November 5, 2010

MPI at MPI

Jens Saak

Max Planck Institute for Dynamics of Complex Technical Systems
Computational Methods in Systems and Control Theory

MAX PLANCK INSTITUTE
FOR DYNAMICS OF COMPLEX
TECHNICAL SYSTEMS
MAGDEBURG

Parallel Computing Basics

Why Parallel Computing?

nce upon a time ...

Parallel Computing Basics

Why Parallel Computing?

nce upon a time ...

...the world was simple...

Parallel Computing Basics

Why Parallel Computing?

nce upon a time ...

...the world was simple...

...and problems were small...

Parallel Computing Basics

Why Parallel Computing?

nce upon a time ...

...the world was simple...

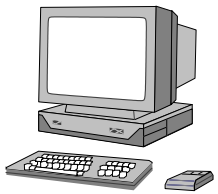
...and problems were small...

...all computations could be carried out on a single computer.

Parallel Computing Basics

Why Parallel Computing?

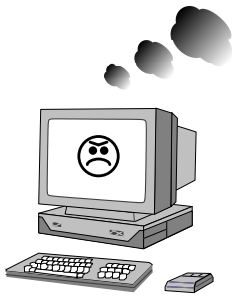
Then the world grew complicated and problems became large and difficult.



Parallel Computing Basics

Why Parallel Computing?

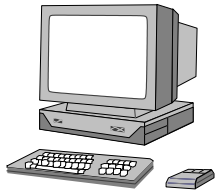
Then the world grew complicated and problems became large and difficult.
Computations became to challenging for a single computer.

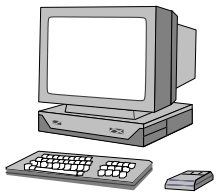


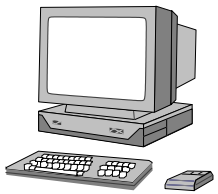
Parallel Computing Basics

Why Parallel Computing?

Then the world grew complicated and problems became large and difficult.
Computations became to challenging for a single computer.
Therefore problems were split



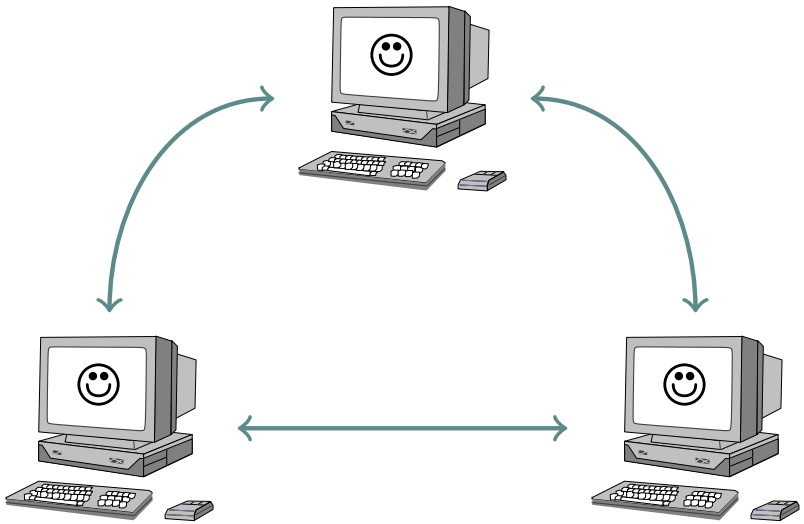




Parallel Computing Basics

Why Parallel Computing?

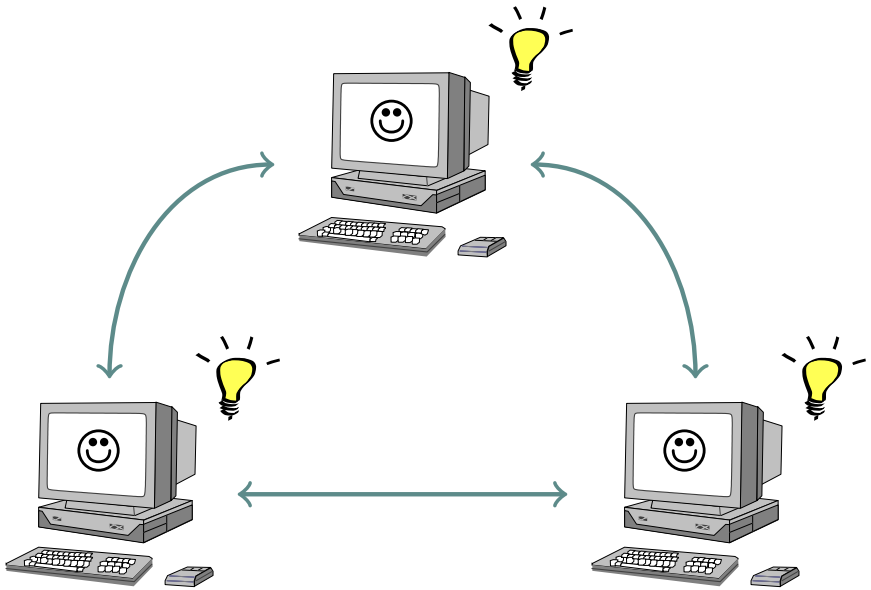
Then the world grew complicated and problems became large and difficult.
Computations became to challenging for a single computer.
Therefore problems were split and distributed to multiple computers



Parallel Computing Basics

Why Parallel Computing?

Then the world grew complicated and problems became large and difficult.
Computations became to challenging for a single computer.
Therefore problems were split and distributed to multiple computers jointly finding their solution.



Parallel Computing Basics

Types of Parallelism

- Quasi Parallelism, or Multitasking

modern operating systems emulate parallelism via time-slicing

Parallel Computing Basics

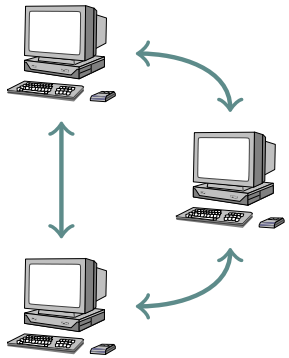
Types of Parallelism

- Quasi Parallelism, or Multitasking

  modern operating systems emulate parallelism via time-slicing

- Distributed Memory Parallelism

  computation on separate units with own memory on each unit



Parallel Computing Basics

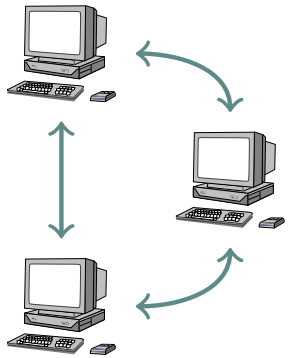
Types of Parallelism

- Quasi Parallelism, or Multitasking

  modern operating systems emulate parallelism via time-slicing

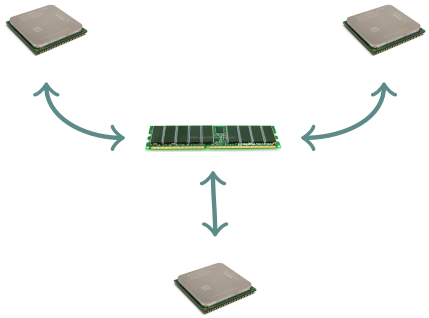
- Distributed Memory Parallelism

  computation on separate units with own memory on each unit



- Shared Memory Parallelism

  multiple computation units access a common memory



Parallel Computing Basics

Implementations and Standards

- Quasi Parallelism
  - operating systems scheduler
  - processes
  - process-threads (e.g. POSIX-threads, pthreads)

Parallel Computing Basics

Implementations and Standards

- Quasi Parallelism
  - operating systems scheduler
  - processes
  - process-threads (e.g. POSIX-threads, pthreads)
- Distributed Memory Parallelism

  Data handling via the Message Passing Interface (MPI)

  (standard for a collection of operations and their semantics, NOT prescribing the actual protocol or implementation.)
  - MVAPICH/MVAPICH2 (implements MPI-1/MPI-2, BSD-License)
  - OpenMPI (implements MPI-1, MPI-2, freely available)
  - MPICH2 (implements MPI-1, MPI-2, MPI-2.1, MPI-2.2, freely available)

Parallel Computing Basics

Implementations and Standards

- Quasi Parallelism
  - operating systems scheduler
  - processes
  - process-threads (e.g. POSIX-threads, pthreads)
- Distributed Memory Parallelism

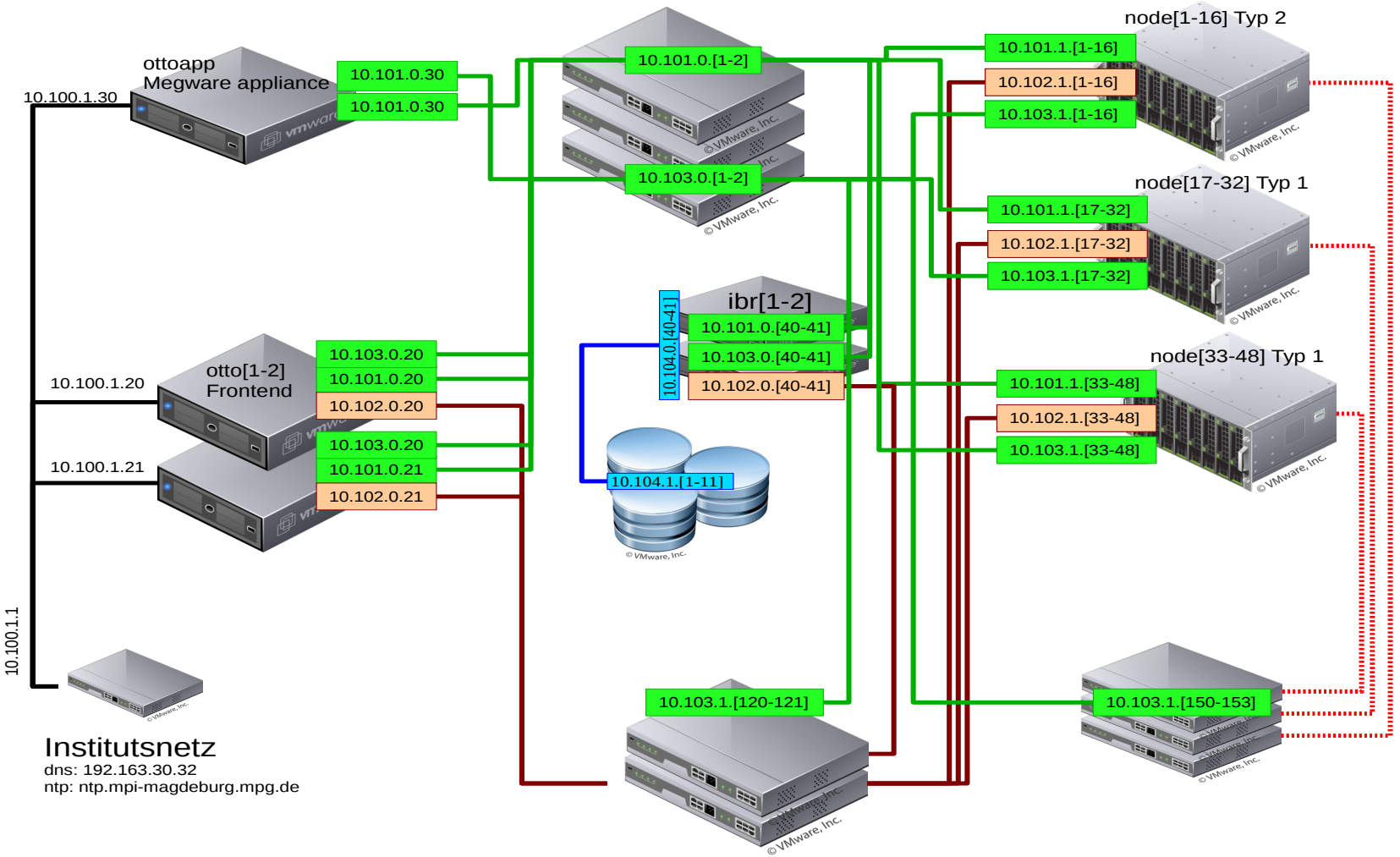
  Data handling via the Message Passing Interface (MPI)

  (standard for a collection of operations and their semantics, NOT prescribing the actual protocol or implementation.)
  - MVAPICH/MVAPICH2 (implements MPI-1/MPI-2, BSD-License)
  - OpenMPI (implements MPI-1, MPI-2, freely available)
  - MPICH2 (implements MPI-1, MPI-2, MPI-2.1, MPI-2.2, freely available)
- Shared Memory Parallelism

  OpenMP API: compiler and preprocessor directives. (latest version 3.0; May 2008)
  - GCC 4.3.2 and later
  - Intel Compilers 10.1 and later
  - Oracle, IBM, Cray, ...

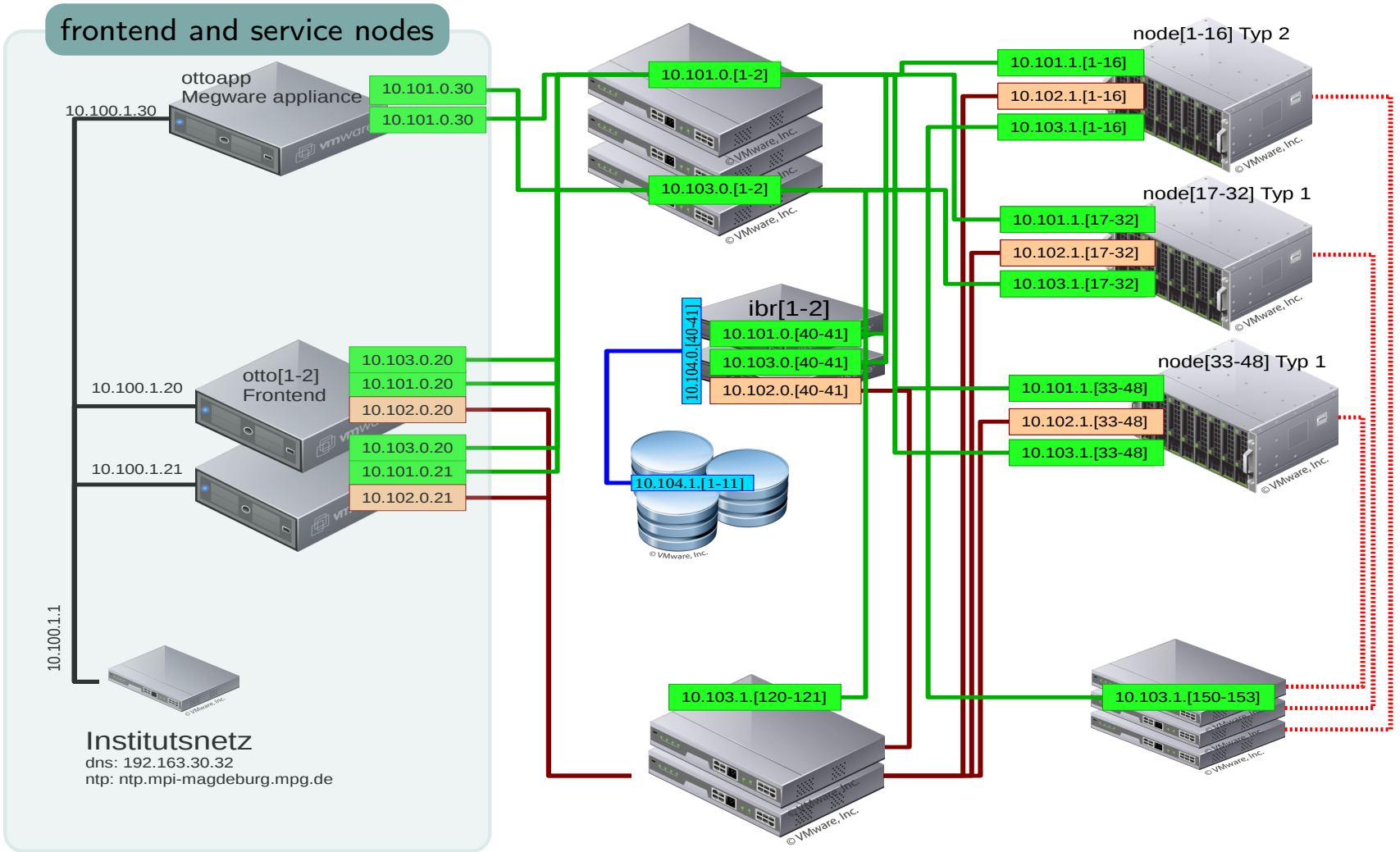
Parallel Computing at MPI Magdeburg

otto



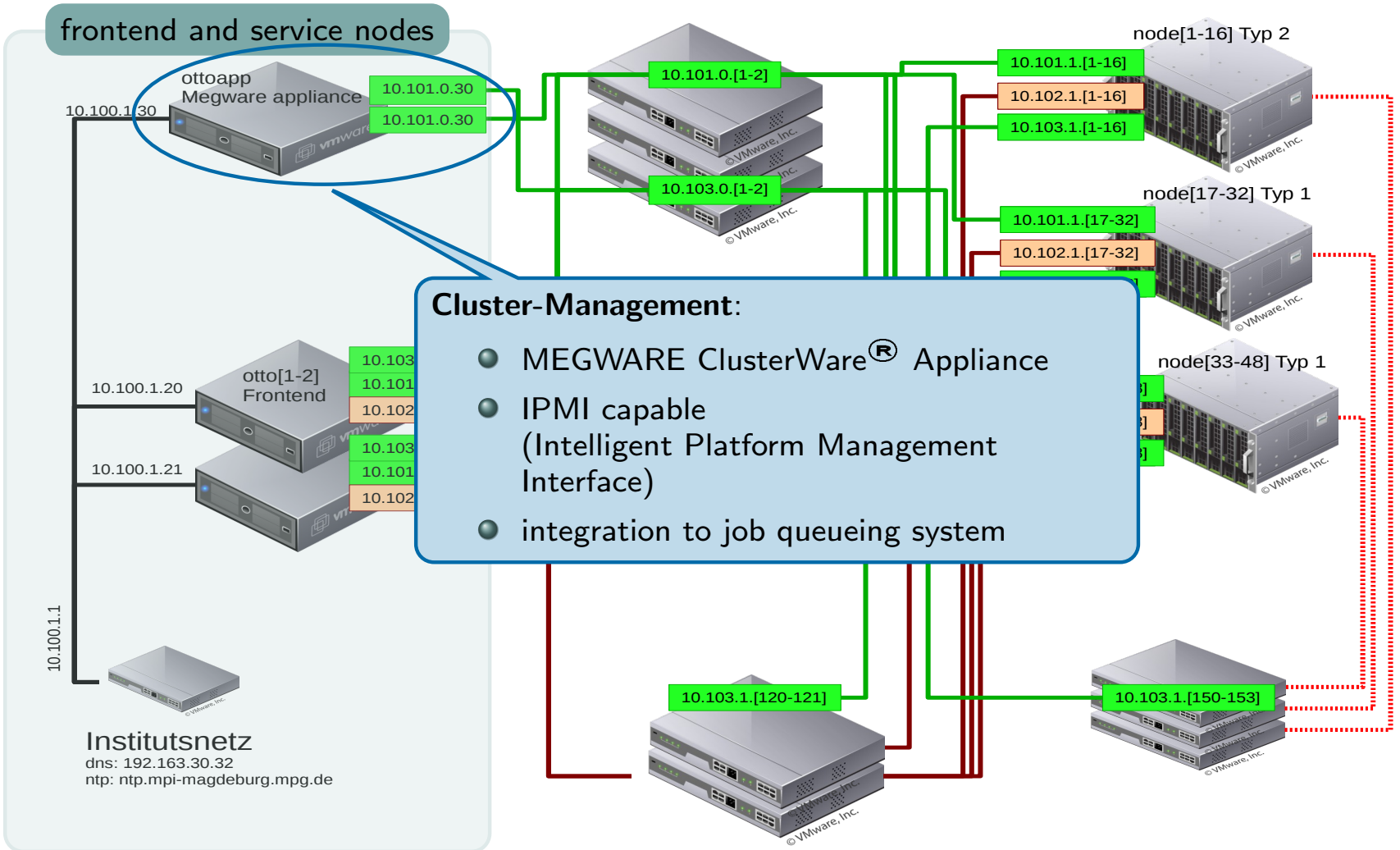
Parallel Computing at MPI Magdeburg

otto



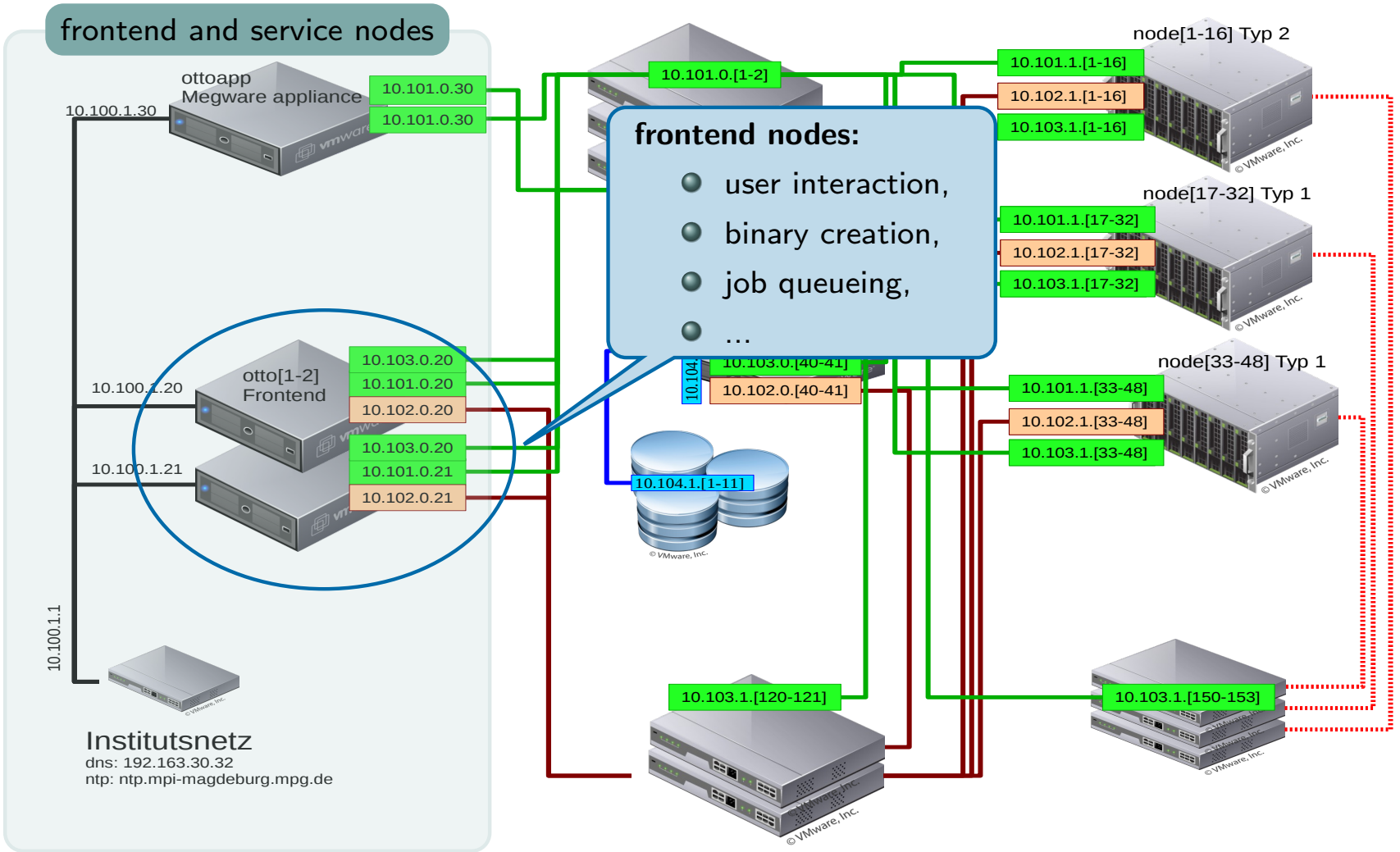
Parallel Computing at MPI Magdeburg

otto



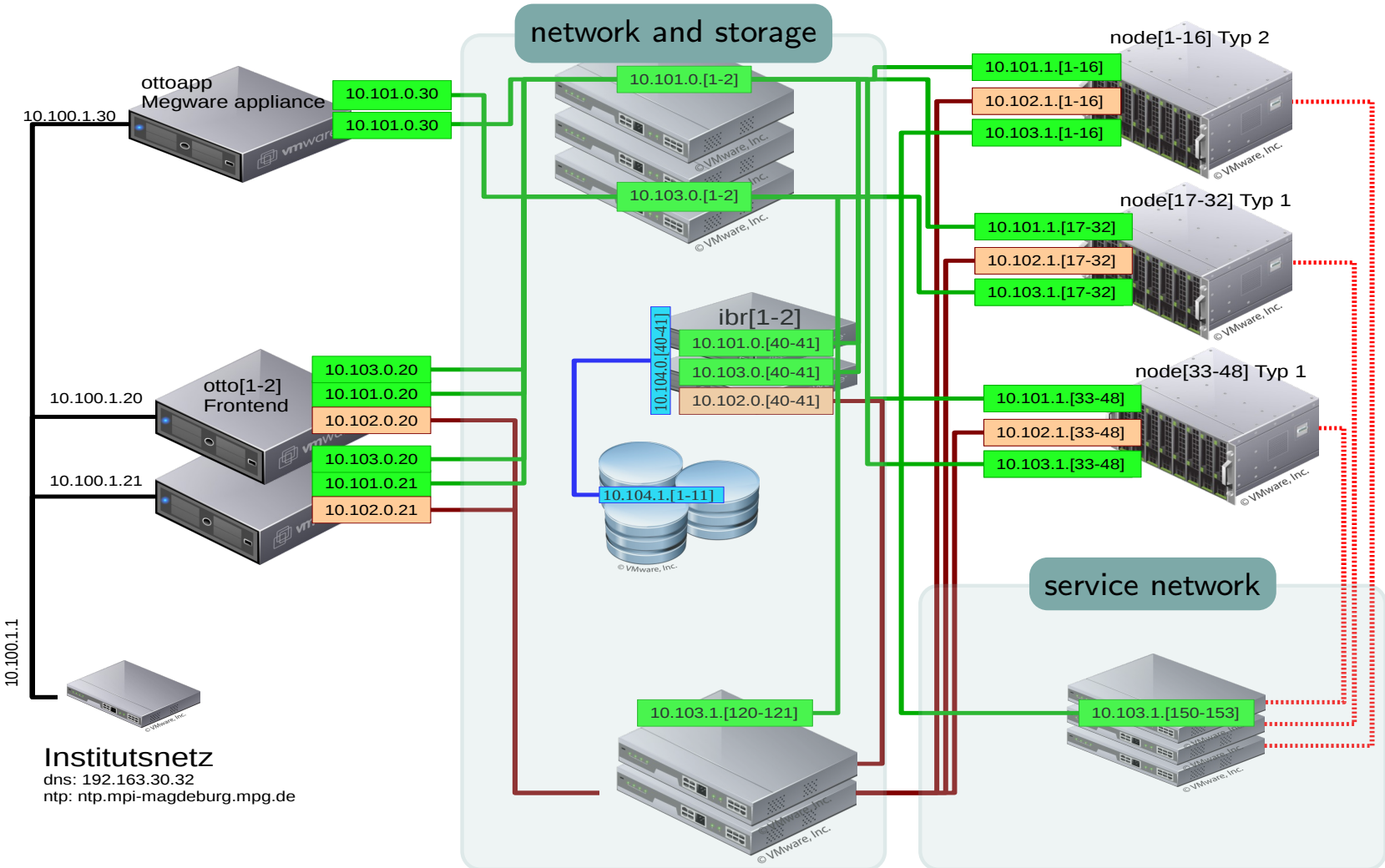
Parallel Computing at MPI Magdeburg

otto



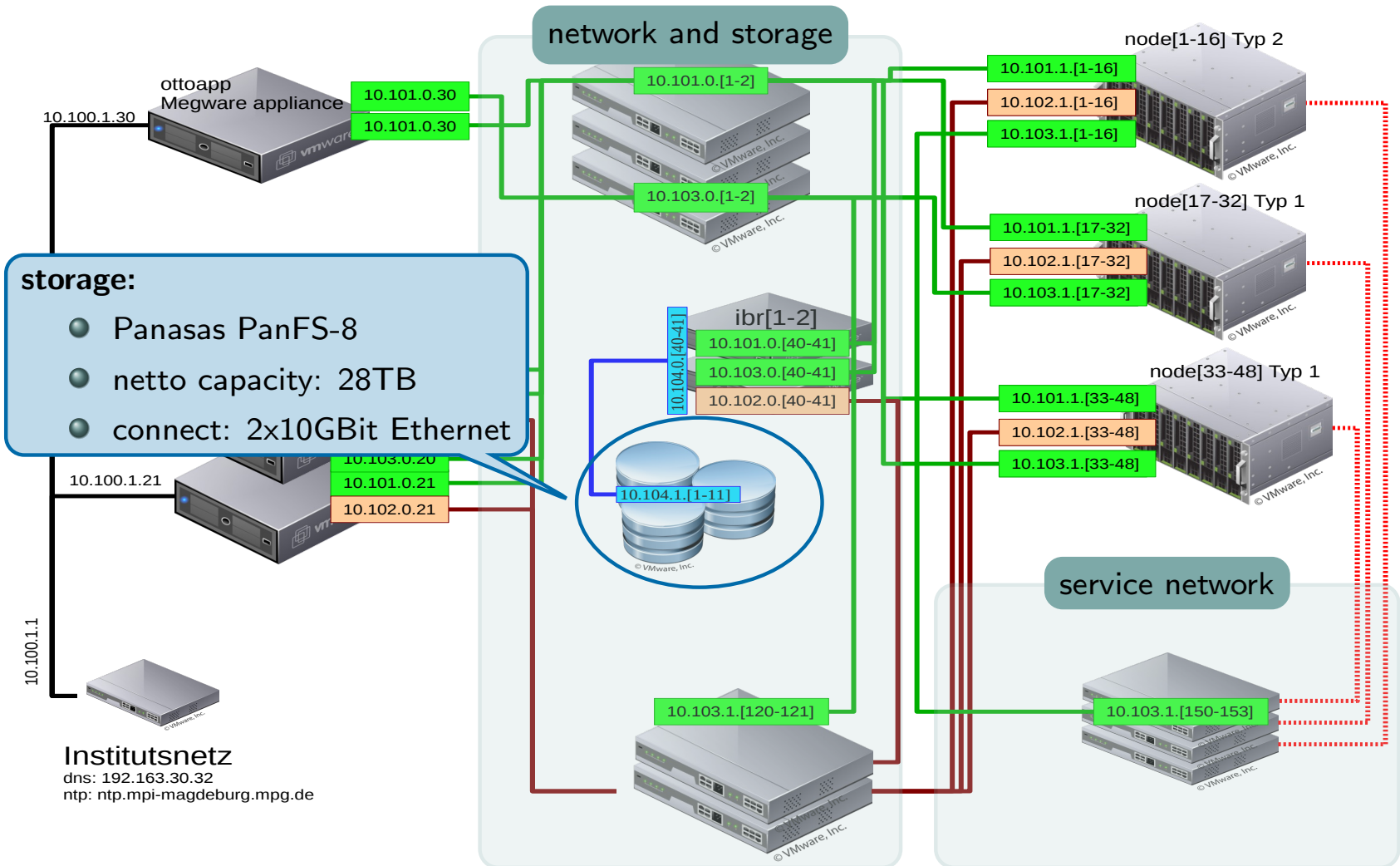
Parallel Computing at MPI Magdeburg

otto



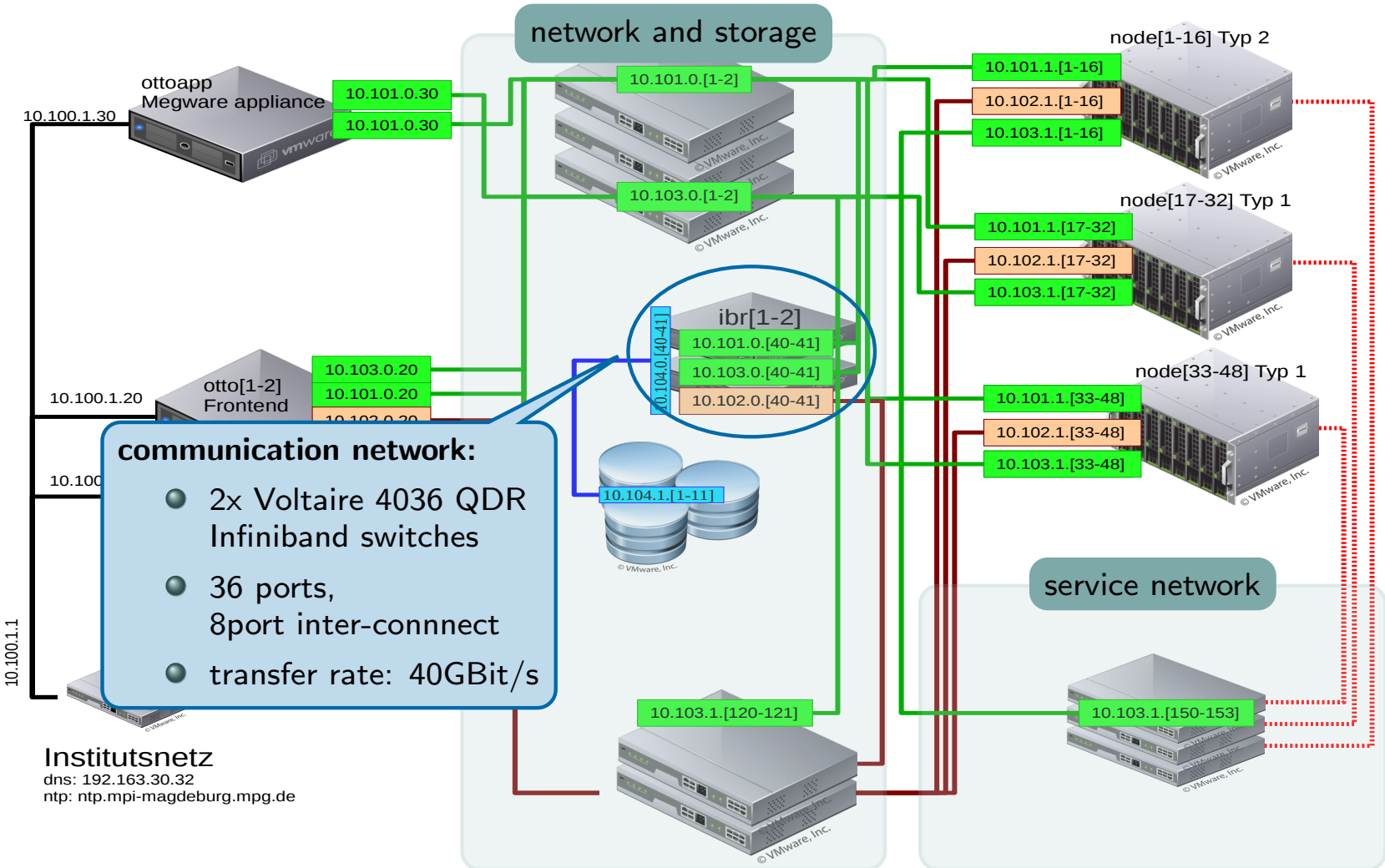
Parallel Computing at MPI Magdeburg

otto



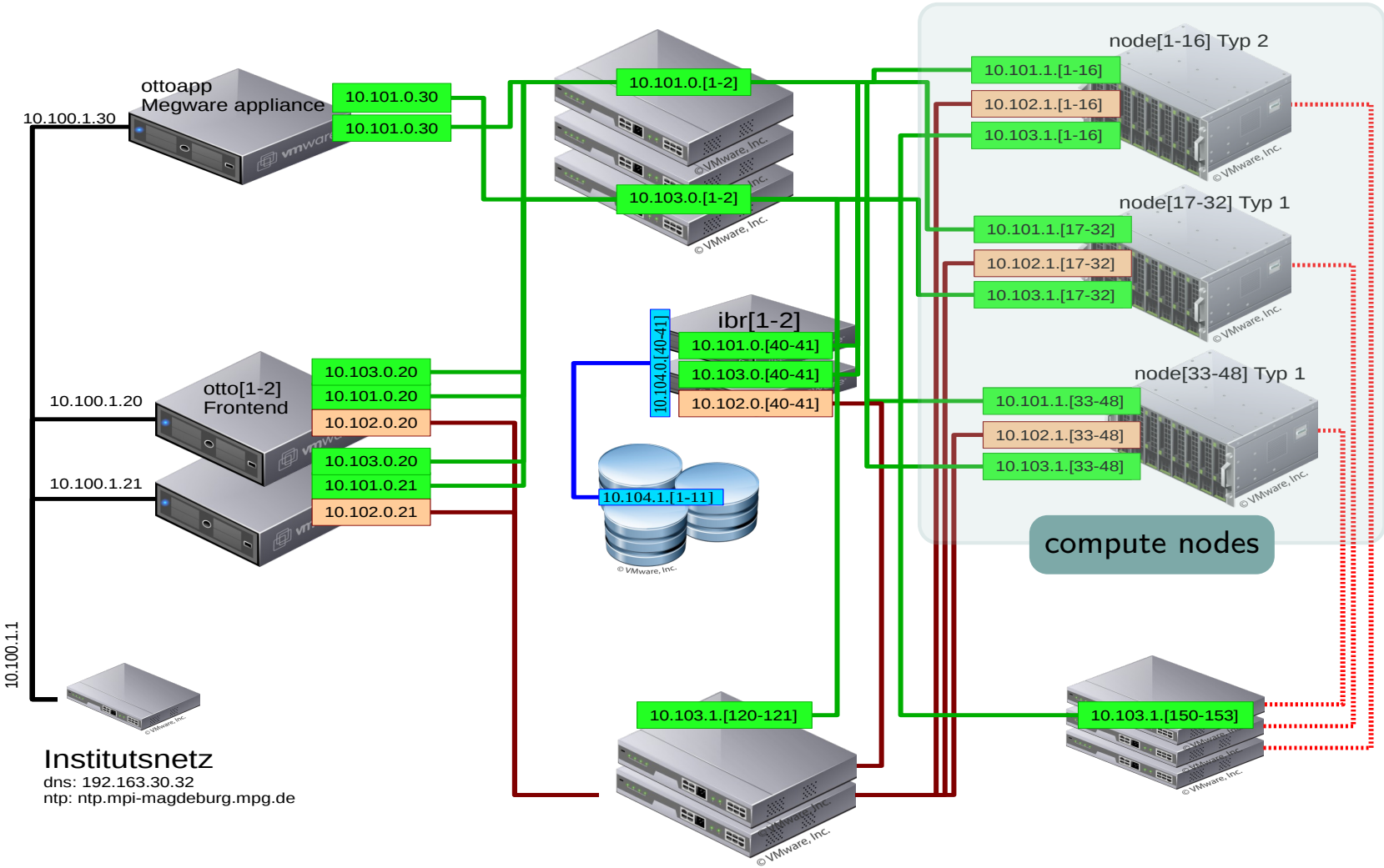
Parallel Computing at MPI Magdeburg

otto



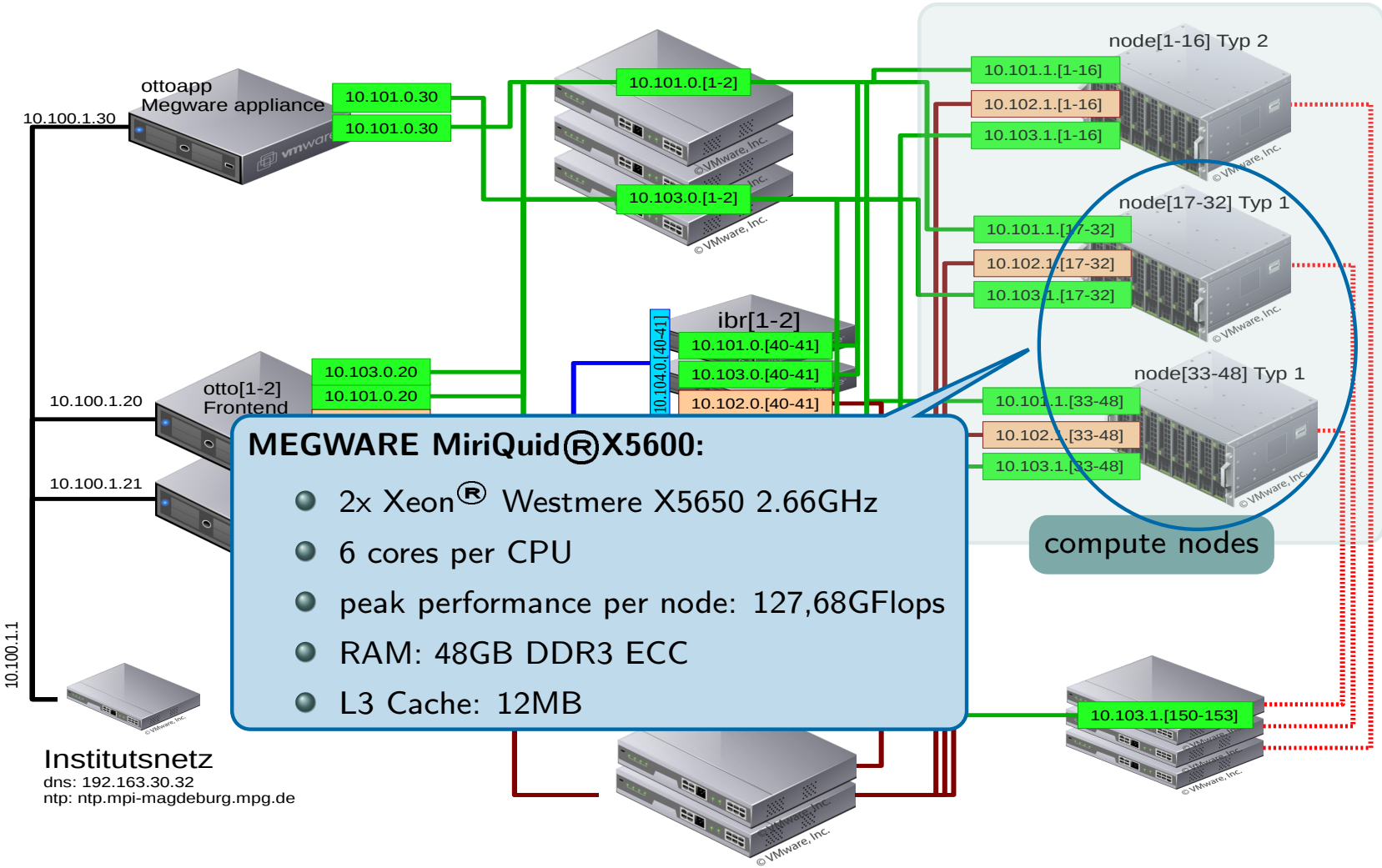
Parallel Computing at MPI Magdeburg

otto



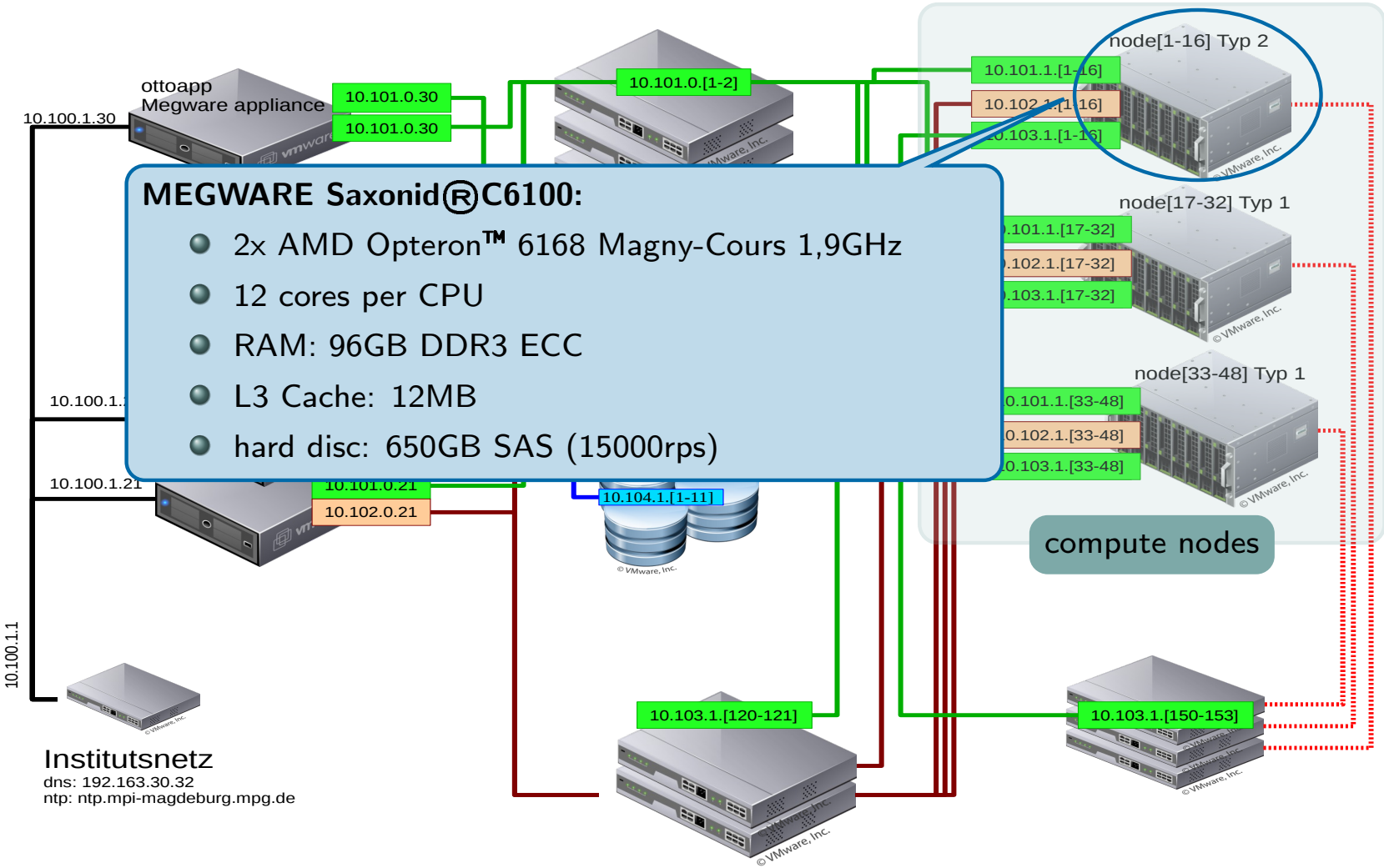
Parallel Computing at MPI Magdeburg

otto



Parallel Computing at MPI Magdeburg

otto

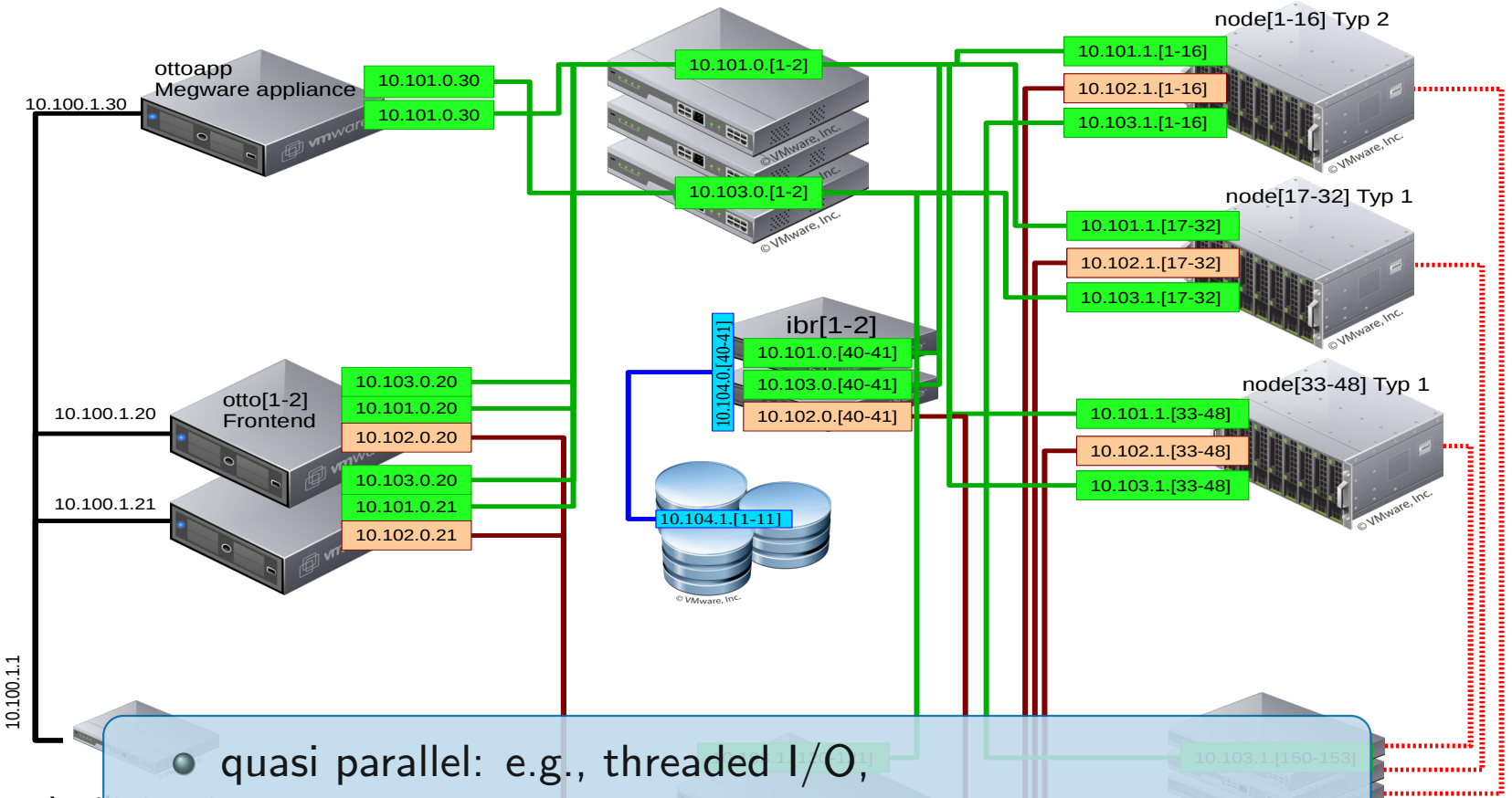


MEGWARE Saxonid®C6100:

- 2x AMD Opteron™ 6168 Magny-Cours 1,9GHz
- 12 cores per CPU
- RAM: 96GB DDR3 ECC
- L3 Cache: 12MB
- hard disc: 650GB SAS (15000rps)

Parallel Computing at MPI Magdeburg

otto



- quasi parallel: e.g., threaded I/O,
- distributed: every node has its own RAM,
- shared: 12/24 cores per node access the common RAM.

Parallel Computing at MPI Magdeburg

Molecular Simulations and Design

From wavefunctions to proteins in networks

The Molecular Simulations and Design (MSD) group aims at an explanation of chemical and biological phenomena at molecular level.

Complex phenomena in chemistry and biology

MSD group develops and applies tools from

- Quantum mechanics
- QM/MM hybrid simulations
- Molecular dynamics
- Brownian dynamics
- Bioinformatics
- Protein structural modeling

Contact: Dr. Matthias Stein

Parallel Computing at MPI Magdeburg

Computational Methods in Systems and Control Theory (CSC)

Interests:

Mathematical methods for application in systems and control theory and their numerical realization, with focus on Numerical Linear Algebra.

Parallel Computing:

- accelerate solvers on modern multi-core processors
- solve very large
  - matrix equations,
  - model order reduction problems,
  - or optimal control problems

  on high performance parallel machines.

Parallel Computing aware Software developed in CSC:

- PLiCMR
- C.M.E.S.S.

Parallel Computing in CSC

PLiCMR

Parallel Library in Control and Model Reduction.

http://www.pscom.uji.es/modred/index.html

Distributed memory parallel model order reduction via

- Balanced Truncation
- Singular Perturbation Approximation
- Hankel-Norm Approximation

Partners: High Performance Computing and Architectures Group
(University Jaume I; Castellón; Spain)
http://www.hpca.uji.es/

Parallel Computing in CSC

C.M.E.S.S.

C-language **M**atrix **E**quation **S**parse **S**olver

C-Library for solving large sparse:

- Lyapunov equations
  $\mathbf{FX} + \mathbf{XF}^T = -\mathbf{GG}^T, \qquad \mathbf{FXE}^T + \mathbf{EXF}^T = -\mathbf{GG}^T,$

See poster, as well.

Parallel Computing in CSC

C.M.E.S.S.

C-language **M**atrix **E**quation **S**parse **S**olver

C-Library for solving large sparse:

- Lyapunov equations
  $\mathbf{FX} + \mathbf{XF}^T = -\mathbf{GG}^T, \qquad \mathbf{FXE}^T + \mathbf{EXF}^T = -\mathbf{GG}^T,$
- Riccati equations
  $\mathbf{C} + \mathbf{A}^T\mathbf{X} + \mathbf{XA} - \mathbf{XBX} = 0, \quad \mathbf{C} + \mathbf{A}^T\mathbf{XE} + \mathbf{E}^T\mathbf{XA} - \mathbf{E}^T\mathbf{XBXE} = 0$

See poster, as well.

Parallel Computing in CSC

C.M.E.S.S.

C-language **M**atrix **E**quation **S**parse **S**olver

C-Library for solving large sparse:

- Lyapunov equations
  $\mathbf{FX} + \mathbf{XF}^T = -\mathbf{GG}^T, \qquad \mathbf{FXE}^T + \mathbf{EXF}^T = -\mathbf{GG}^T,$
- Riccati equations
  $\mathbf{C} + \mathbf{A}^T\mathbf{X} + \mathbf{XA} - \mathbf{XBX} = 0, \quad \mathbf{C} + \mathbf{A}^T\mathbf{XE} + \mathbf{E}^T\mathbf{XA} - \mathbf{E}^T\mathbf{XBXE} = 0$
- Model Reduction
  - Balanced Truncation (first and second order systems)
  - $\mathcal{H}_2$-optimal Reduction (IRKA, TSIA)

See poster, as well.

Parallel Computing in CSC

C.M.E.S.S.

C-language **M**atrix **E**quation **S**parse **S**olver

C-Library for solving large sparse:

- Lyapunov equations
  $\mathbf{FX} + \mathbf{XF}^T = -\mathbf{GG}^T, \qquad \mathbf{FXE}^T + \mathbf{EXF}^T = -\mathbf{GG}^T,$
- Riccati equations
  $\mathbf{C} + \mathbf{A}^T\mathbf{X} + \mathbf{XA} - \mathbf{XBX} = 0, \quad \mathbf{C} + \mathbf{A}^T\mathbf{XE} + \mathbf{E}^T\mathbf{XA} - \mathbf{E}^T\mathbf{XBXE} = 0$
- Model Reduction
  - Balanced Truncation (first and second order systems)
  - $\mathcal{H}_2$-optimal Reduction (IRKA, TSIA)
- Linear Quadratic Regulator Problems

See poster, as well.

Parallel Computing in CSC

C.M.E.S.S.

C-language **M**atrix **E**quation **S**parse **S**olver

C-Library for solving large sparse:

- Lyapunov equations
  $\mathbf{FX} + \mathbf{XF}^T = -\mathbf{GG}^T, \qquad \mathbf{FXE}^T + \mathbf{EXF}^T = -\mathbf{GG}^T,$
- Riccati equations
  $\mathbf{C} + \mathbf{A}^T\mathbf{X} + \mathbf{XA} - \mathbf{XBX} = 0, \quad \mathbf{C} + \mathbf{A}^T\mathbf{XE} + \mathbf{E}^T\mathbf{XA} - \mathbf{E}^T\mathbf{XBXE} = 0$
- Model Reduction
  - Balanced Truncation (first and second order systems)
  - $\mathcal{H}_2$-optimal Reduction (IRKA, TSIA)
- Linear Quadratic Regulator Problems

Currently focused on multi-core desktop computers, i.e., shared memory.

See poster, as well.

The End

Read "MPI at MPI" as

The End

Read "MPI at MPI" as

Message Passing Interface at Max Planck Institute

The End

Read “MPI at MPI” as

Message Passing Interface at Max Planck Institute

Meet `otto` at MPI this November,

The End

Read “MPI at MPI” as

Message Passing Interface at Max Planck Institute

Meet `otto` at MPI this November,

and let us hope that this story ends:

The End

Read “MPI at MPI” as

Message Passing Interface at Max Planck Institute

Meet `otto` at MPI this November,

and let us hope that this story ends:

> “...and they all computed happily ever after.”