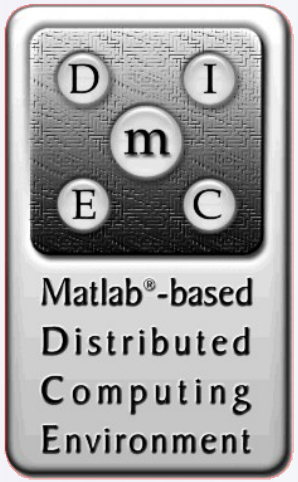
ParCo 2003

MDICE

R. Pfarrhofer, P. Bachhiesl, M. Kelz,
H. Stögner, and A. Uhl

0056pfro@edu.fh-kaernten.ac.at



A MATLAB Toolbox for Efficient Cluster Computing

Outline

- Introduction
- Fundamentals of MDICE
- Experiment set-up and application
- Experimental results
- Conclusion

Introduction (1)

- MATLAB established as the numerical computing environment
- Many scientific applications demand a high level of performance
- Availability of HPC systems in many universities and companies

➔ Employing MATLAB on such systems is obvious

Introduction (2)

- Ways to use MATLAB on HPC architectures
  - Calling high performance numerical libraries
  - Compiling MATLAB to another language
  - Developing a high performance interpreter
    - Message passing
    - Split up work among multiple MATLAB sessions

Introduction (3)

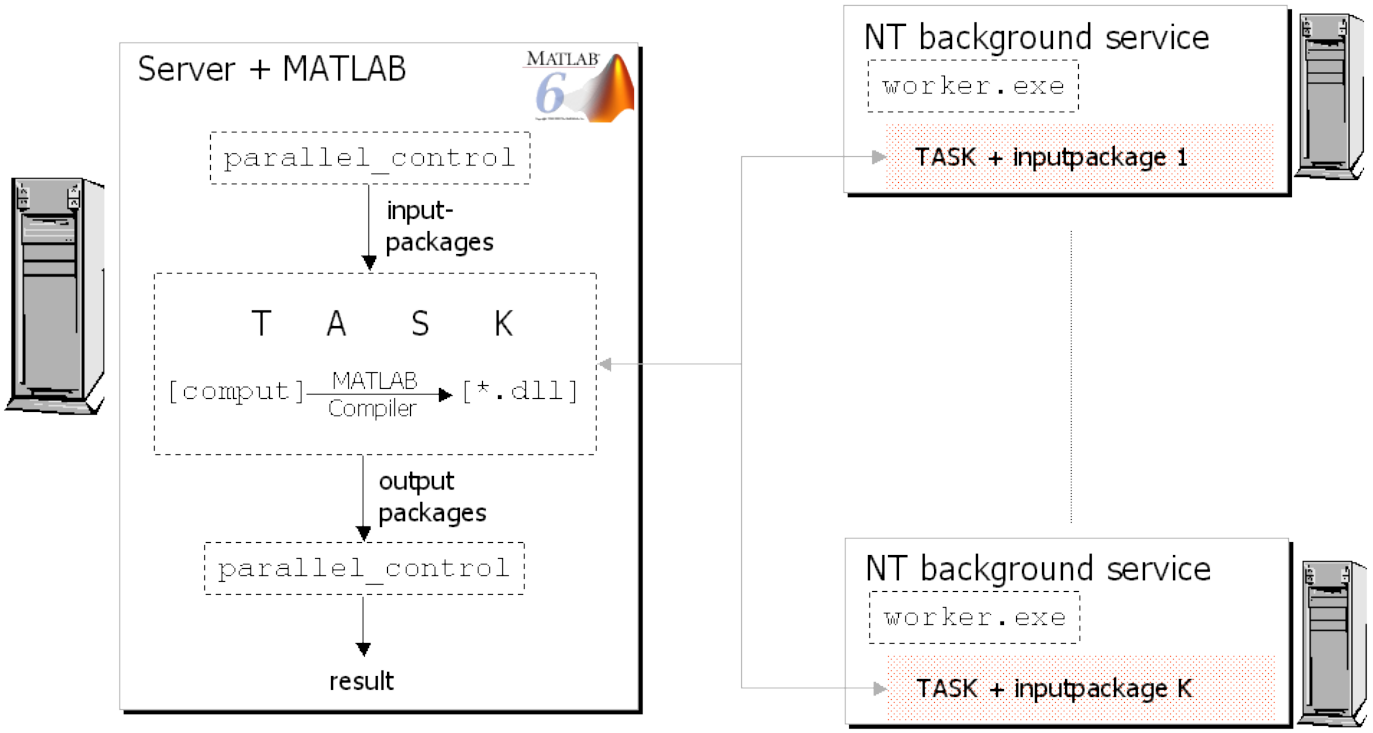
- Our intention is to develop
  - a high performance interpreter
  - which only requires one MATLAB Client on the server machine
- The development should avoid
  - expensive licensing fees and
  - expensive computational resources

MDICE (1)

- MDICE
  - is based on the PARMATLAB and TCPIP toolboxes
  - supports coarse grained parallelization and
  - distributes processes over the intranet/internet

MDICE (2)

- Client-server concept of MDICE



MDICE (3)

- Important compiler limitations and restrictions
  - Built-in MATLAB functions cannot be compiled
    - but available in the MATLAB C/C++ Library
    - About 70 functions (e.g. `diary`, `whos`, etc.) are not supported
  - Arguments of `load`, `save`, `exist`, `eval`, `input` and `feval` need to be known at compile-time

MDICE (4)

- Illustration of the replacement of the functions `eval` und `diary`:
  - This code ist taken from the client where the computation result is being sent to the server

```
Original PARMATLAB Code:
========================
%%%% SEND ARGUMENTS
disp('Sending output arguments')
sendvar(ip_fid,hostname)
for i=1:func.argout
 eval(['sendvar(ip_fid,' ...
          'argo' int2str(i) ')'])
end
```

```
MDICE Code:
==========
%%%% SEND ARGUMENTS
displog('Sending output arguments');
sendvar(ip_fid,hostname);
for i=1:func.argout,
 sendvar(ip_fid,var_argout(i).data);
end;
```

Experiment set-up

- Computing infrastructure:
  - server machine
    - 1.99 GHz P4, 504 MB, with MATLAB 6.5.0, MATLAB Compiler 3.0, and LCC C compiler 2.4
  - client machines
    - 996 MHz P3 and 730 MHz P3, 128 MB
  - operating system
    - Windows XP Prof.
  - network
    - 100 MBit/s Ethernet

Experiment application

- Monte Carlo Simulation from the ArgeSim comparison of parallel simulations techniques
- A damped second order mass-spring system

$$m\ddot{x}(t) + kx(t) + d\dot{x}(t) = 0$$

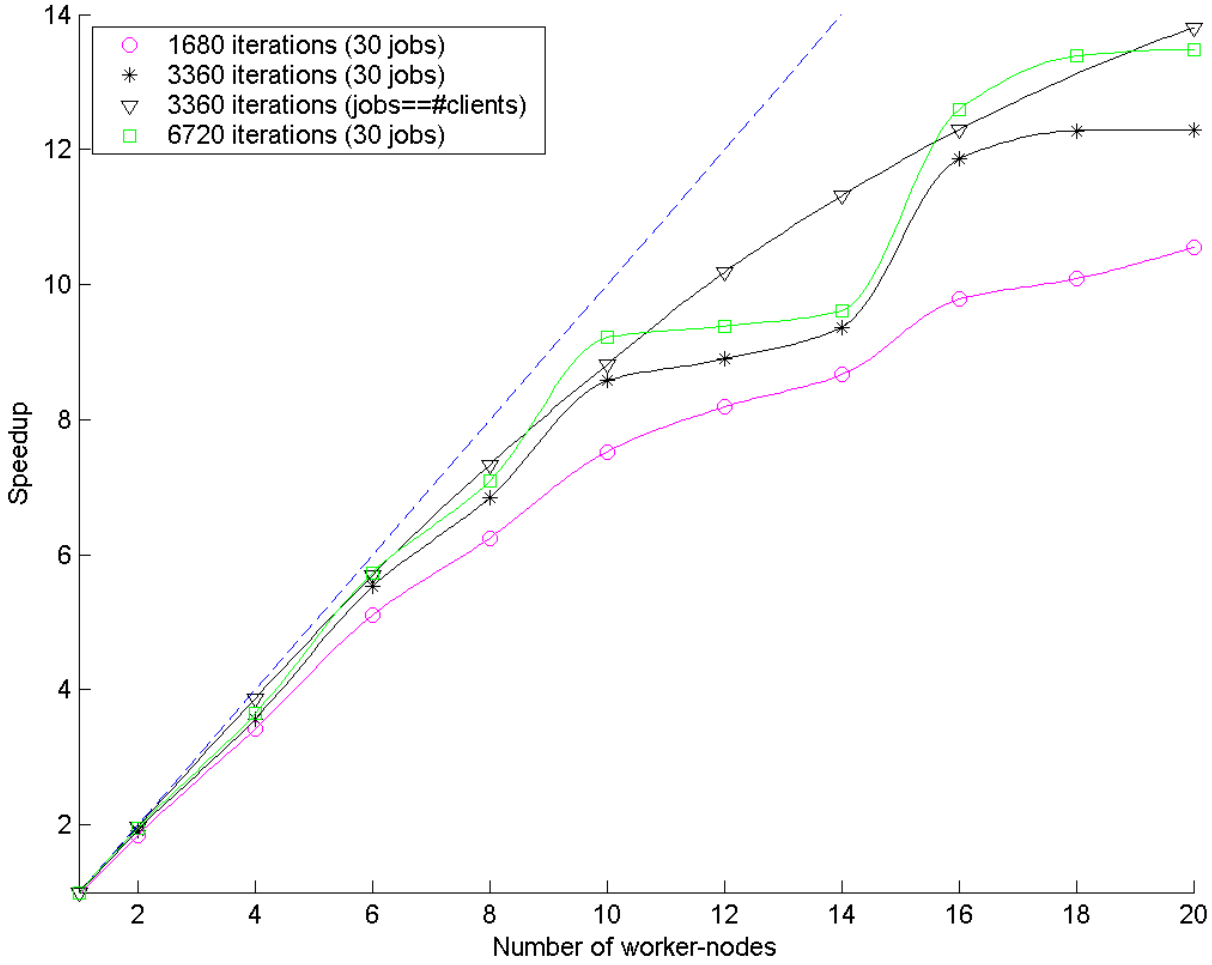
$$\dot{x}(0) = 0, x(0) = 0.1, k = 9000 \; and \; m = 450$$

- The damping factor `d` is a random quantity in the interval [800,1200]
- The solution is the average response over the time interval [0,2] with the step size 0.0005

Experiment Results (1)

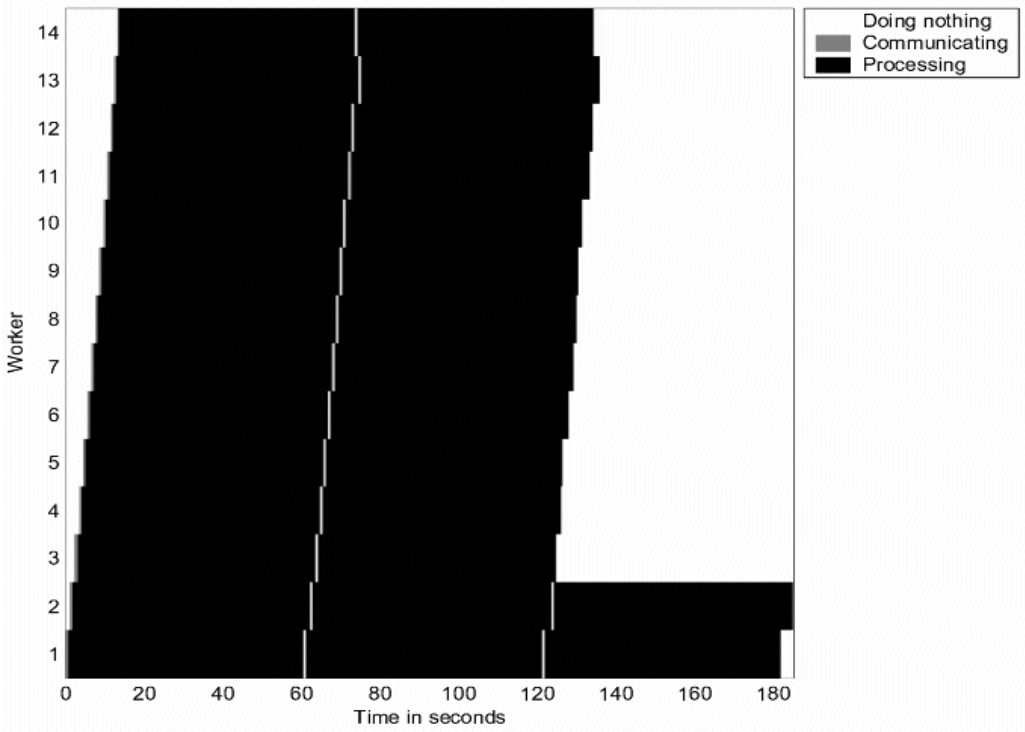
- Speedup of the MC-Simulation on
  - a homogenous environment,
  - when varying the problemsize,
  - keeping the number of jobs distributed among the clients fixed and
  - varying the number of clients

Experiment Results (2)



Experiment Results (3)

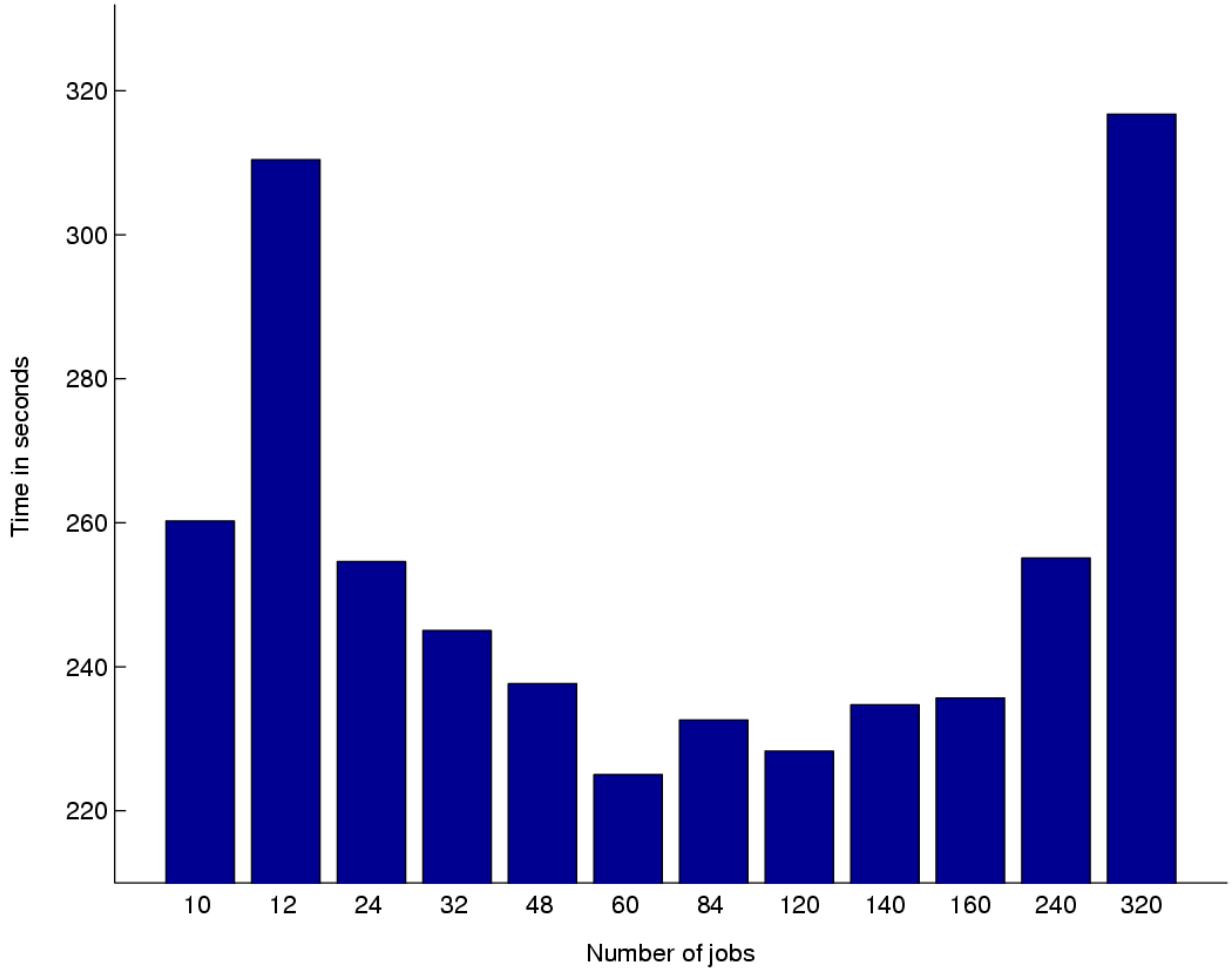
- Visualisation of execution with 14 clients and 30 jobs (6720 iterations)



Experiment Results (4)

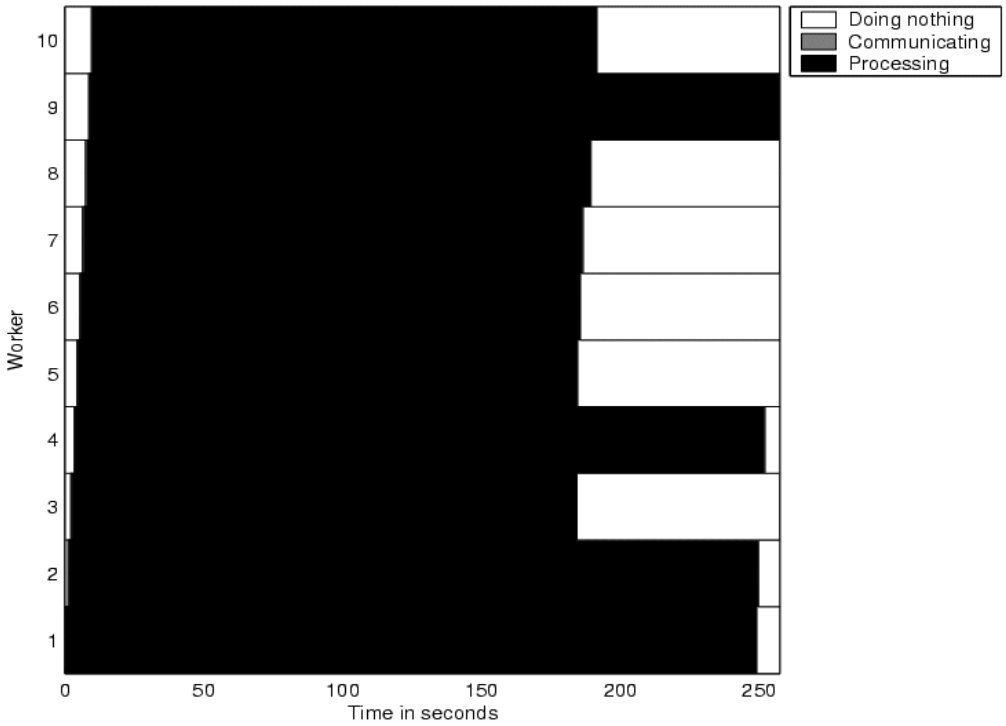
- Execution times on
  - a heterogeneous environment,
  - when keeping the problemsize fixed,
  - varying the number of jobs distributed among the 6 faster and 4 slower clients

Experiment Results (5)



Experiment Results (6)

- Visualisation of execution with 10 clients and 10 jobs (6720 iterations)



Experiment Results (7)

- Effects of interfering application running on MDICE client maschines
- We execute our MC-Simulation code in sequential on two out of four clients running the MC code unter MDICE

Execution time (seconds)	4 jobs	12 jobs	30 jobs
MDICE MC (6720 iter., 4 dedicated clients)	462	463	495
MDICE MC (6720 iter., seq. MC on 2 clients)	915	761	706
Sequential MC (1680 iterations, 452 sec.)	458	458	458

Conclusion

- MDICE may take advantage of the large number of Windows NT based machines
- The most important property is that no MATLAB client is required
- Based on the experimental setup, it proves to be
  - very flexible and
  - efficient in terms of
    - low licencing fees and
    - execution behaviour