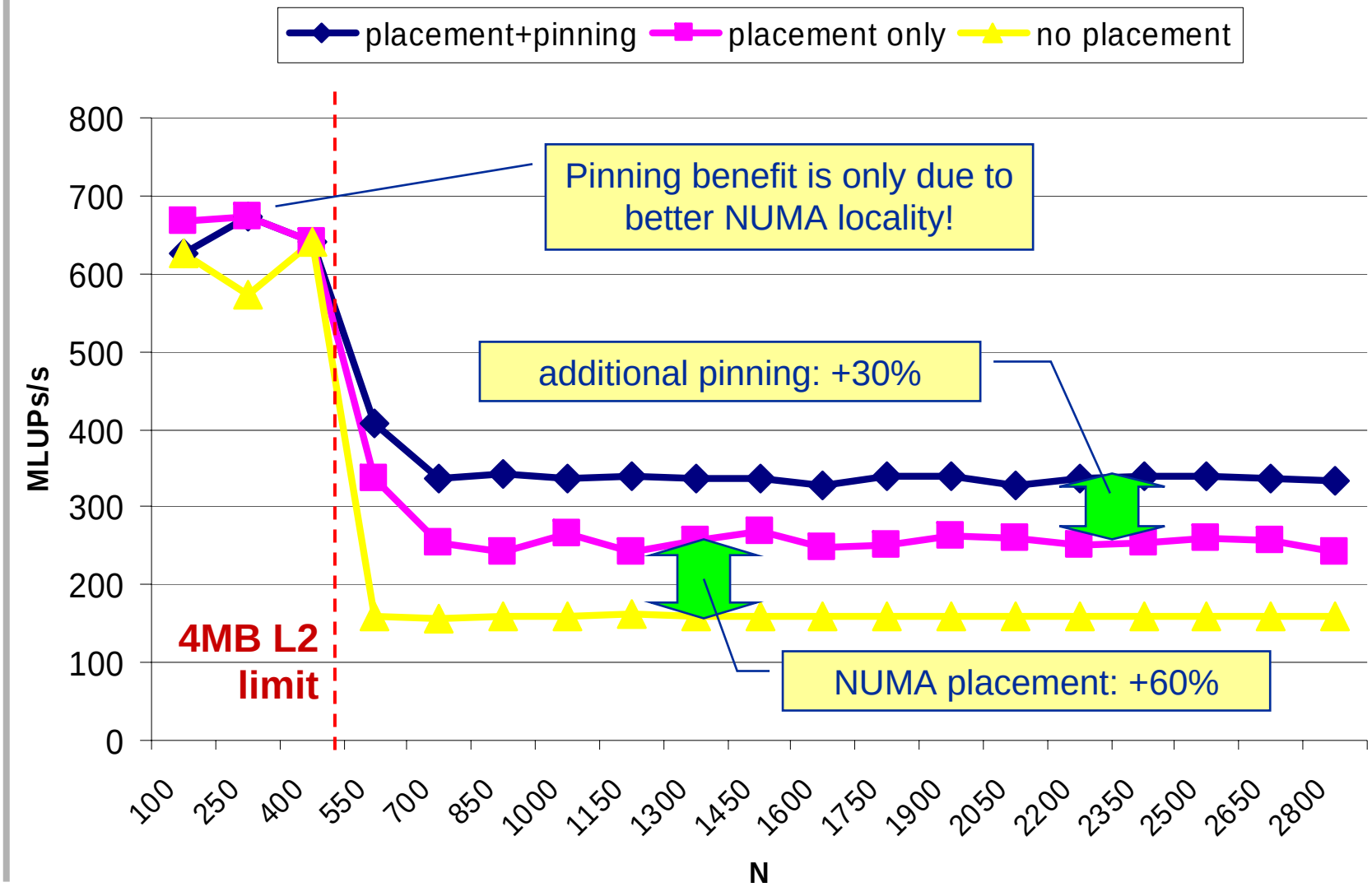


Windows on ccNUMA

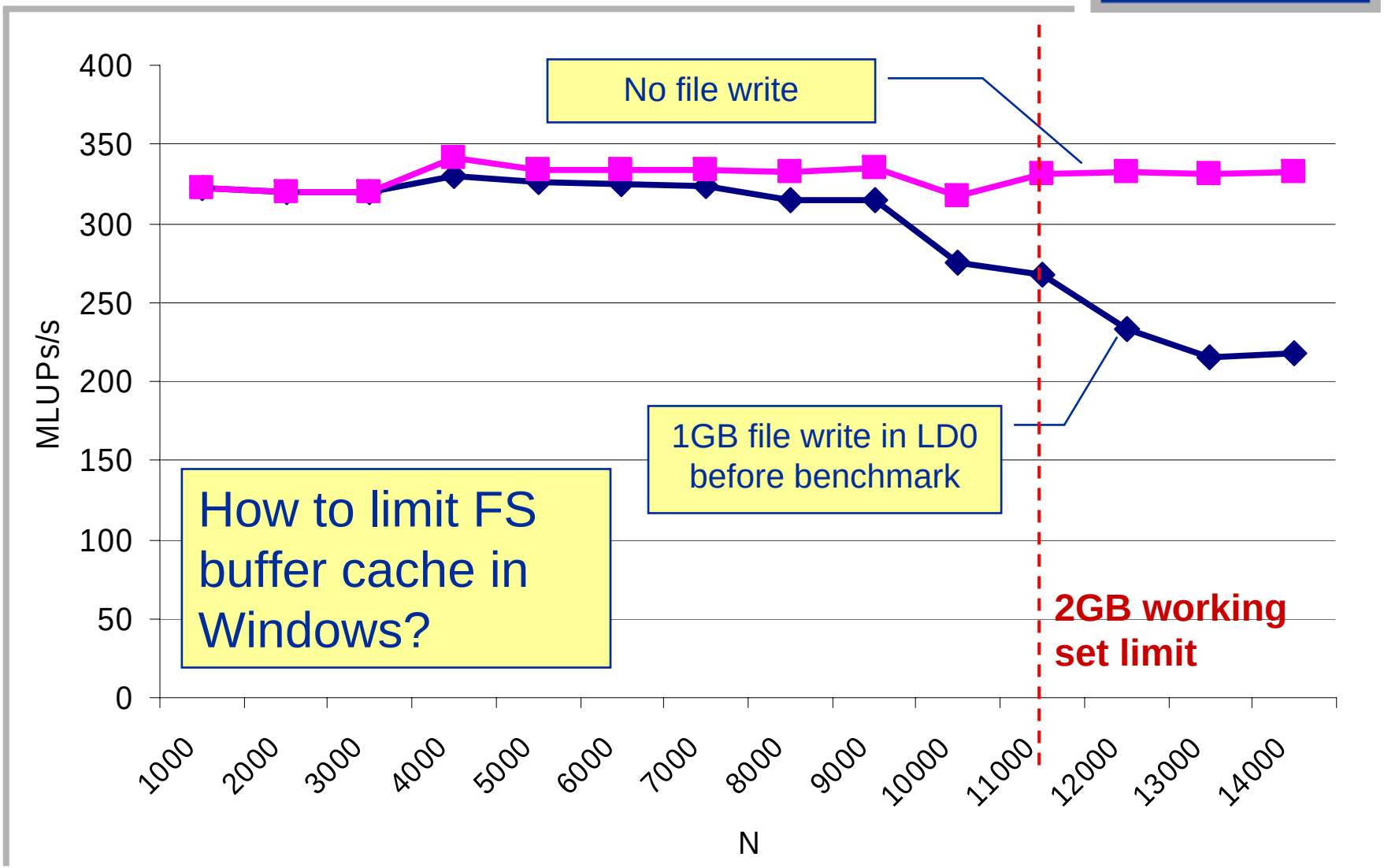
- Locality and congestion problems on ccNUMA
- "First touch" policy for memory pages ensures local placement
  - Watch OpenMP init loops
- Even if placement is correct, make sure it stays that way
  - "pin" threads/processes to initial sockets
  - Issue with OpenMP and MPI
- To make matters worse, FS buffer cache can fill LDs so that apps must use nonlocal memory
  - Use "memory sweeper" before production
- How does all this work on Windows?



NUMA Placement and Pinning with Heat Conduction Solver (Relaxation)



Buffer Cache and Page Placement



Future plans

- Test of Ansys CFX 11
- Test of StarCD
- Test of Allinea DDTLite for Visual Studio
- WalbErla (CFD) development and evaluation of Windows Performance (see Projects)
- Customized Excel sheets for standard production applications (see above)

Conclusions

- “Well-known” development environment with HPC add-ons
- Batch system/scheduler is not “enterprise-class”
- Ease of use (develop/compile/debug/job submit)
- Room for improvement with parallel debugging
- Similar ccNUMA issues as with Linux, same remedies
  - Process/thread pinning absolutely essential
- CCS VBA API is extremely fun to play with
  - May be attractive to production-only users
  - Still lacks some coherent documentation
  - Only available with Office Professional