UNIVERSITÄT OSNABRÜCK

Parallel Back-Propagation for the Prediction of Time Series

Frank M. Thiesing, Ulrich Middelberg, Oliver Vornberger

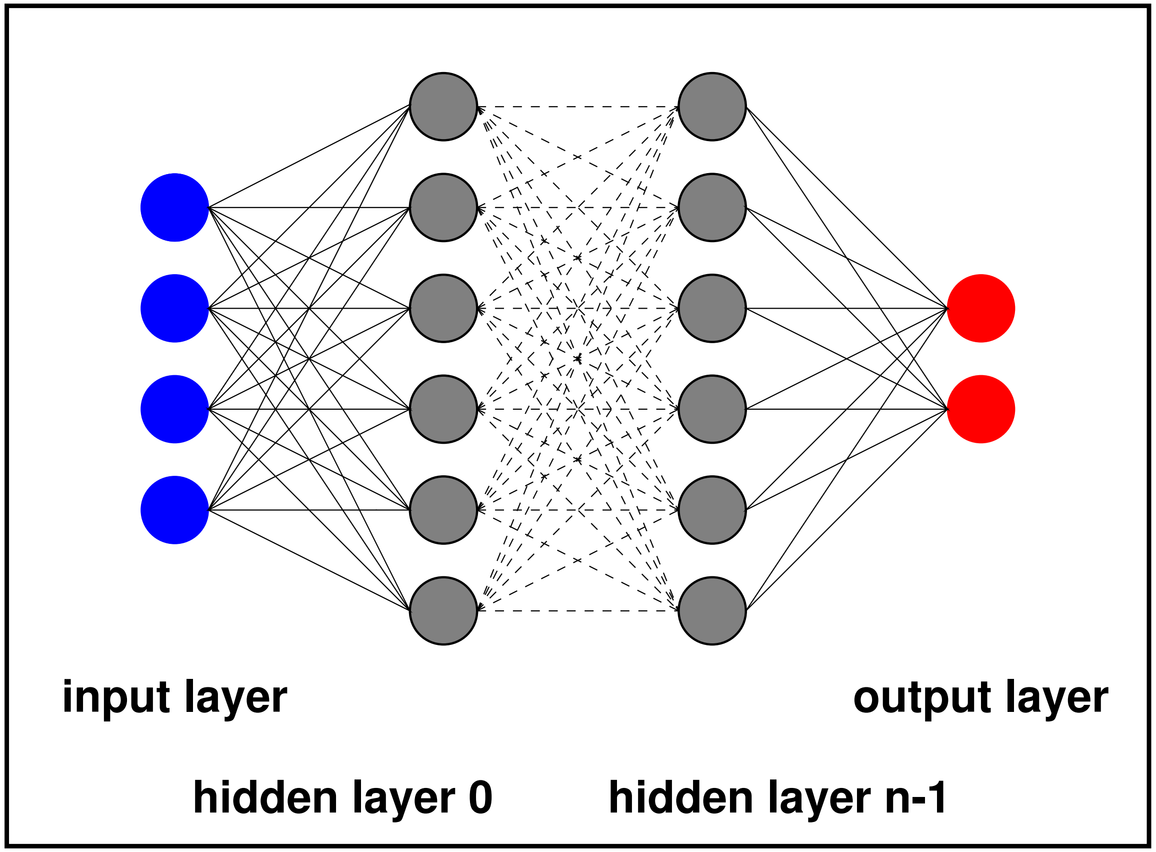
University of Osnabrück
Dept. of Math. / Computer Science
D-49069 Osnabrück
frank@informatik.uni-osnabrueck.de
http://brahms.informatik.uni-osnabrueck.de/prakt/prakt.html

Contents

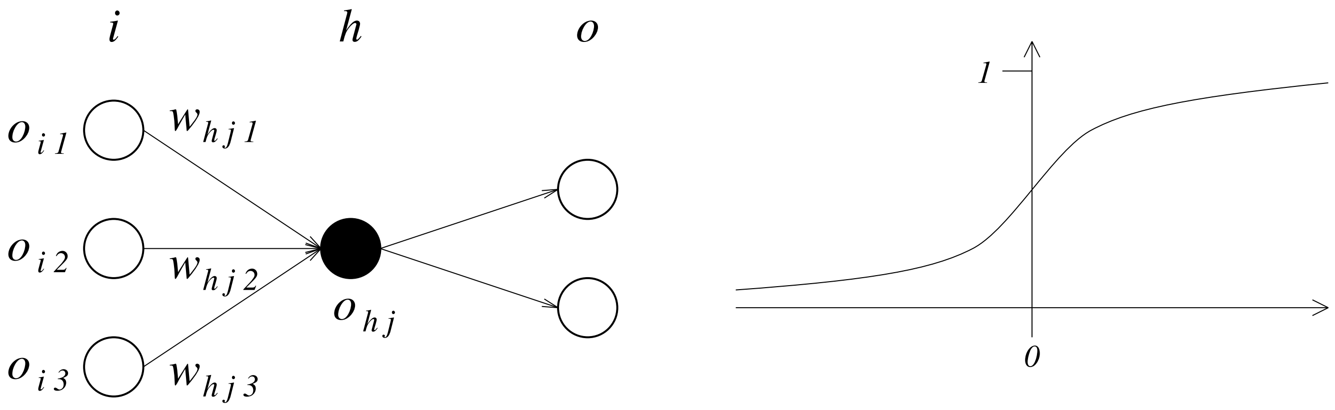
Problem

- analysis and prediction of time series:
  for example consumption of electrical power or goods
- feed-forward multilayer perceptron
- modified back-propagation training
- parallel: PVM, Parix

Feed-Forward Multilayer Perceptron



How a neuron works

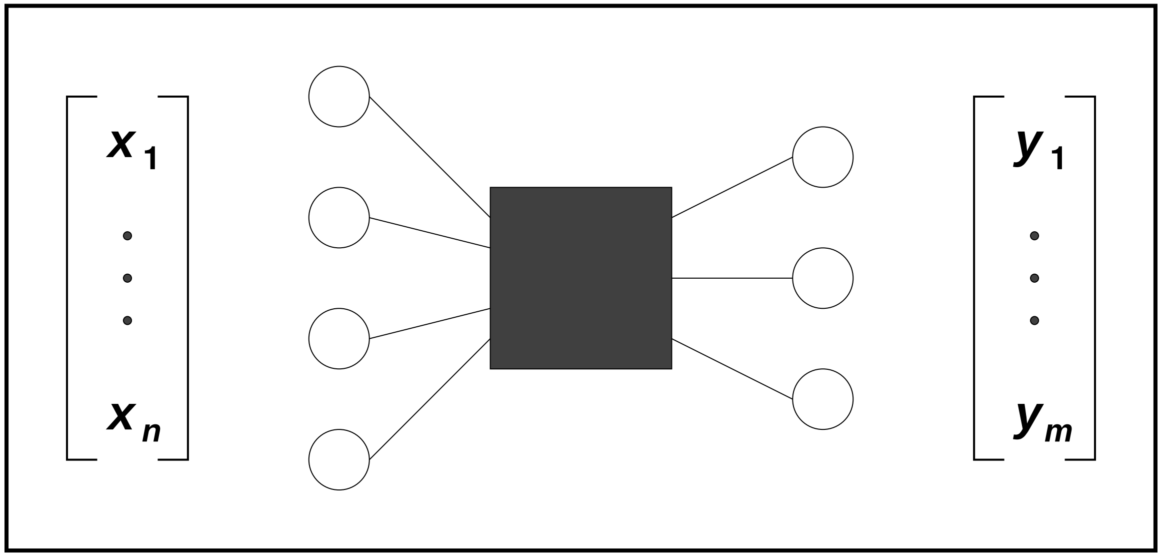


$$o_{hj} = f(o_{i1} w_{hj1} + o_{i2} w_{hj2} + o_{i3} w_{hj3} + \beta_{hj})$$

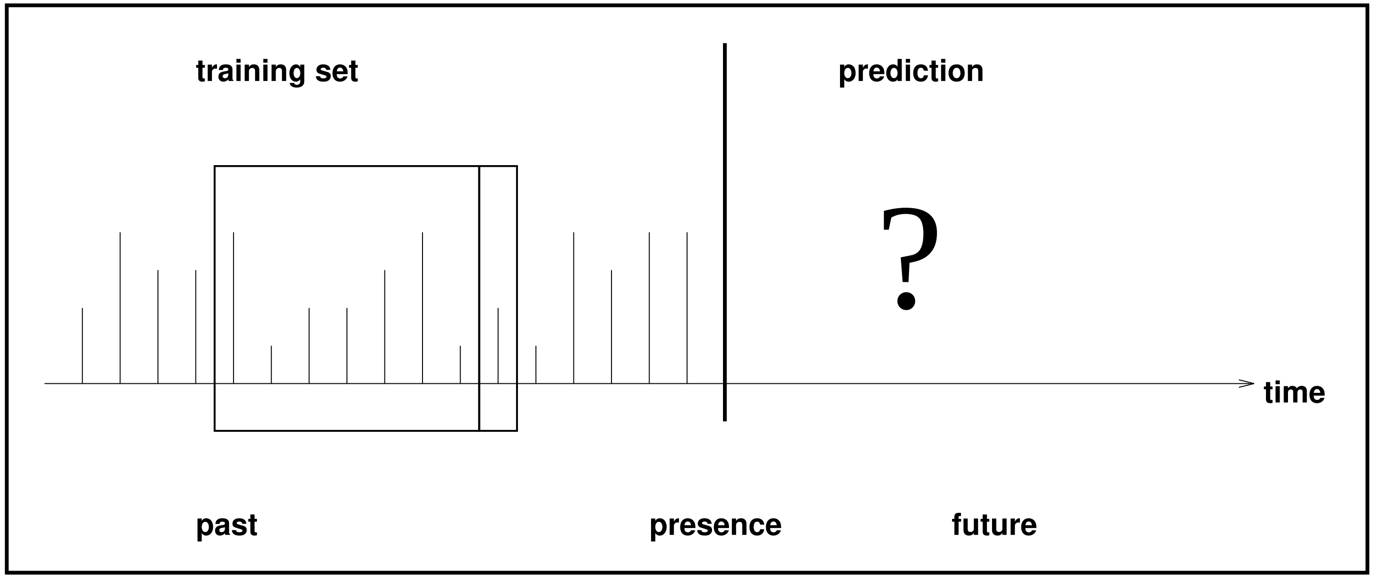
Training and Using ANNs

- presentation of discrete arguments and values of a function
- minimization of errors by correcting the weights

$$g(x_1, ..., x_n) = (y_1, ..., y_m)$$



Prediction of Time Series

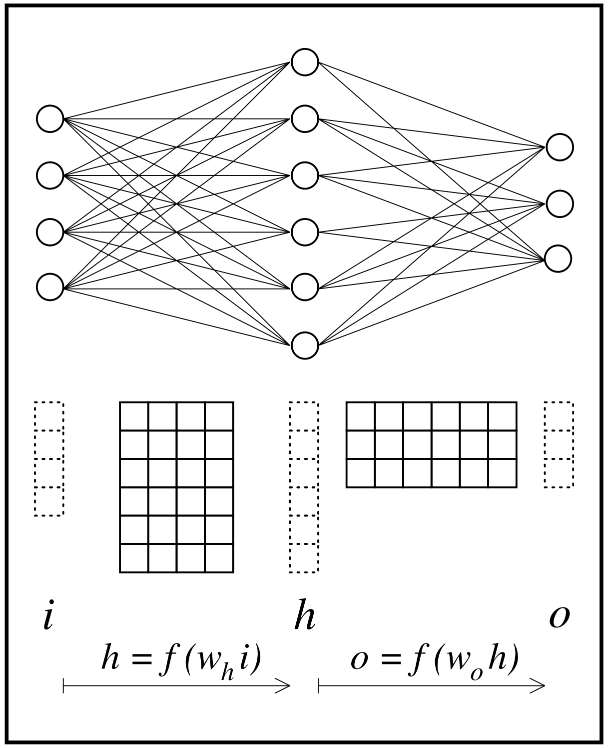


epoch = learn the training set once

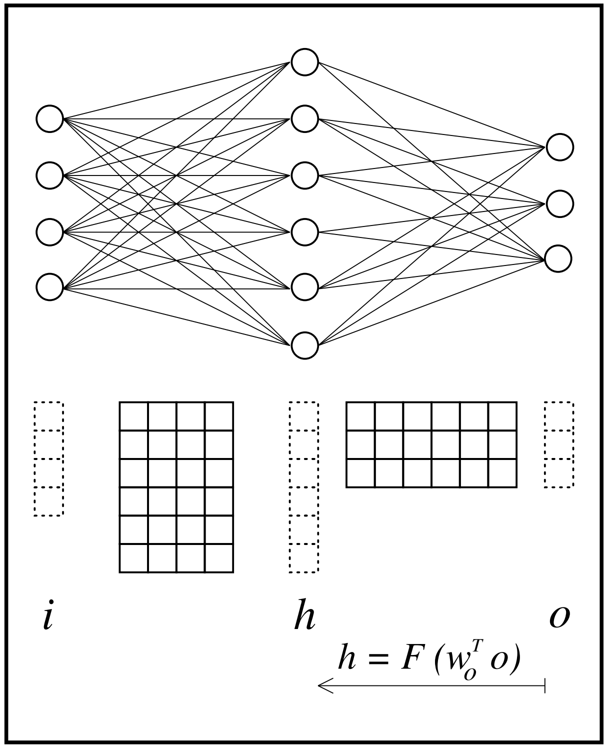
Back-Propagation

- optimization by gradient descent

forward propagation



backward propagation



How many neurons?

- sizes of input and output layers are given
- How many hidden layers?
- How many neurons within a hidden layer?

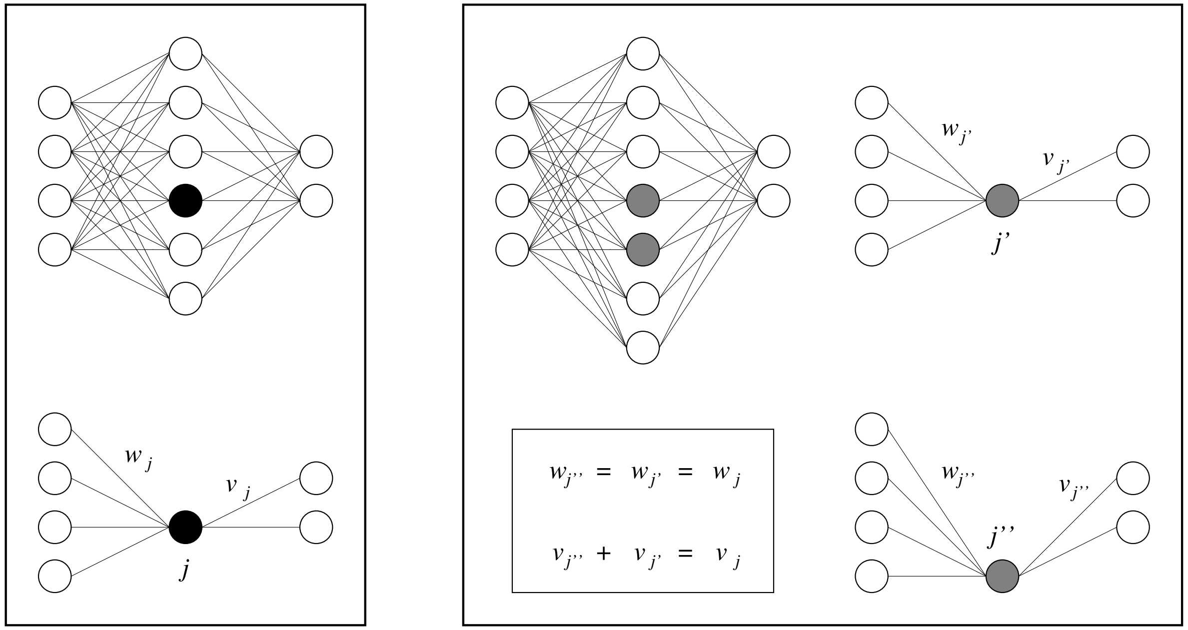
experience:

- better generalization with fewer neurons
- more neurons minimize the error

Modified Back-Propagation

- badly trained and often switching neurons are split from time to time
- description of the net connectivity: `2::4:6(12):2`

before splitting **after splitting**



2 Back-Propagation Training Methods

on-line training

weight corrections within each backward propagation

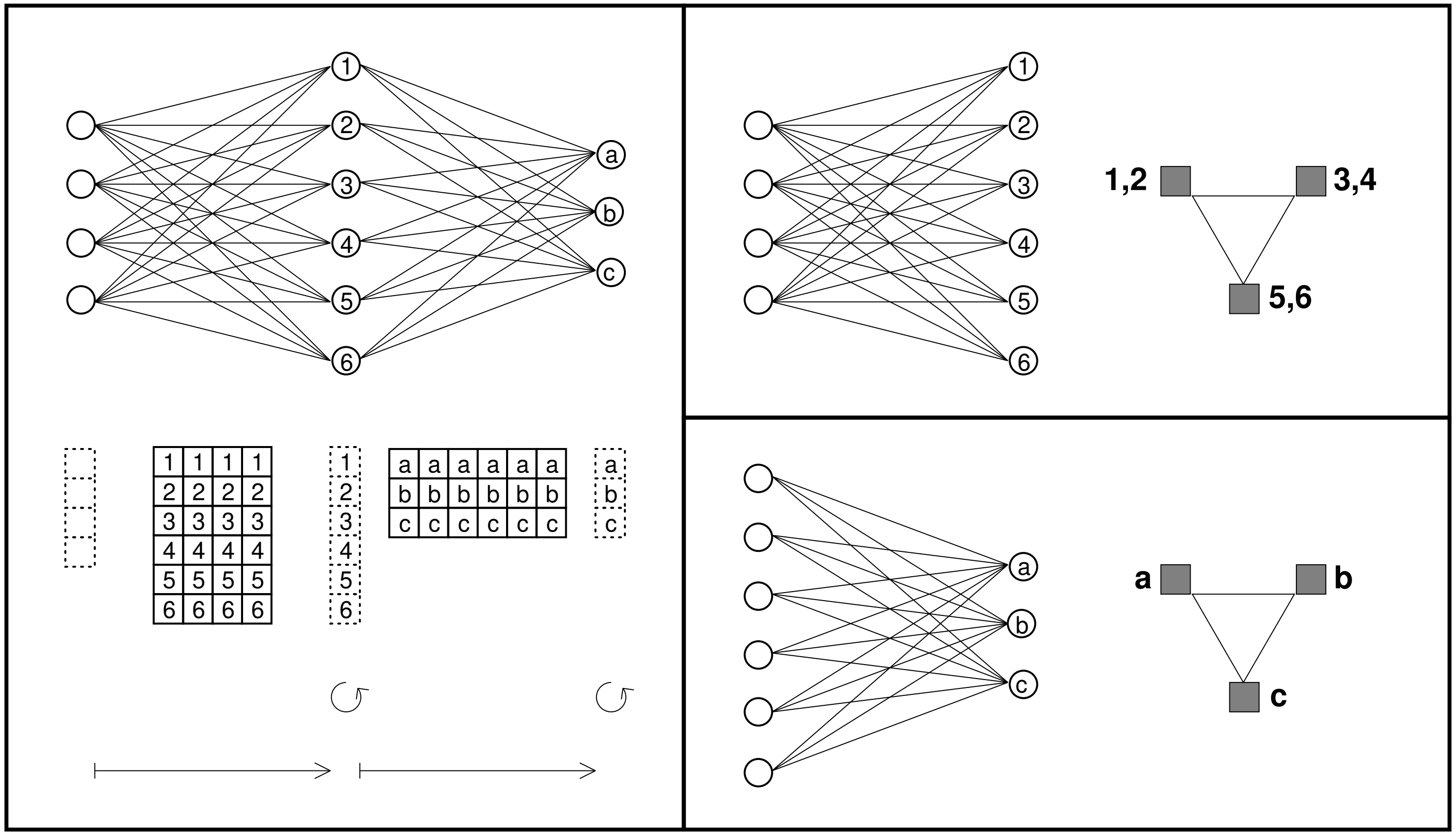
batch learning

accumulation of the weight corrections for one epoch and correction after that

Parallelization: On-Line Training

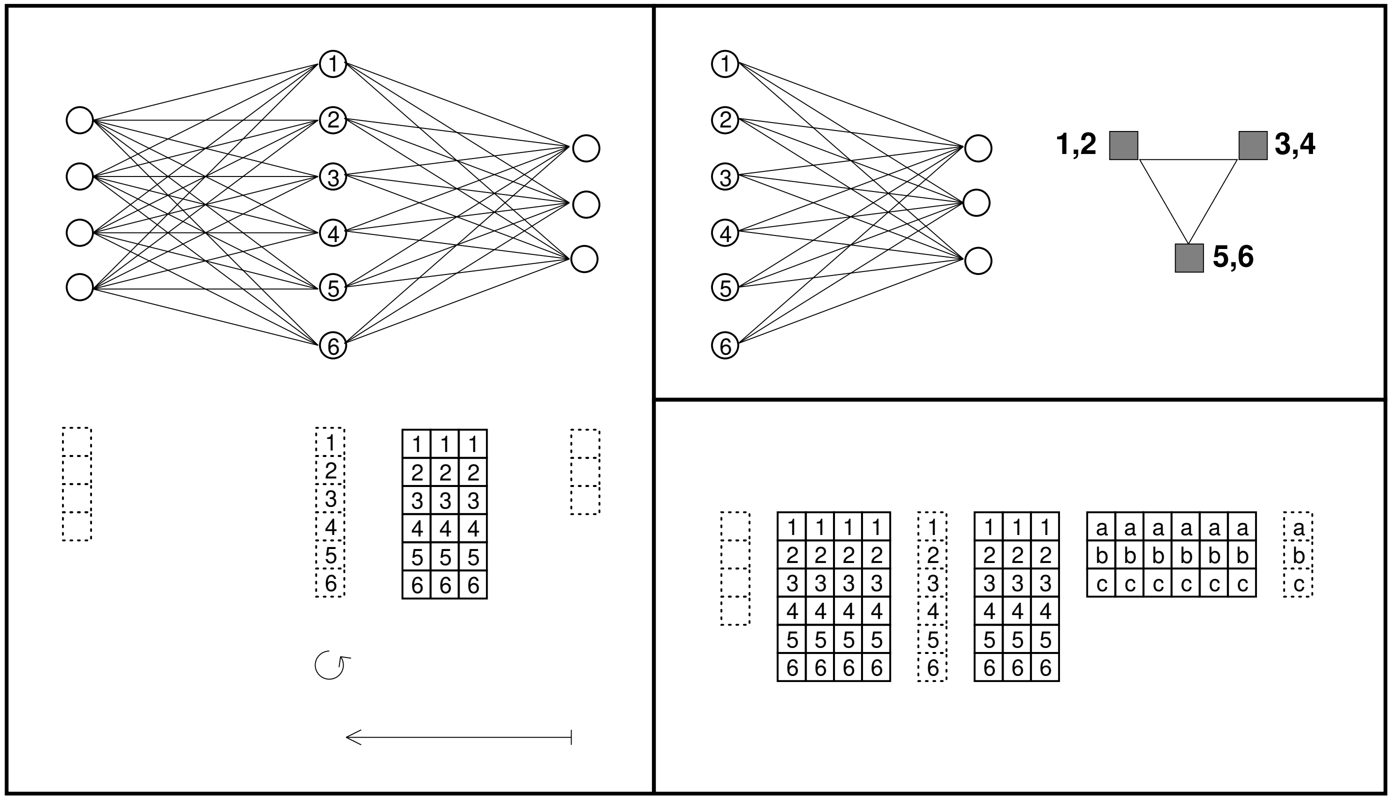
- distribution of the layers under the processors for parallel matrix-vector-operations
- problem: neuron splitting $\Rightarrow$ dynamic distribution of neurons

Parallel On-Line Forward Propagation



Parallel On-Line Backward Propagation

(idea of Yoon, Nang, Maeng)



Parallel On-Line Training

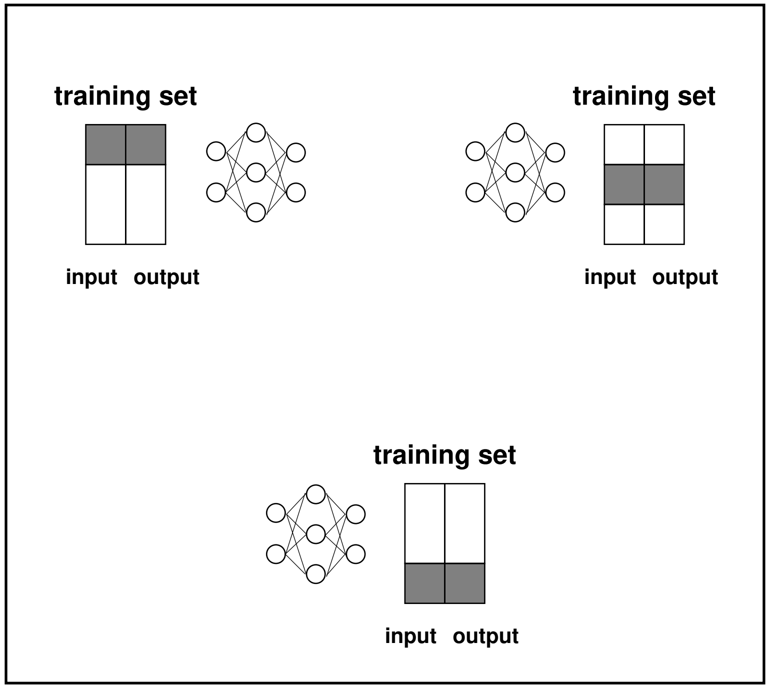
- fine-grained
- high communication demands

  $\Rightarrow$ on tightly-coupled parallel computers

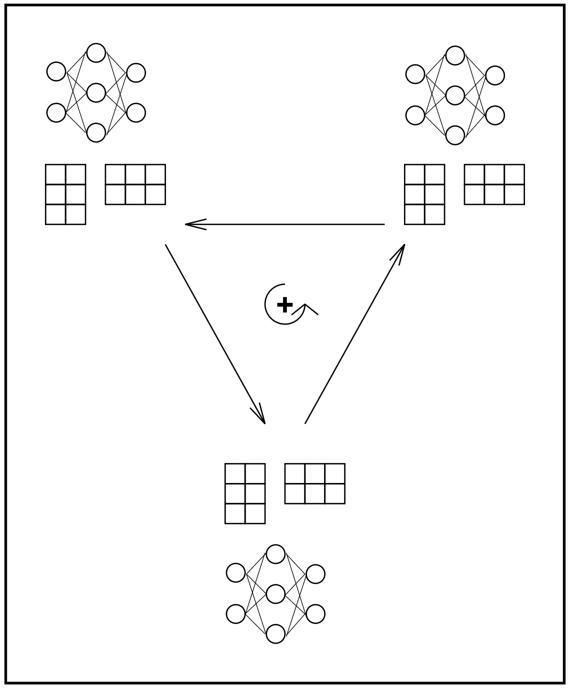
Parallelization: Batch Learning

parallelism by distributing the training set

parallel training



weight corrections



Parallel Batch Learning

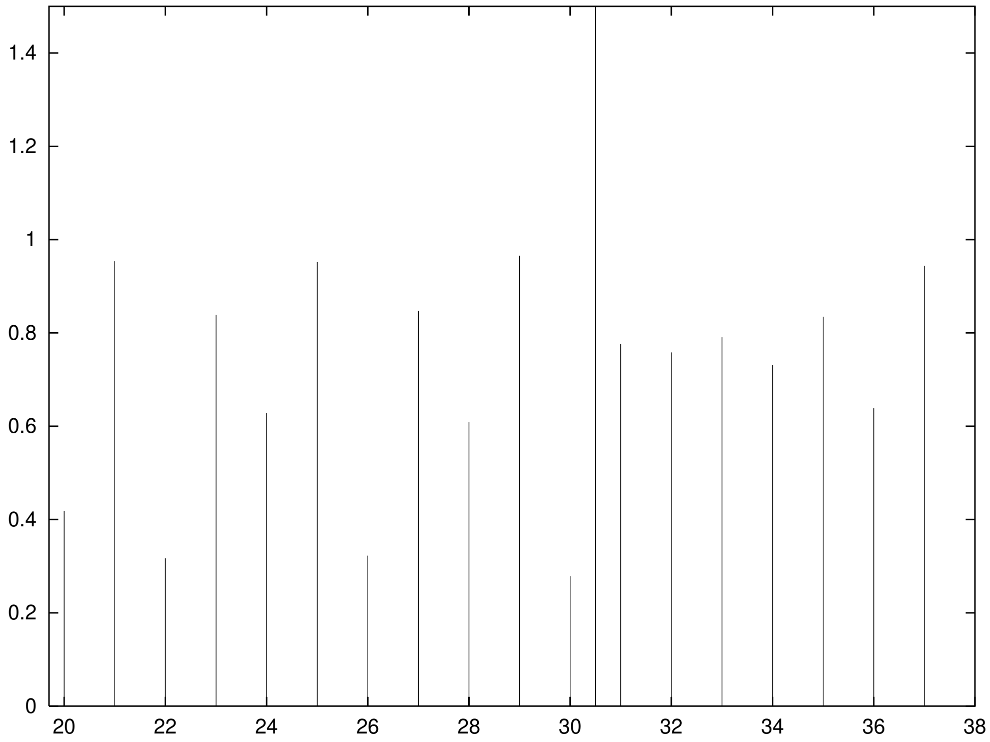
- coarse-grained
- few communication demands

  $\Rightarrow$ on loosely-coupled computer clusters

Test Data

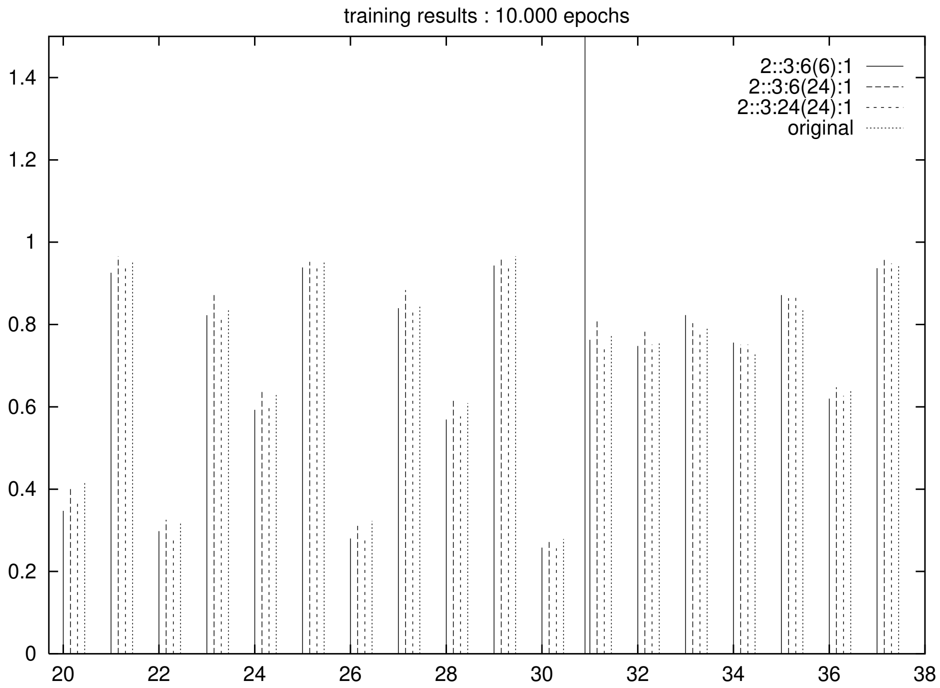
The VERHULST-process: deterministic chaos

$$x_{n+1} = x_n \cdot (1 + r \cdot (1 - x_n)) \qquad \text{with} \qquad r = 2,8$$

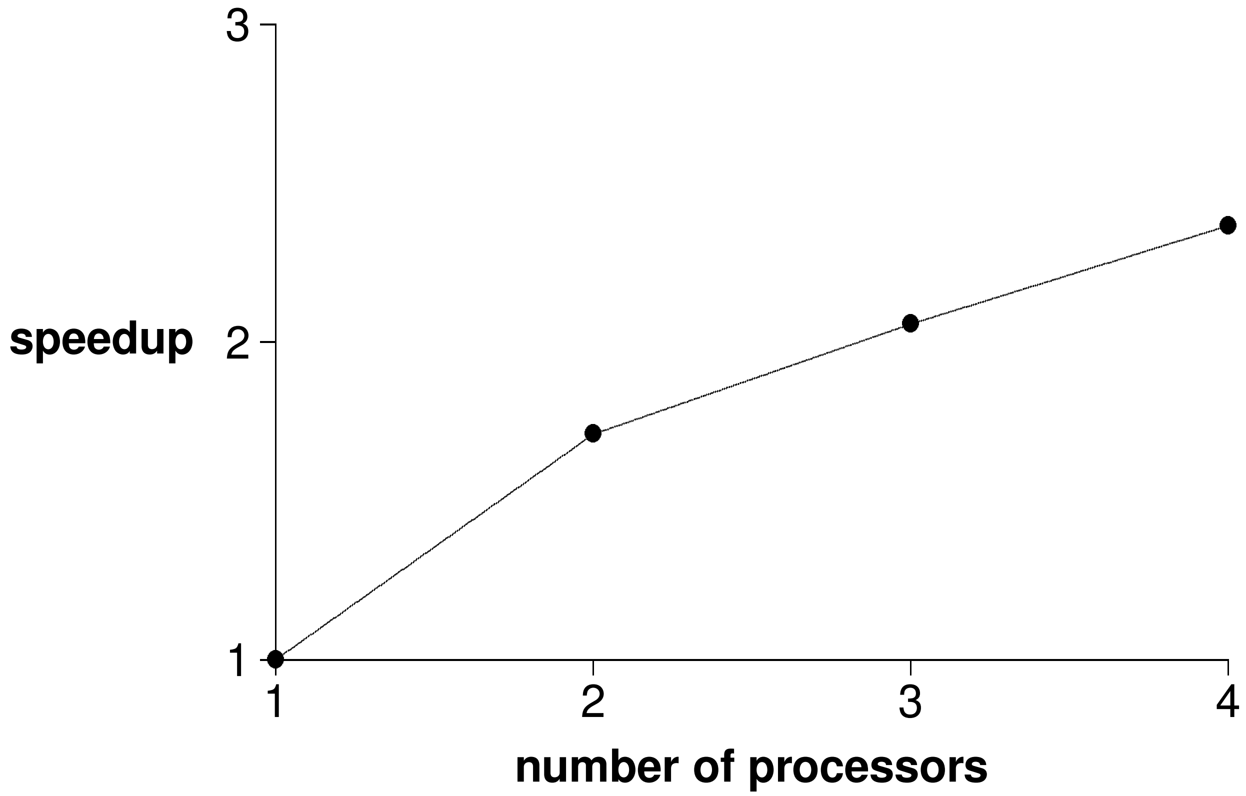


Results of the Forecast

comparison of the training quality with and without neuron splitting

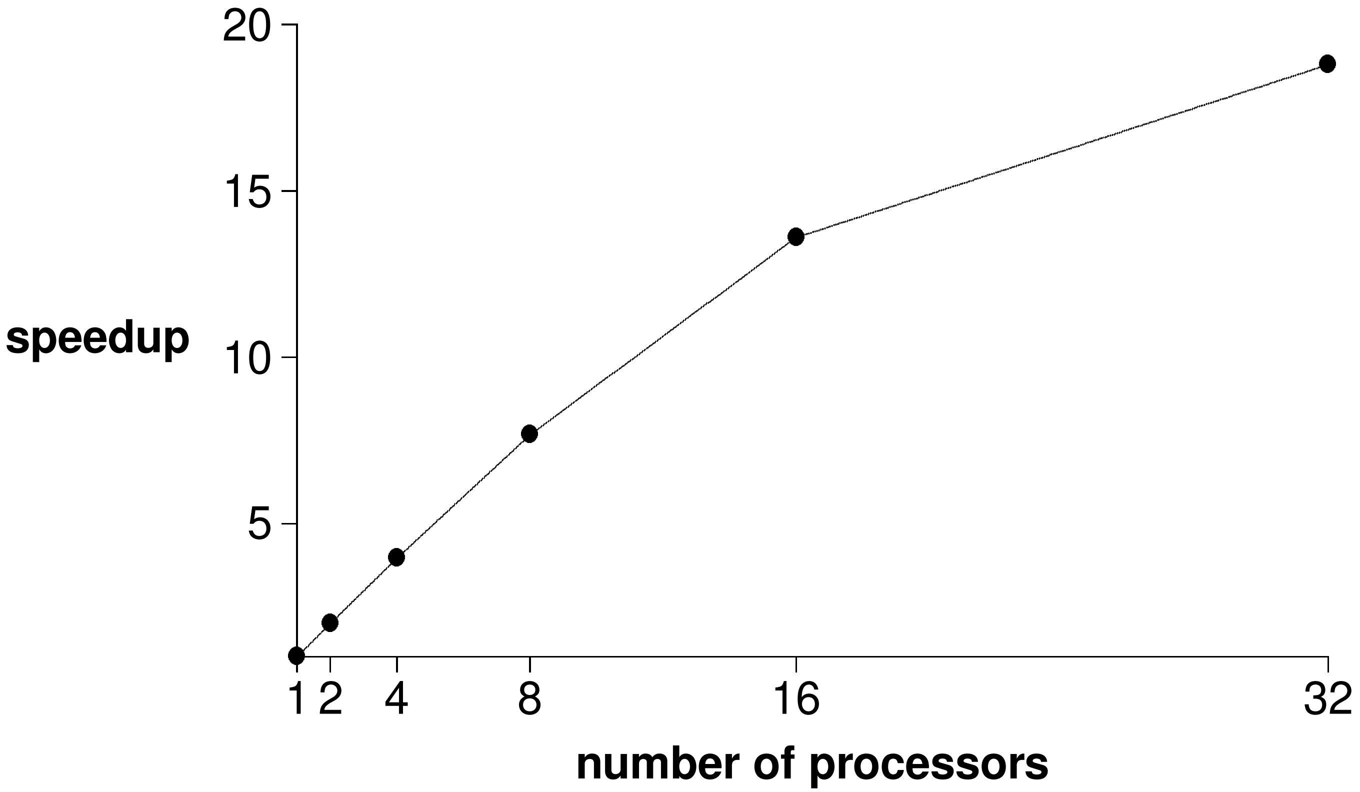


Batch Learning: PVM on Sparc 10



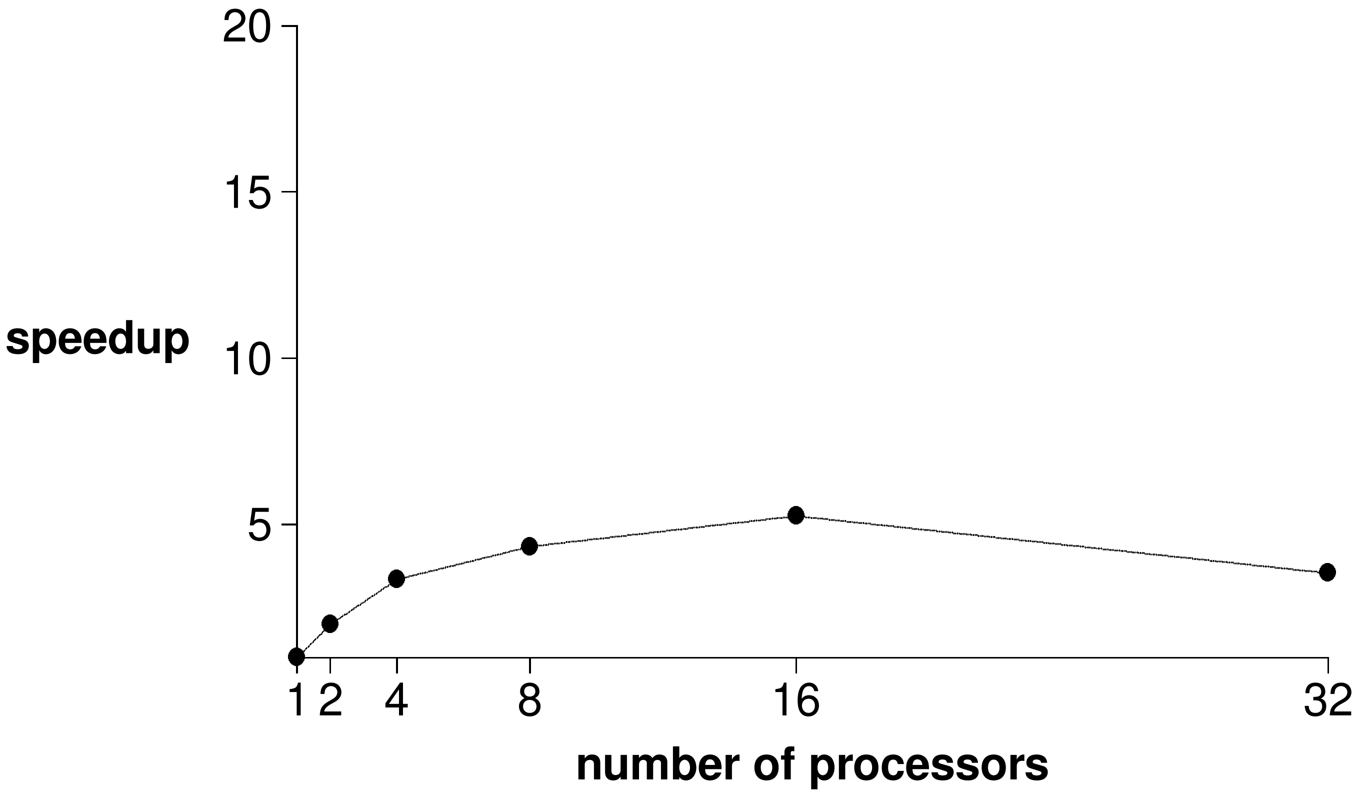
sequential run time: 62.3 seconds

Batch Learning: Parix on T800-Transputers



sequential run time: 159 seconds

Batch Learning: PVM/Parix on T800-Transputers



sequential run time: 159 seconds

Future Work

- implementation of the parallel on-line training
- prediction of real time series from economical processes