Database Algorithms ↔ Networks
Jens Teubner · ETH Zurich, Systems Group
Modern computers are **networks**!

(8-way Intel Nehalem-EX)
Networks are all over the place:

- **data center** networks
- **system-level** networks (QPI, PCIe, …)
- **on-chip** interconnect (*e.g.*, in FPGAs)

We need **topology-aware algorithms**.
communication pattern
distance
synchronization
Joins Over Data Streams:

Current window for Stream R

\[ w_R \]

Current window for Stream S

\[ w_S \]

\[ \text{Task: Find all } \langle r, s \rangle \text{ in } w_R, w_S \text{ that satisfy } p(r, s). \]
Implementation: [Kang et al., ICDE 2003]

1. scan window, 2. insert new tuple, 3. invalidate old

Parallel Execution?
**CellJoin:** [Gedik et al., VLDBJ 2009]

- **1.** Bandwidth bottlenecks
- **2.** Long-distance communication
- **3.** Centralized coordination and memory
Handshake Join:

Streams flow by in **opposite directions**
Compare tuples when they **meet**
Data flow representation → parallelization:

- No bandwidth bottleneck 1
- Communication/synchronization stays local 2
Coordination can now be done autonomously

- no more centralized coordination
- Autonomous **load balancing**
- Lock-free message queues between neighbors
Example: AMD “Magny Cours” (48 cores)
Experiments (AMD Magny Cours, 2.2 GHz)

Results validated by the SIGMOD 2011 repeatability committee.
Topology-awareness:

- communic. pattern
- distance
- synchronization

E.g.,

- look at data flow
- asynchronous oper.

http://www.systems.ethz.ch/projects/avalanche/