Development of Parallel DBMS on the Basis of PostgreSQL

Mikhail Zymbler, Constantin Pan
South Ural State University (SUSU)
Supercomputer Simulation Laboratory (SSL)
Chelyabinsk, Russia

Database Systems Research Group
headed by Prof. Dr. Michael Gertz,
Institute of Computer Science
Ruprecht-Karl University Heidelberg, Germany

Wednesday, May 09, 2012

Outline

- Background
  - South Ural State University (SUSU)
  - SUSU's Supercomputer Simulation Laboratory
- PargreSQL, a parallel DBMS based upon PostgreSQL
  - Partitioned parallelism
  - PargreSQL architecture
  - Current results

South Ural State University

http://www.susu.ac.ru


St .Petersburg
Moscow
Nizhniy Novgorod
Yekaterinburg
Chelyabinsk
Tomsk
Novosibirsk

South Ural State University

- History
  - 1943-1951: Chelyabinsk Mechanics and Engineering Institute
  - 1951-1990: Chelyabinsk Polytechnic Institute
  - 1990-1997: Chelyabinsk State Technical University
  - 1997-now: South Ural State University
- Present
  - 32 faculties
  - 2 700 professors and assistant professors
  - 55 000 students
  - 400 international students
  - 300 programs of higher professional education
  - 200 programs of further education
- Achievements
  - In top 10 Russian Universities
  - National Research University status since 2010

Centers and institutes

- Center of metallurgy and material study
- Center of mechanical engineering
- Nanotechnology research and education center
- Research-manufacturing institute "Educational engineering and technologies"
- Supercomputer Simulation Laboratory




Supercomputer Simulation Laboratory

http://supercomputer.susu.ac.ru/en/

- Supercomputer Center
  - supercomputers administration and software license management
  - research in parallel and distributed computing
  - development of software for grid computing and supercomputer systems
- Distributed Computing and Embedded Systems Department
  - development of software for distributed computing, embedded systems, mobile platforms, electronic resources
- Support and Training Department
  - consultation for the users of the applied and system software
  - training courses based on the SSL resources
- Data Mining and Virtualization Department
  - research in the field of data mining and virtualization technologies, solutions of practical problems based on these technologies, implementation and maintenance of appropriate software

Supercomputer Resources Evolution in SUSU

Physics
Computer cluster

Peak performance
1 Gigaflops
2000

Intel Pentium 3
0,8 GHz

Infinity
Computer cluster

Peak performance
333 Gigaflops
2004

Intel Xeon64
3,2 GHz

SKIF URAL
Computer cluster

Peak performance
16 Teraflops
2008

Intel Xeon E5472
3 GHz

SKIF-Aurora SUSU
Supercomputer

Peak performance
24 Teraflops
2010

Intel Xeon x5570
2,93 GHz

SKIF-Aurora SUSU (upgraded)
Supercomputer

Peak performance
117 Teraflops
2011

Intel Xeon x5680
3,33 GHz

SKIF-Aurora SUSU Supercomputer

- Russian TOP50 list: **4th** (Apr 2012)
- World TOP500 list: **121st** (Nov 2011)
- Peak performance: **117** Teraflops
- Qty of computing cores: **8832**
- RAM: **9** TByte
- Disk space: **Intel SSD 108** TByte

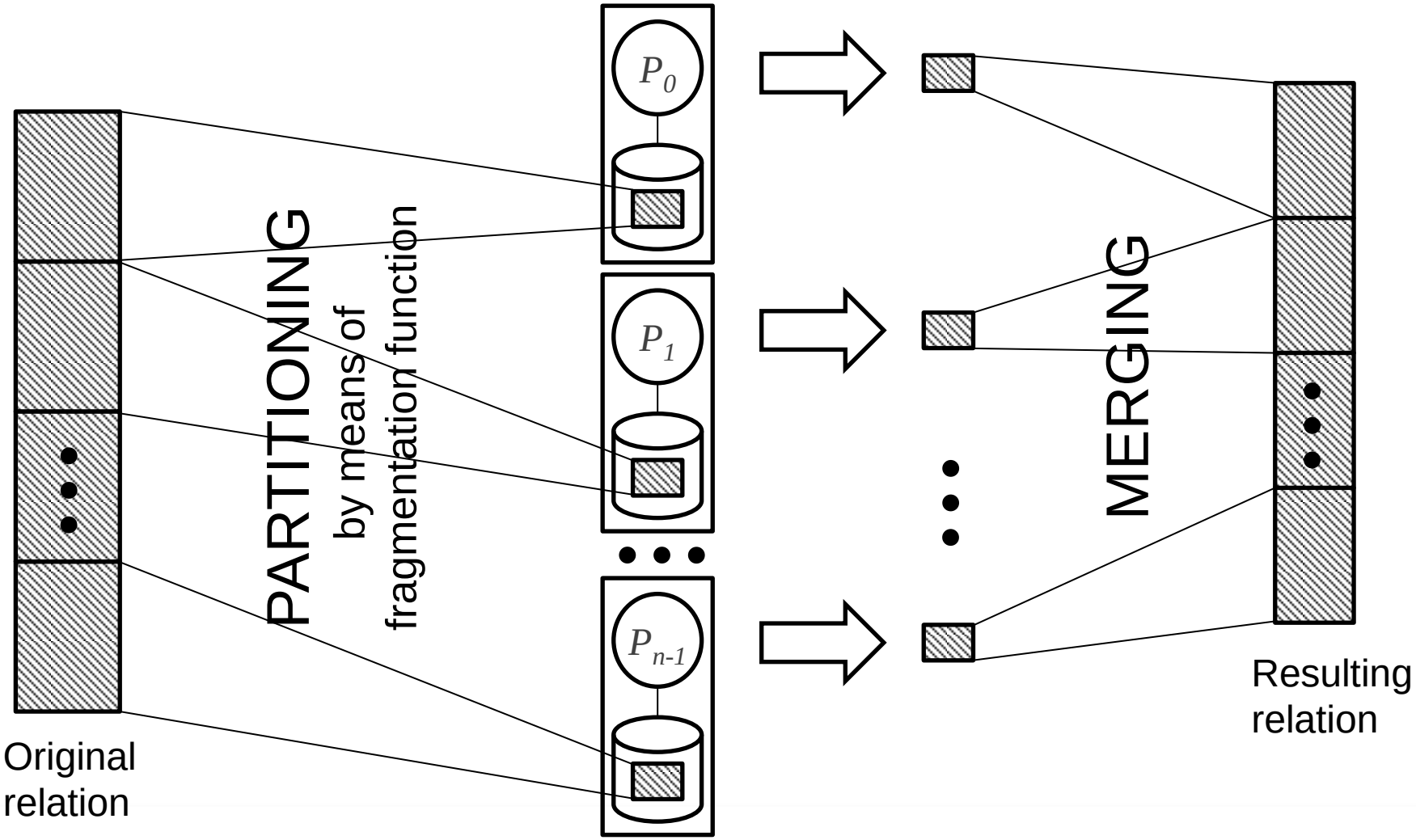
Communication networks

- System network: 3D torus, **60** Gbit/s
- InfiniBand QDR, **40** Gbit/s
- Gigabit Ethernet

Omega project http://omega.susu.ru

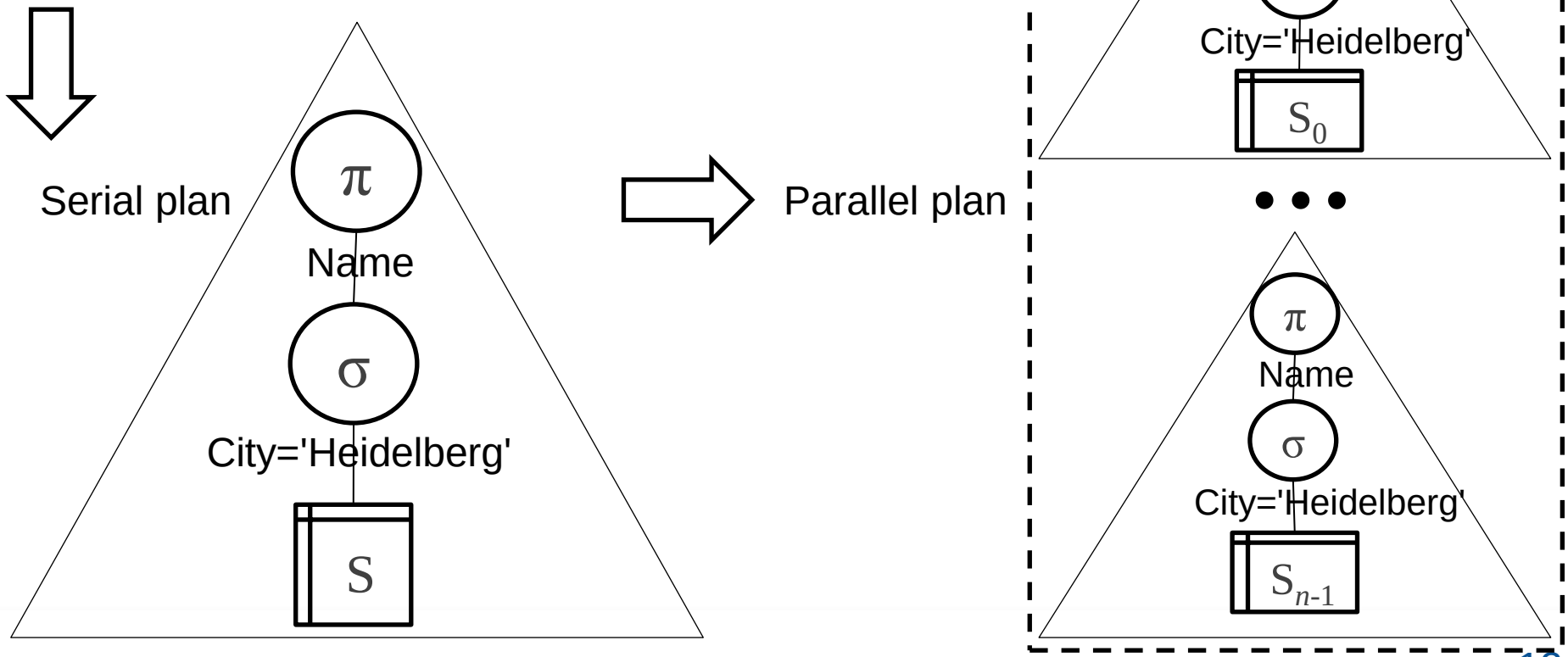
- *Goal*: development of a prototype of parallel RDBMS for cluster systems
  - Based on partitioned parallelism and EXCHANGE operator
  - Testbed of various research ideas (data replication, load balancing etc.)
- *Grants*: financially supported by the Russian Foundation for Basic Research
- *Outcome (since 1997)*
  - *Papers*: more than 60 and 10 papers in recognizable Russian and International scientific proceedings/journals, respectively
  - *Talks*: more than 50 and 10 talks at Russian and International scientific conferences (incl. ADBIS, DASFAA, DEXA), respectively
  - *Dissertations*: 1 Dr. of Science, 4 Cand. of Science

Partitioned parallelism



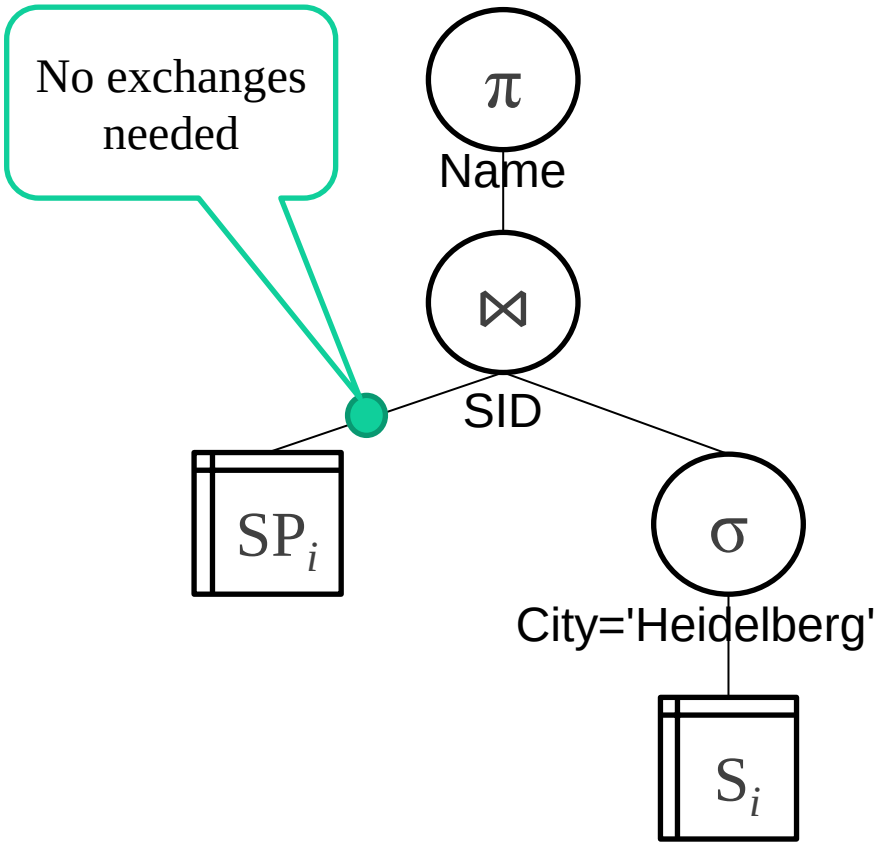
Serial vs parallel query plan

```
SELECT Name
FROM S
WHERE City='Heidelberg'
```



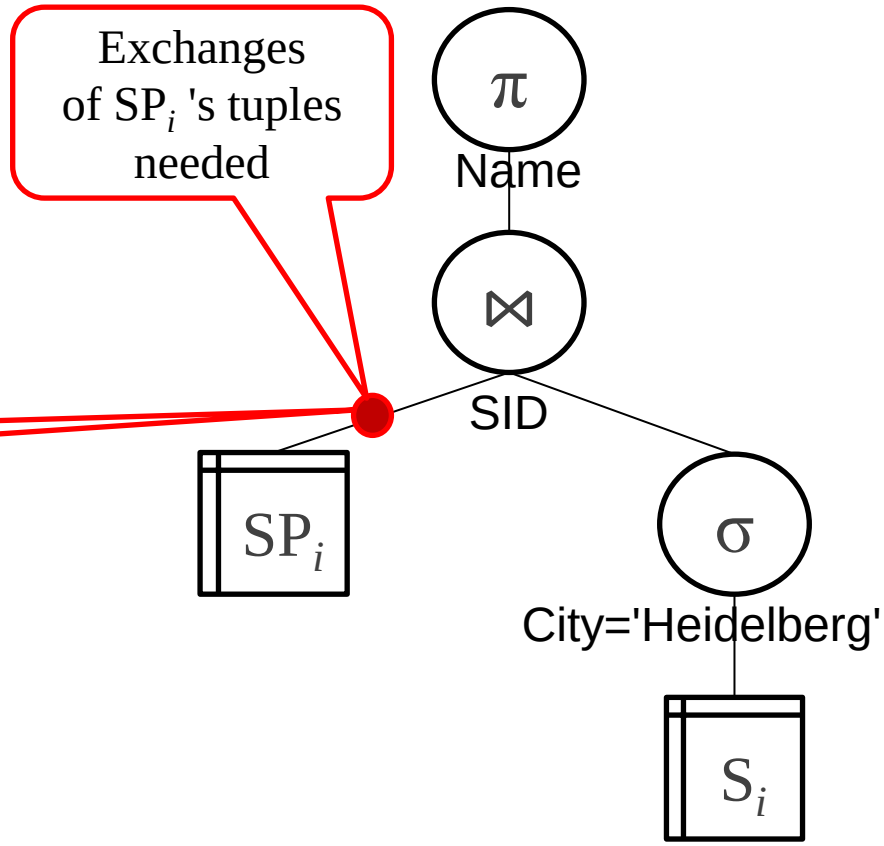
Exchanges of tuples: not needed

- P (PID, Name) – parts
- S (SID, Name) – suppliers
  - partitioned using SID
  - $\varphi_S(t) = t.SID \bmod N$
- SP(SID, PID, Qty) – supplies
  - partitioned using SID
  - $\varphi_{SP}(t) = t.SID \bmod N$
- SELECT Name
  FROM S, SP
  WHERE S.SID=SP.SID and
  S.City='Heidelberg'

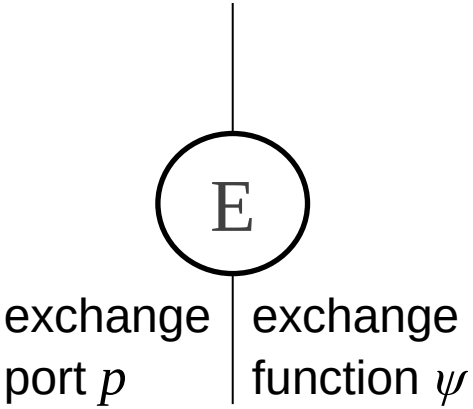


Exchanges of tuples: needed

- P (PID, Name) – parts
- S (SID, Name) – suppliers
  - partitioned using SID
  - $\varphi_S(t) = t.SID \bmod N$
- SP(SID, PID, Qty) – supplies
  - **partitioned using PID**
  - $\varphi_{SP}(t) = t.PID \bmod N$
- SELECT Name
  FROM S, SP
  WHERE S.SID=SP.SID and
  S.City='Heidelberg'



EXCHANGE operator

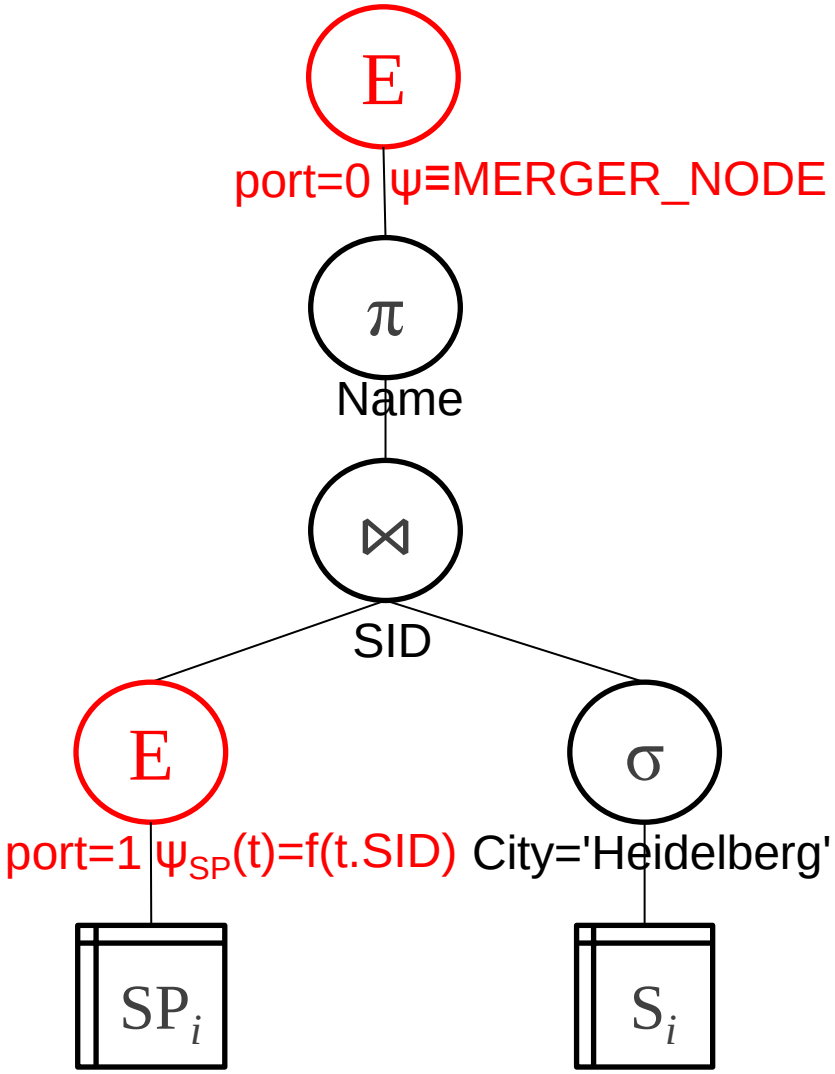


- Exchange port p means ID to differ such operators.
- Exchange function ψ returns a number of node where tuple should be processed.
- Pseudo code

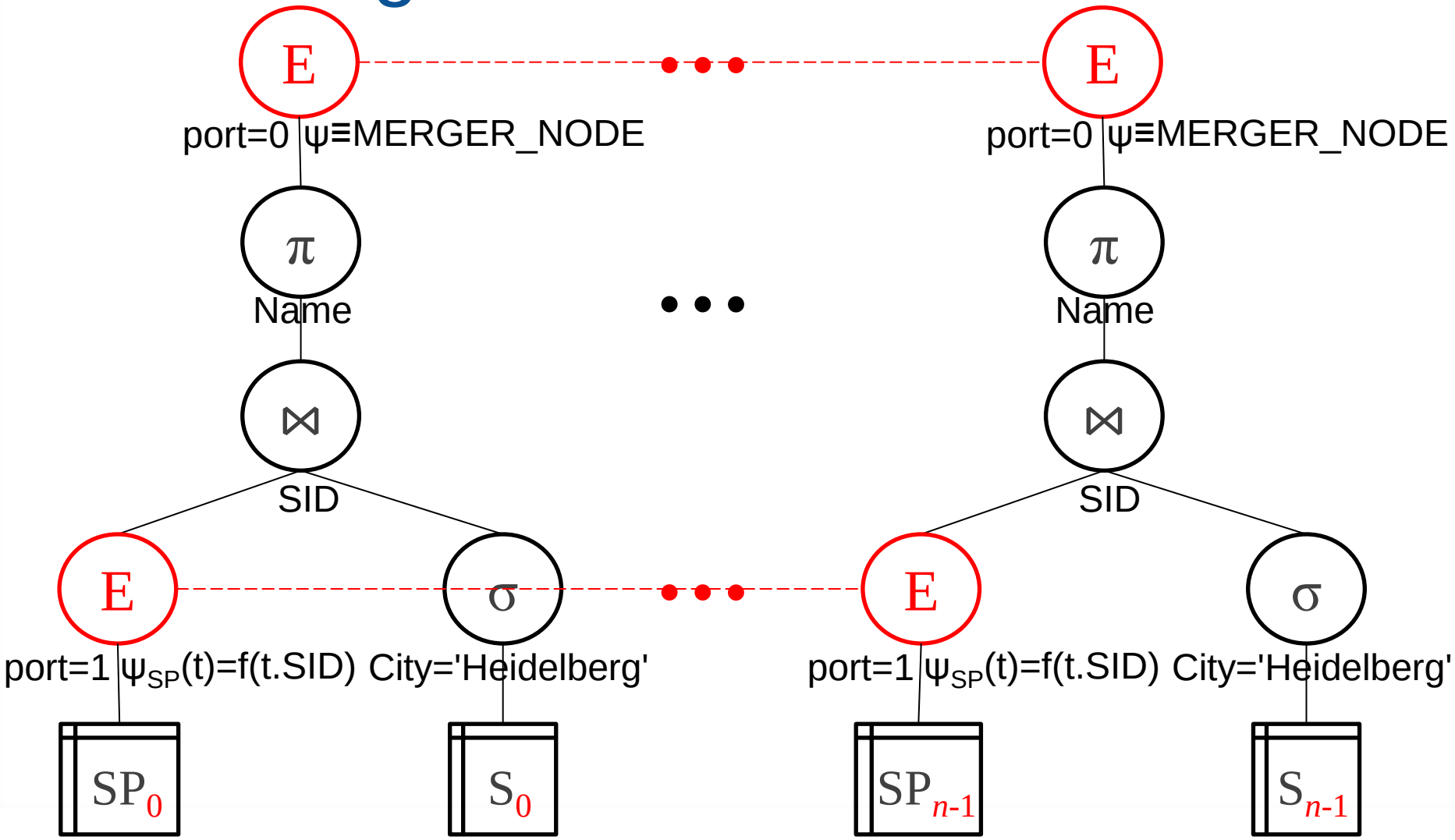
```
if (ψ(t)==Mynode())
   Put(t, this_output_buffer);
else {
   Send(t, ψ(t));
   Put(t, that_output_buffer);
}
```

Parallel agent

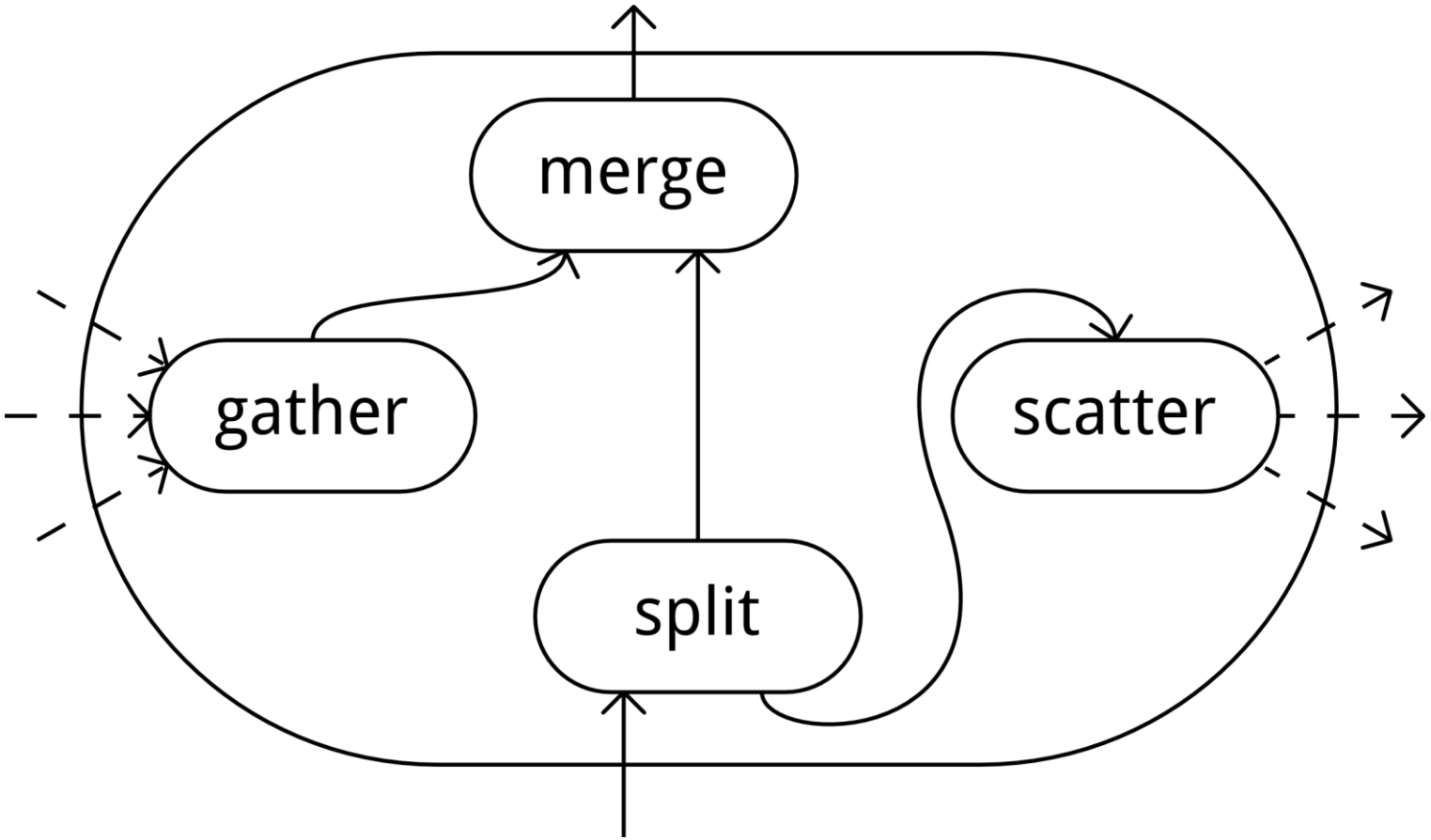
- P (PID, Name) – parts
- S (SID, Name) – suppliers
  - partitioned using SID
  - $\varphi_S(t) = t.SID \bmod N$
- SP(SID, PID, Qty) – supplies
  - partitioned using PID
  - $\varphi_{SP}(t) = t.PID \bmod N$
- SELECT Name
  FROM S, SP
  WHERE S.SID=SP.SID and
  S.City='Heidelberg'



Parallel agents

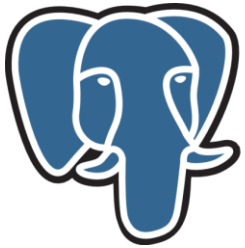


EXCHANGE operator

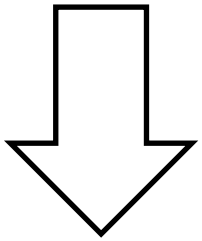


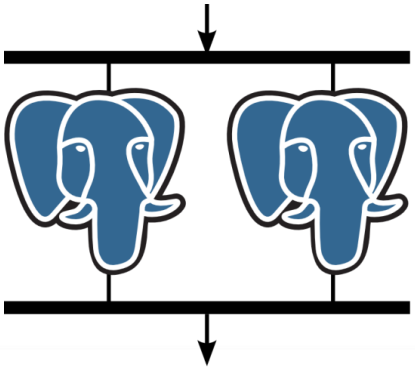
PargreSQL project

- PargreSQL = PostgreSQL + Partitioned parallelism



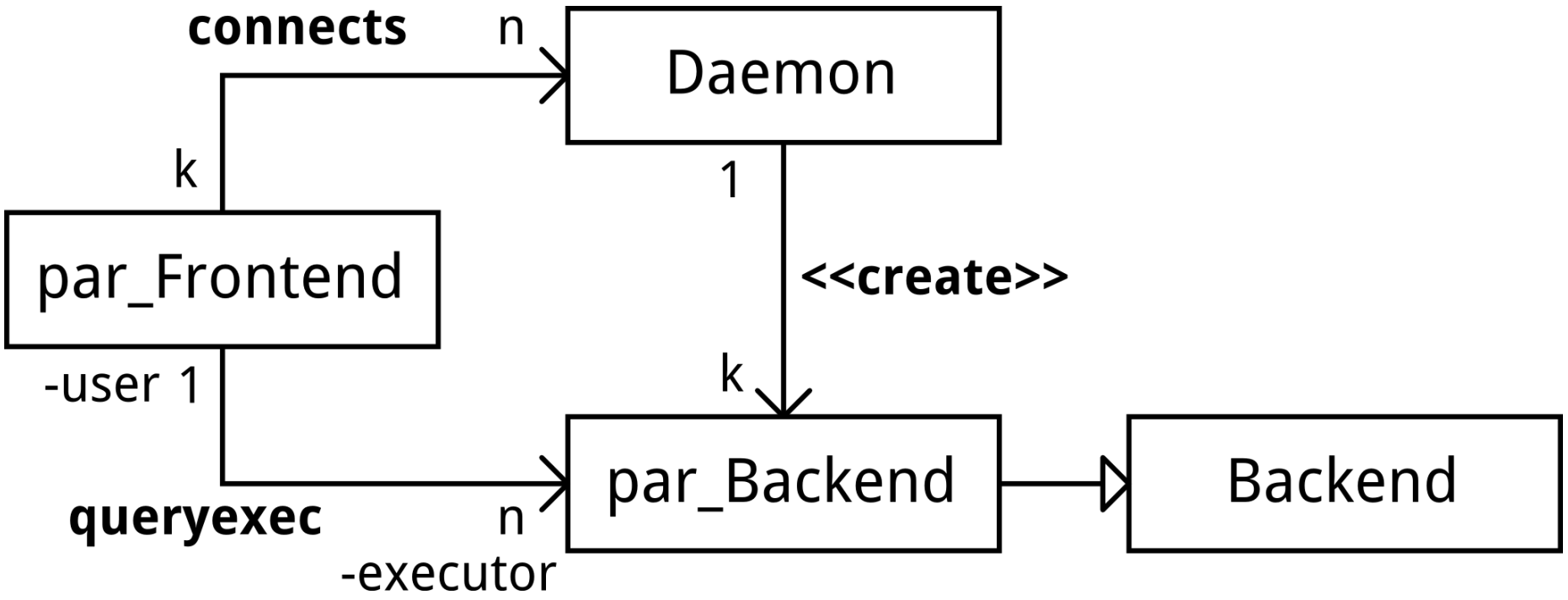
PostgreSQL



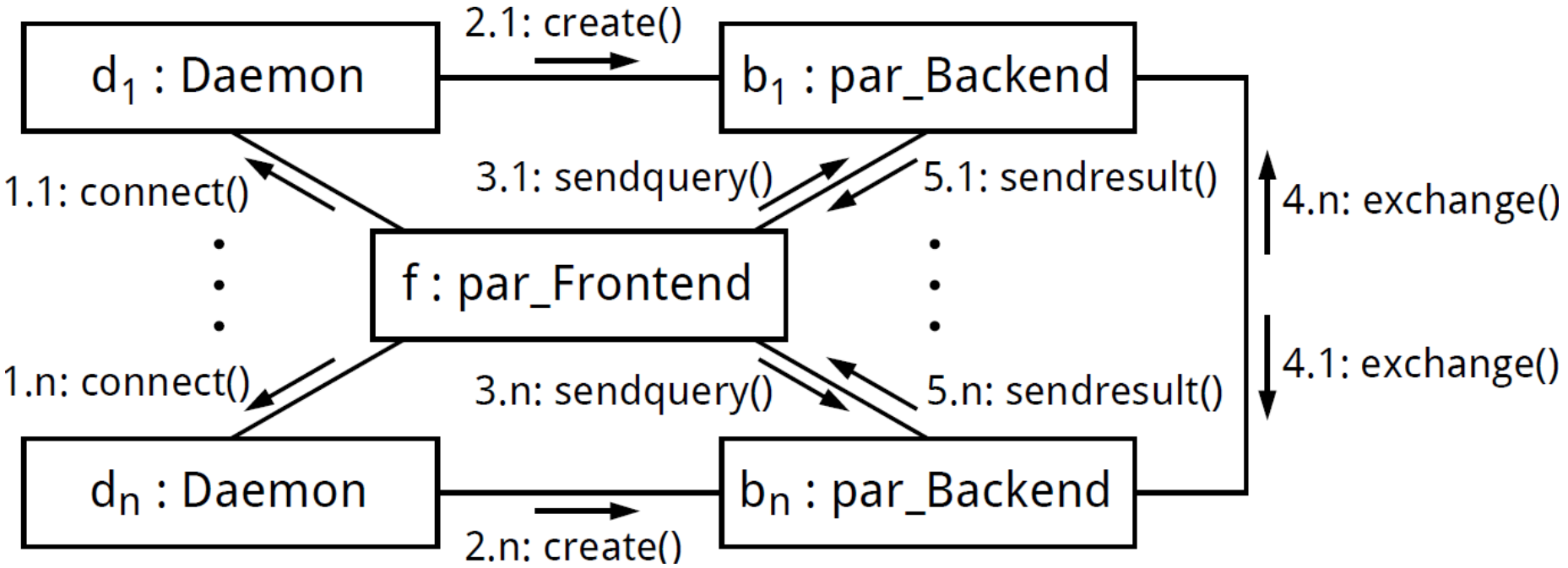


PargreSQL

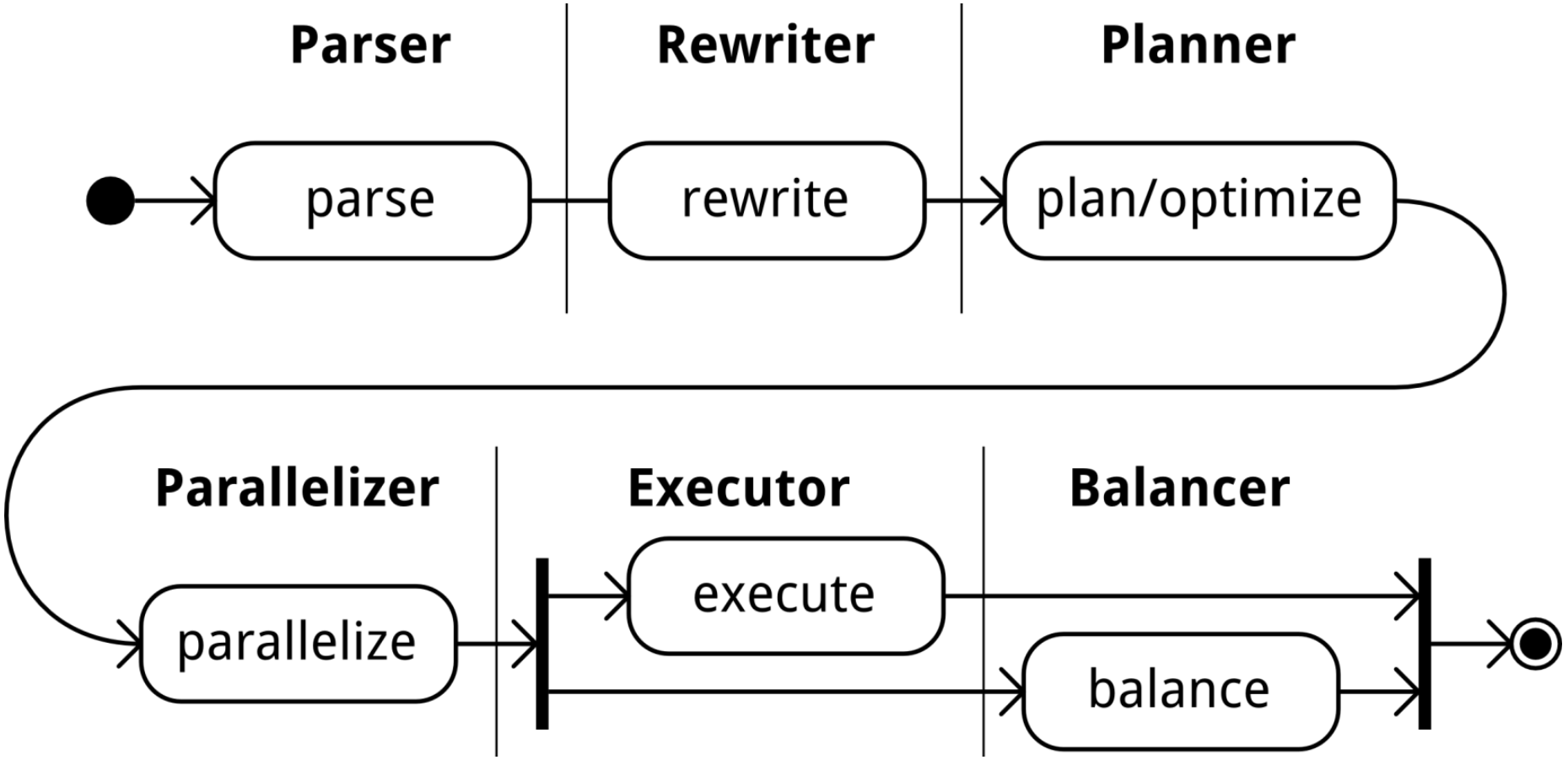
DBMS processes: PargreSQL



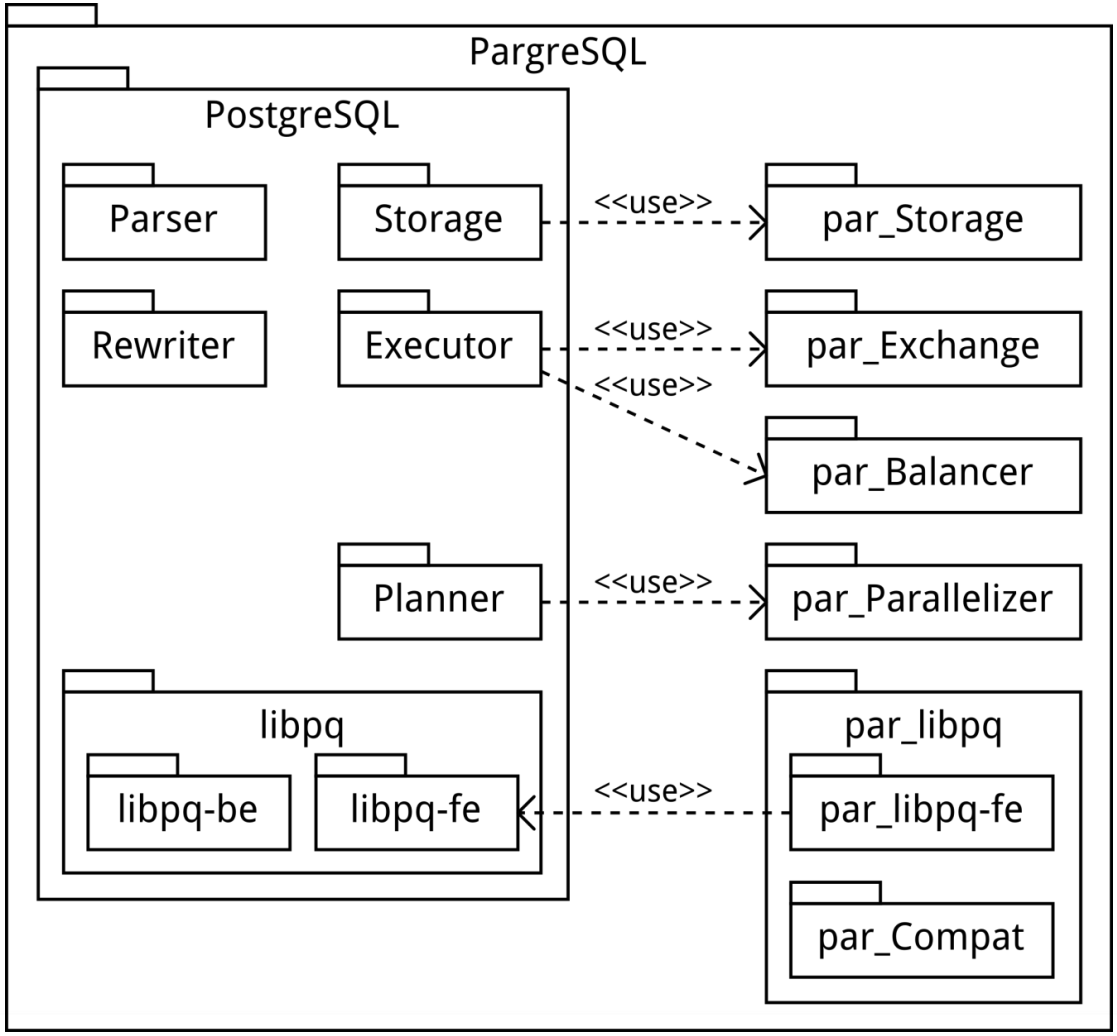
DBMS processes: PargreSQL



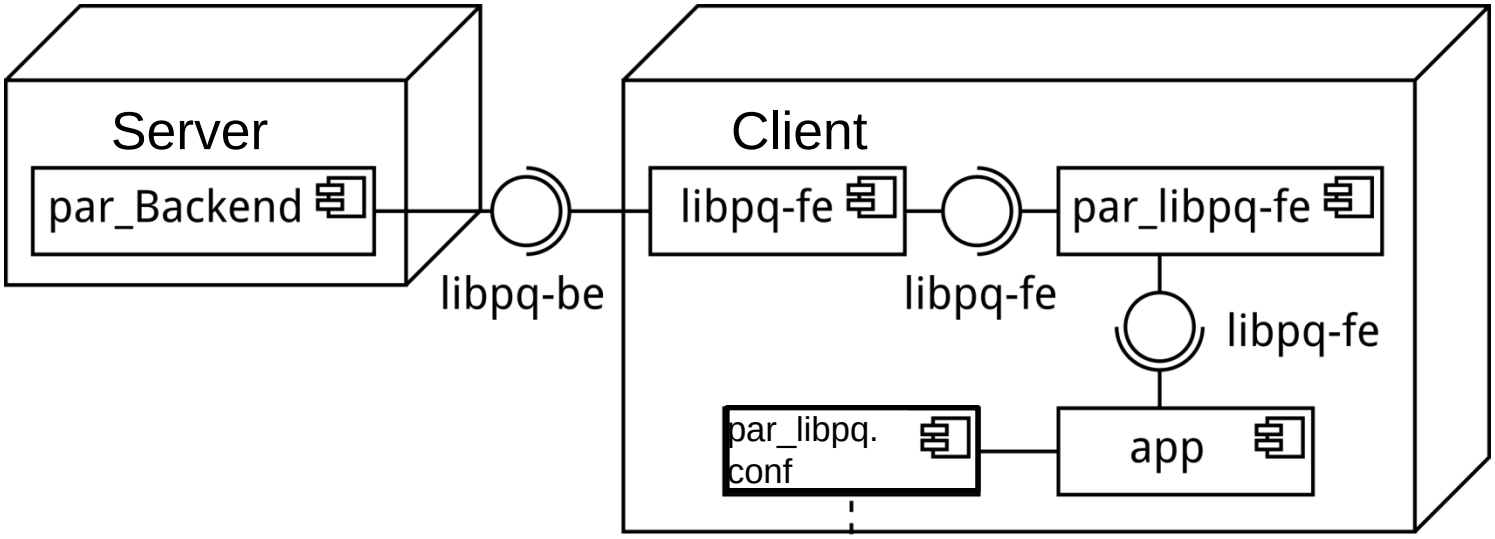
Query processing: PargreSQL



DBMS architecture: PargreSQL



Components deployment: PargreSQL




```
dbname=postgres   hostaddr=10.1.6.16   port=5432
dbname=postgres   hostaddr=10.1.7.1    port=5432
dbname=postgres   hostaddr=10.1.7.3    port=5432
dbname=postgres   hostaddr=10.1.7.4    port=5432
dbname=postgres   hostaddr=10.1.7.5    port=5432
dbname=postgres   hostaddr=10.1.7.6    port=5432
```

Migration of applications

PostgreSQL App

```
// app.c
#include <libpq-fe.h>

void main()
{
    PGconn c = PQconnectdb(...);
    PGresult r = PQexec(c, ...);
    ...
    PQfinish(c);
}
```

PargreSQL App

```
// par_app.c
#include <par_libpq-fe.h>

void main()
{
    PGconn c = PQconnectdb(...);
    PGresult r = PQexec(c, ...);
    ...
    PQfinish(c);
}
```

+

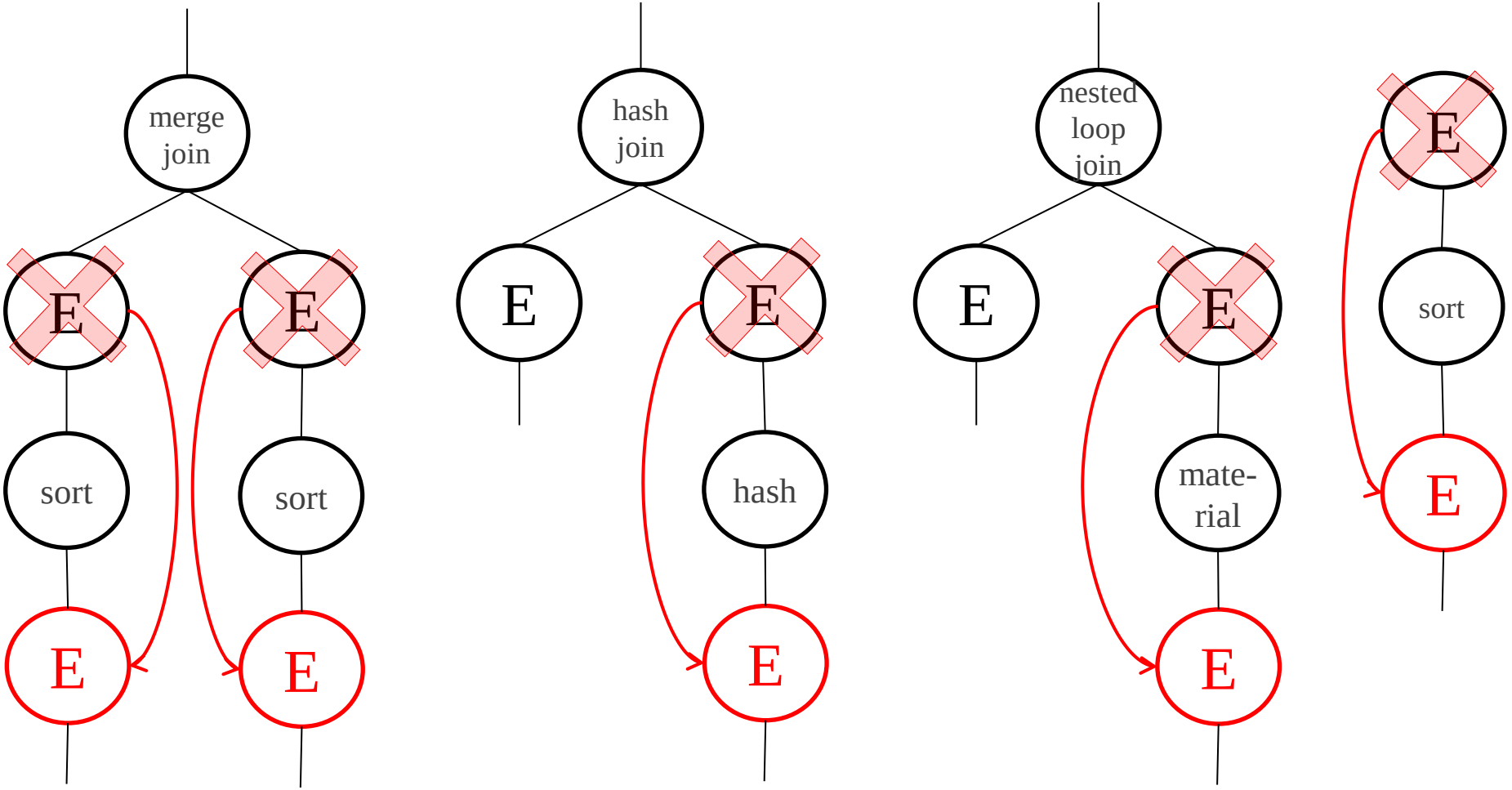
```
// par_Compat.h

#define PQconnectdb(...) \
    par_PQconnectdb(...)
#define PGconn \
    par_PGconn
...
```

PargreSQL: partitioning

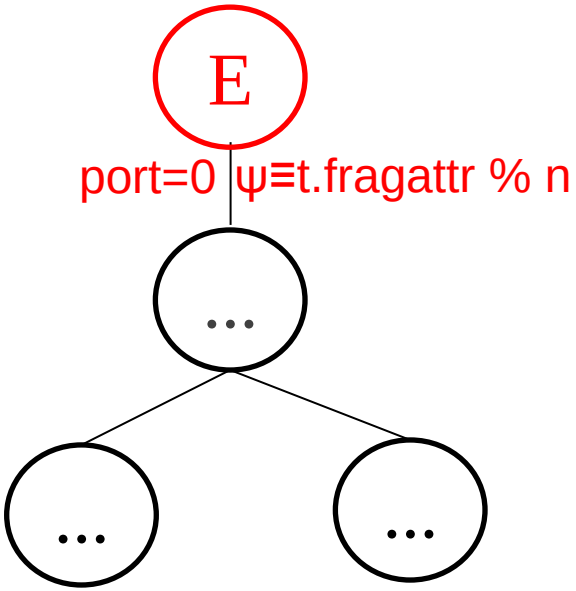
- create table S (
  SID integer primary key,
  Name char(50))
  with (fragattr = SID);
  -- Set SID as fragmentation attribute
  -- with fragmentation function SID % N,
  -- where number of nodes N
  -- is a number of lines in par_libpq.conf file.

PargreSQL: Parallelizer

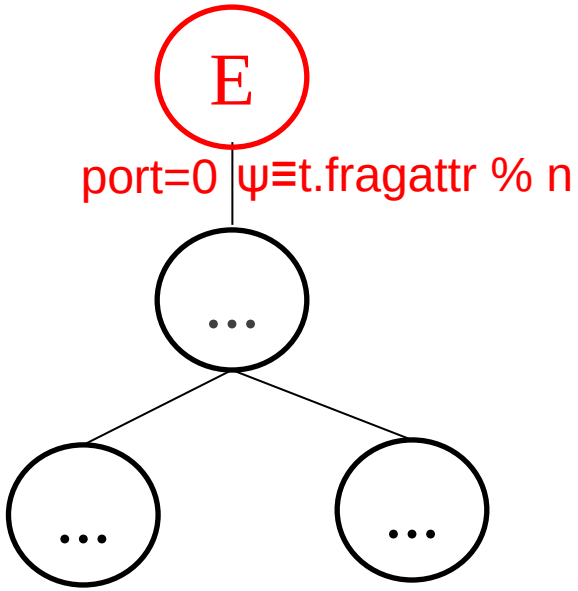


PargreSQL: Parallelizer (INSERT)

- insert into T values (...);
- insert into T select ...;



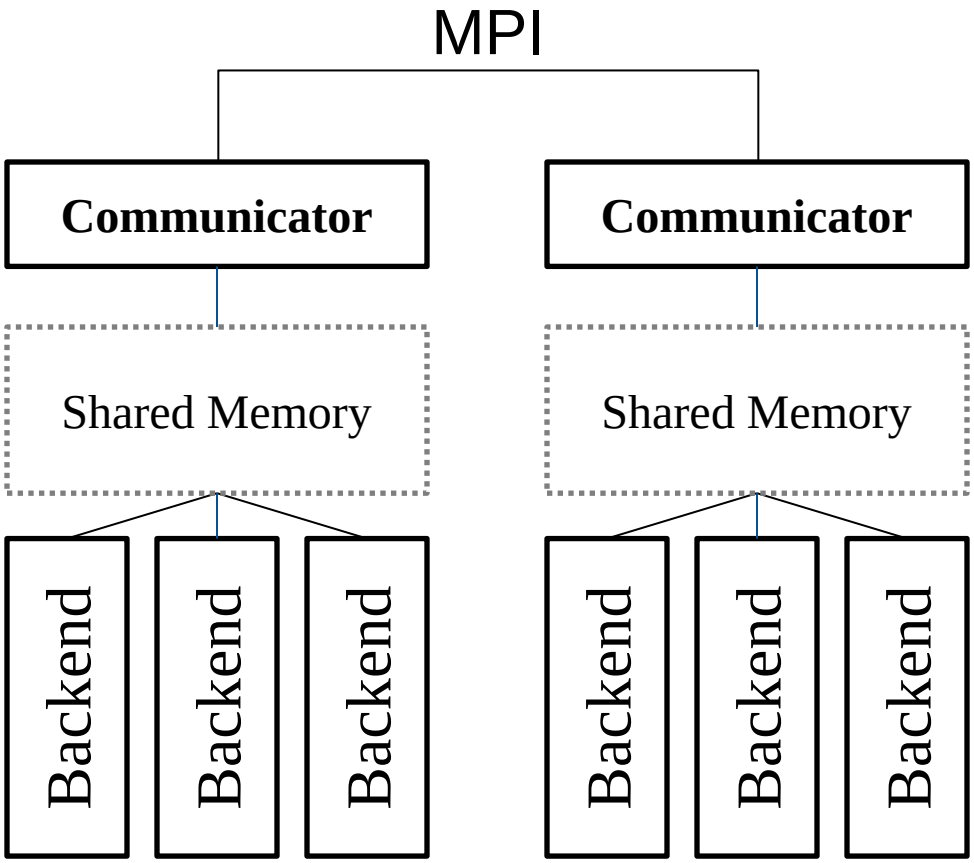
PargreSQL: Parallelizer (UPDATE)



Exchange behaves differently:

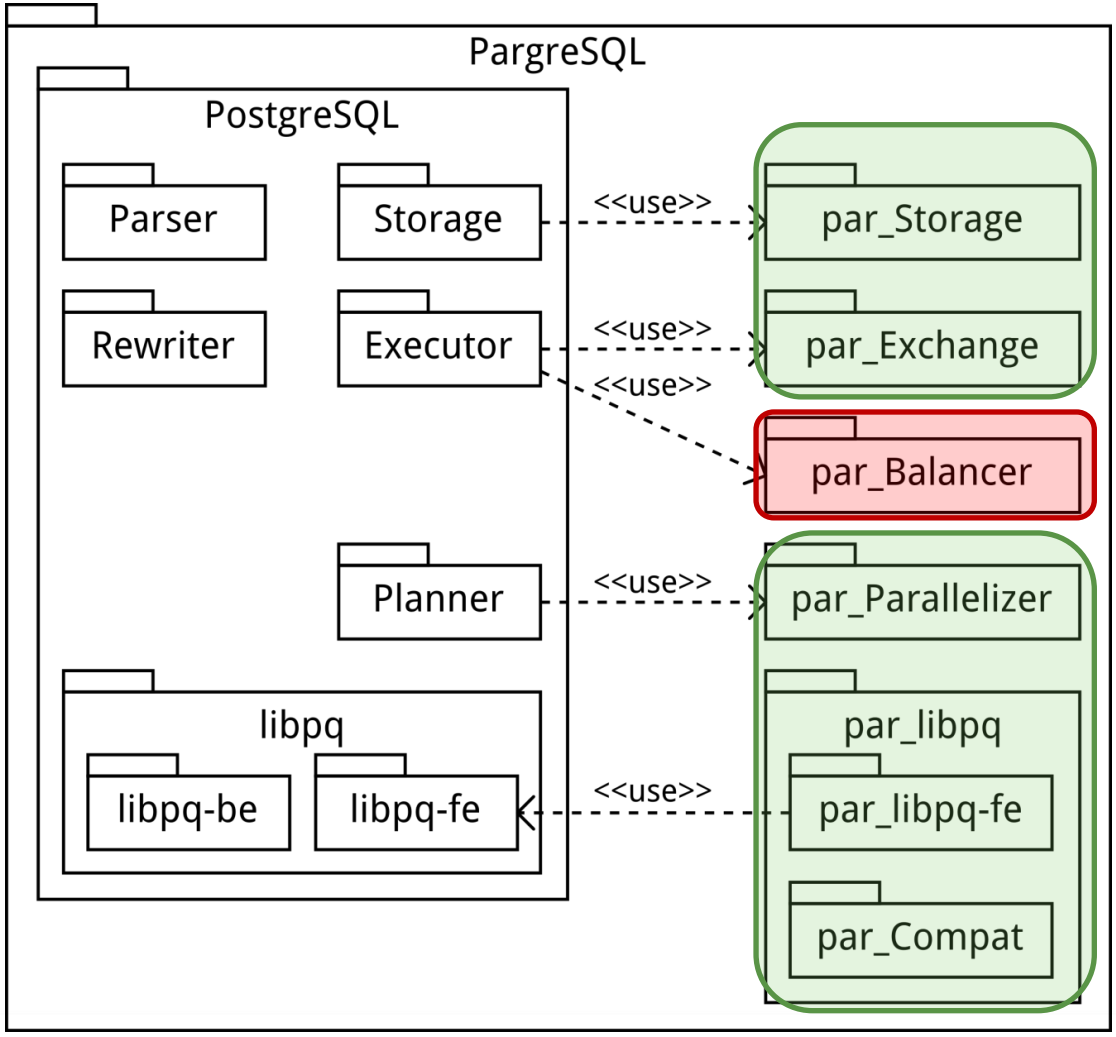
```
if (IsFoe(t)) {
   dup=Duplicate(t);
   t.SystemFlag=DO_INSERT;
   dup.SystemFlag=DO_DELETE;
   Send(dup, ψ(t));
   return (dup);
} else {
   do as usual
}
```

PargreSQL: Message Passing



- Why not plain MPI?
  - Because of a fork() inside the PostgreSQL daemon
  - Use shared memory for exchanges within one node, otherwise MPI
- MPI-like interface
  - Init()
  - Finalize()
  - GetRank()
  - GetSize()
  - IRecv()
  - ISend()
  - Test()
  - Finalize()

Current results



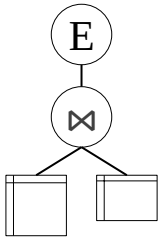
- Implemented
- To Do

- Source code size: 5K lines

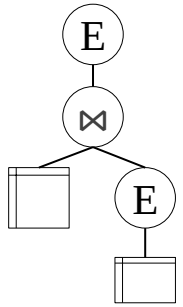
Experiments

- Hardware
  - SKIF-Aurora SUSU
  - Nodes: 1 to 10
- Database (synth. data)
  - T1 (f0, f1), frag attr is f1, 10^9 tuples
  - T2 (f0, f1), frag attr is f1, 10^5 tuples

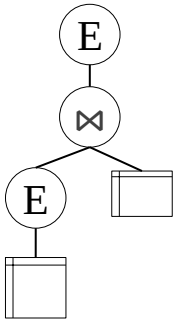
- Queries
  - select *
    from T1, T2
    where T1.f1=T2.f1



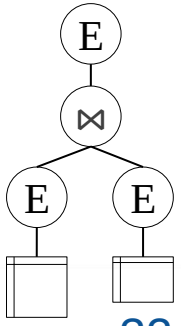
  - select *
    from T1, T2
    where T1.f1=T2.f0

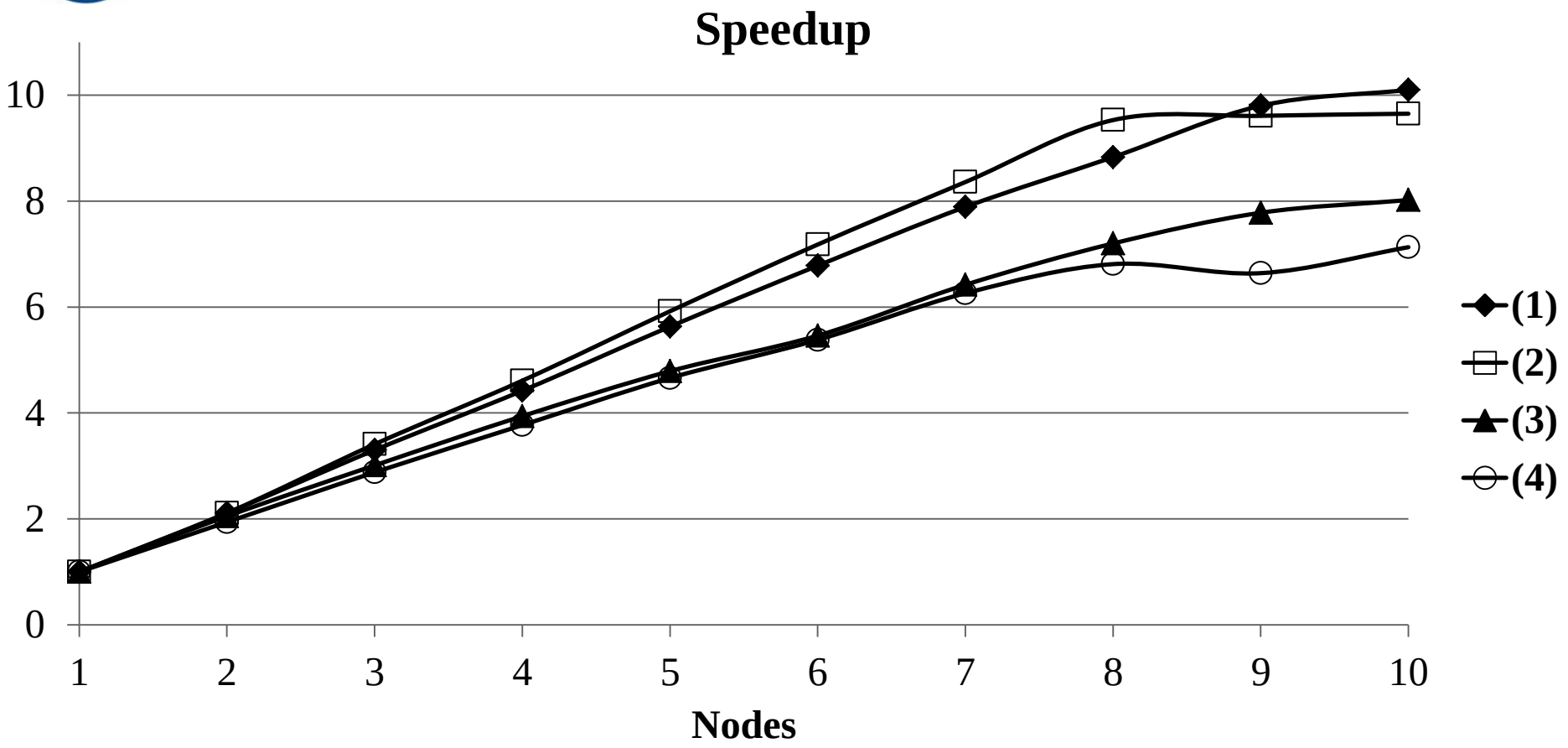


  - select *
    from T1, T2
    where T1.f0=T2.f1

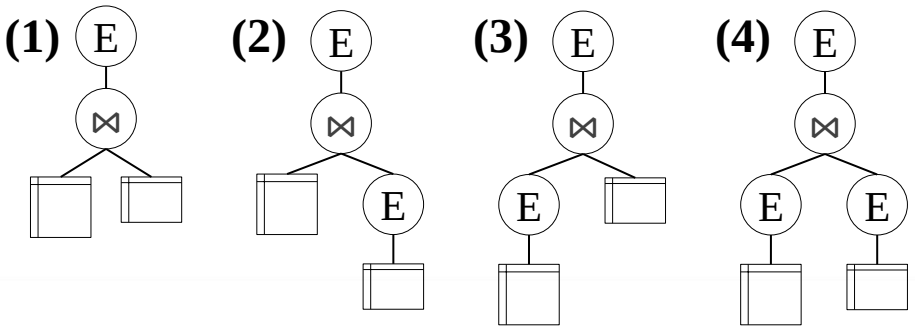


  - select *
    from T1, T2
    where T1.f0=T2.f0



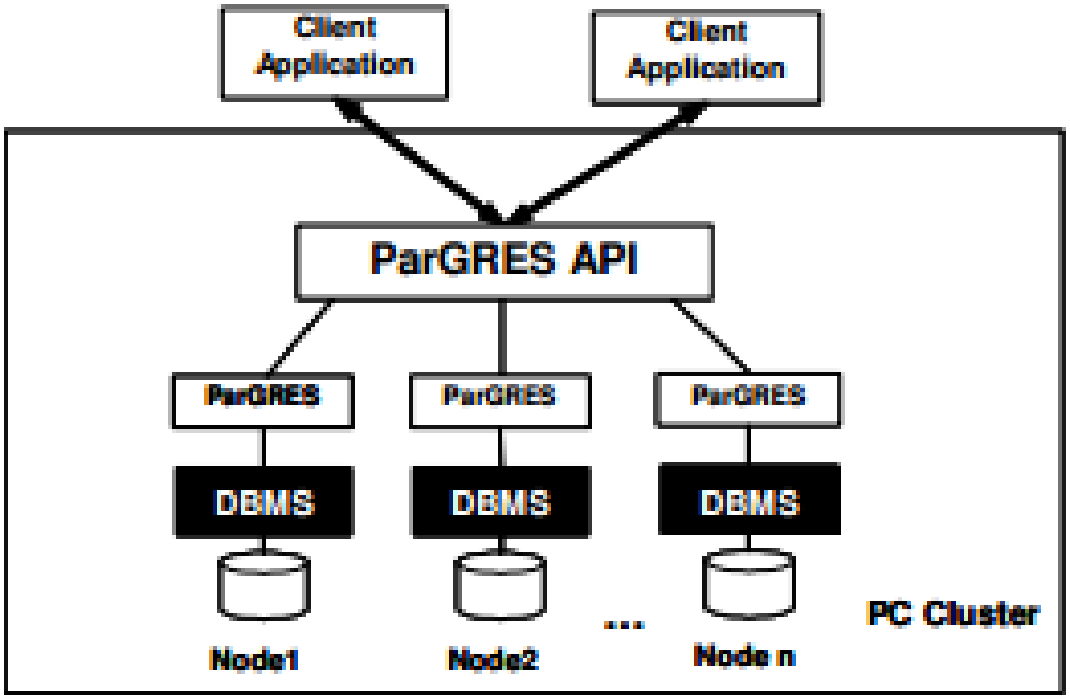


Experiments



Related work : ParGRES

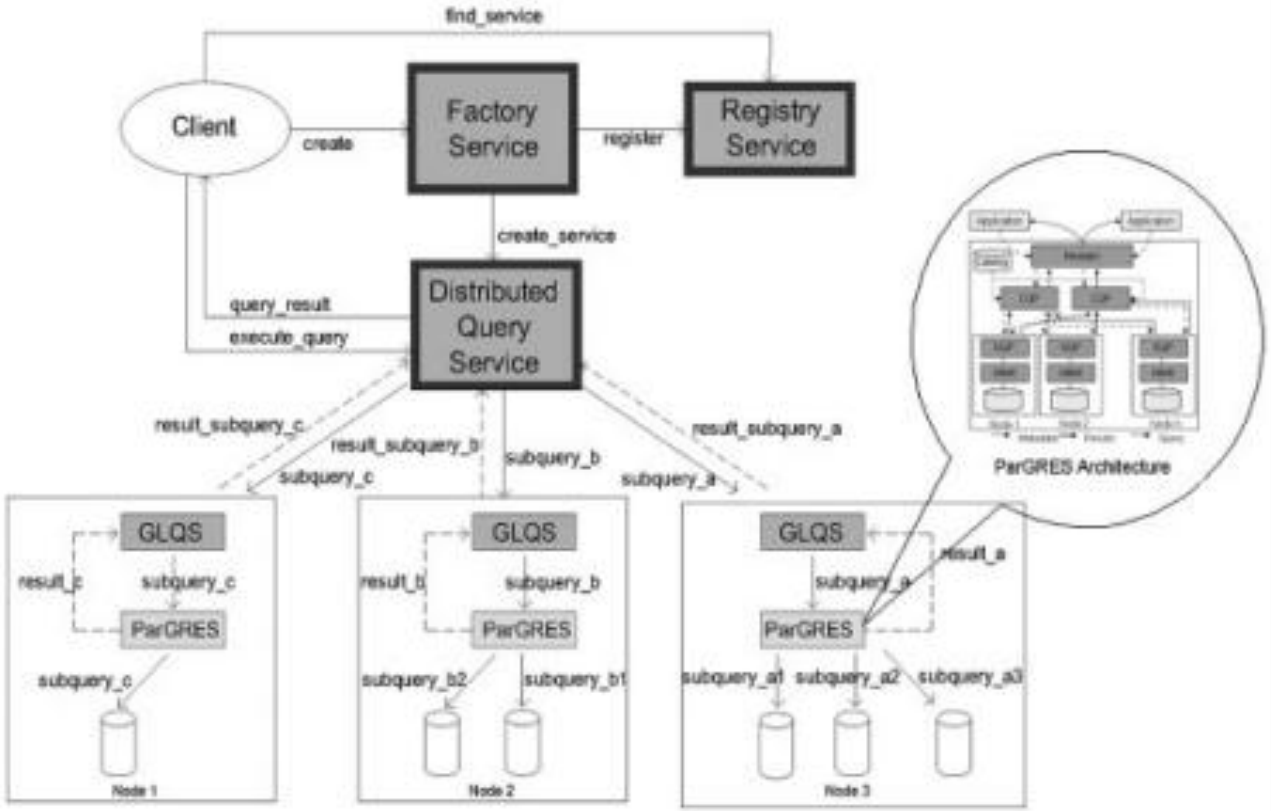
- *ParGRES* is a middleware over cluster of PostgreSQL DBMSes to process OLAP queries.



- *Paes M., Lima A.A., Valduriez P., Mattoso M.* High-Performance Query Processing of a Real-World OLAP Database with ParGRES // High Performance Computing for Computational Science - VECPAR 2008: 8th International Conference, Toulouse, France, June 24-27, 2008. LNCS. Vol. 5336. P. 188-200.

Related work: GParGRES

- *GParGRES* is an extension of ParGRES for grids.



- *Kotowski N. , Pacitti E., Valduriez P., Mattoso M.* Parallel query processing for OLAP in grids // Concurrency and Computation: Practice and Experience. 2008. Vol. 20. No. 17. P. 2039-2048.

Team

- Leonid Sokolinsky
  - Prof., Dr. of Science (Phys&Math)
  - Dean of Computational Mathematics and Informatics, Head of SSL

- Mikhail Zymbler
  - Assoc. Prof., Cand. of Science (Phys&Math)
  - Head of DM Dept, SSL

- Constantin Pan
  - Postgraduate student
  - Programmer of DM Dept, SSL

- Ruslan Miniakhmetov
  - Postgraduate student
  - Programmer of DM Dept, SSL

Team (continued)

- Aleksey Koltakov
  - Master student

- Elena Aksenova
  - Postgraduate student
  - Programmer of the S&T Dept, SSL

- Lyudmila Utkina
  - Master student
  - Programmer of the S&T Dept, SSL

- Alexander Medvedev
  - Master student

- Evgeny Gavrish
  - Master student

Thank you for paying attention!

- Questions?
  - Mikhail Zymbler
    zymbler@gmail.com
  - Constantin Pan
    kvapen@gmail.com
- More info
  - http://supercomputer.susu.ac.ru/en/
  - http://omega.susu.ru