South Ural State University: National Research University

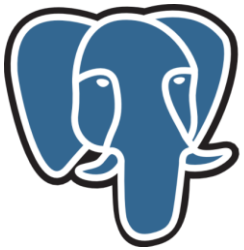
# Taming Elephants, or How to Embed Parallelism into PostgreSQL

**Mikhail Zymbler, Constantin Pan**
South Ural State University,
Chelyabinsk, Russian Federation

**24th International Conference on Database and Expert Systems Applications - DEXA 2013**
Prague, Czech Republic, August 26-29, 2013

The reported study was partially supported by the Russian Foundation for Basic Research, research projects No. 12-07-31217 and No. 12-07-00443.

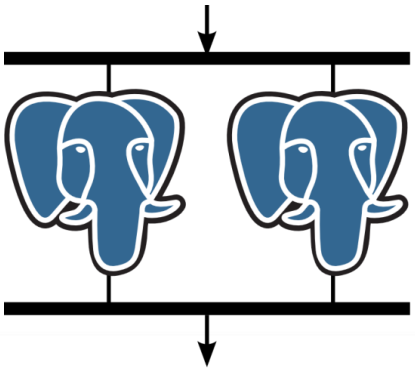
# PargreSQL project



PostgreSQL

+

**PARTITIONED PARALLELISM**

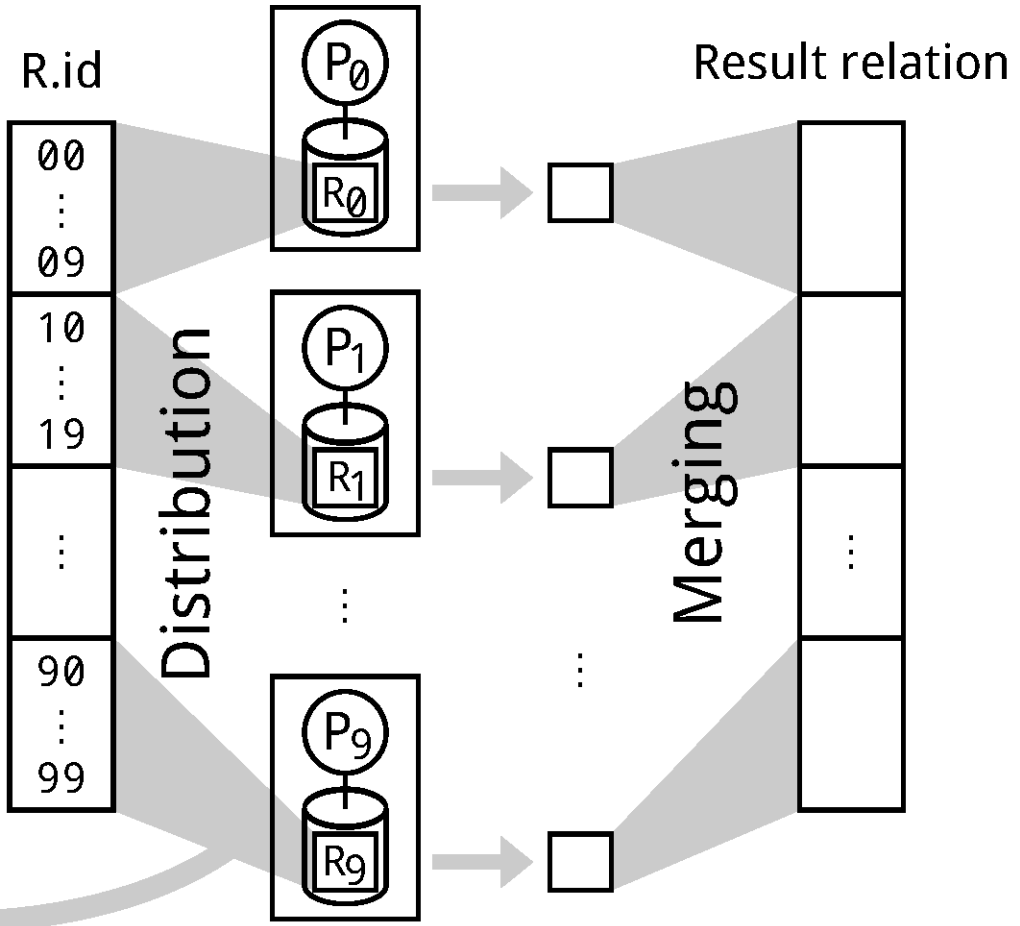


PargreSQL

# Partitioned parallelism

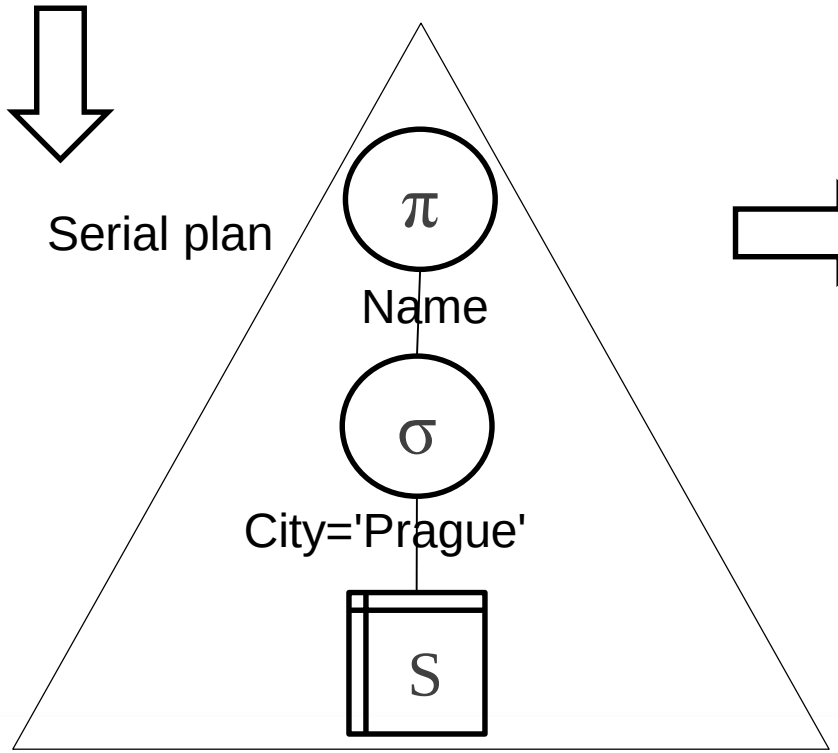
$$R_i = \{t \mid t \in R,\ \phi(t) = i\}$$
$$i = 0, \ldots, 9$$

Fragmentation function
$\phi(t) = (t.id \text{ div } 10) \bmod 10$

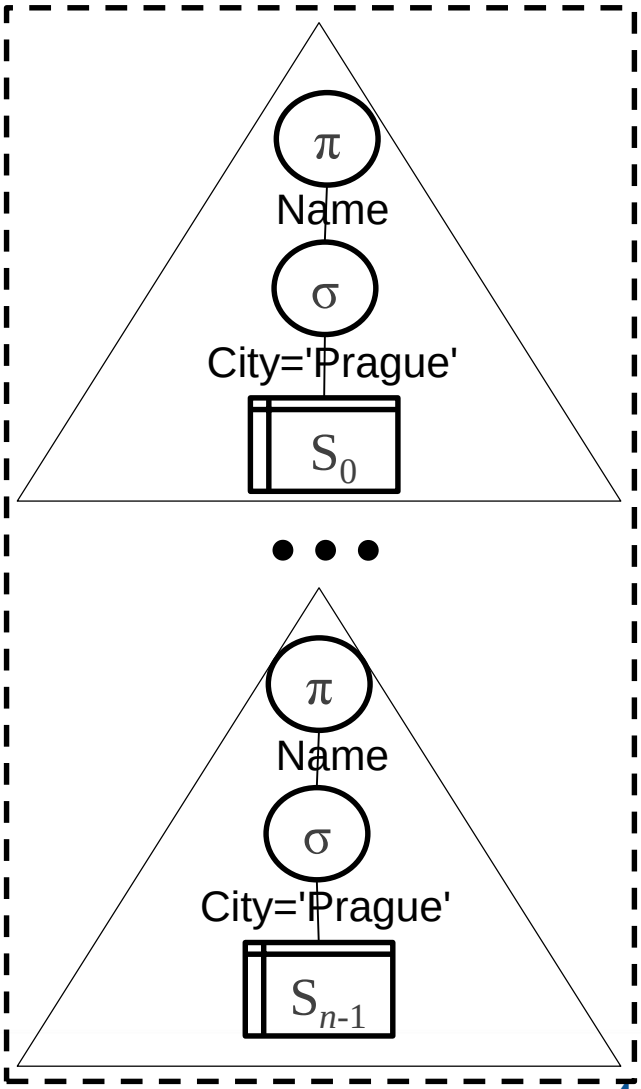


# Serial vs parallel query plan

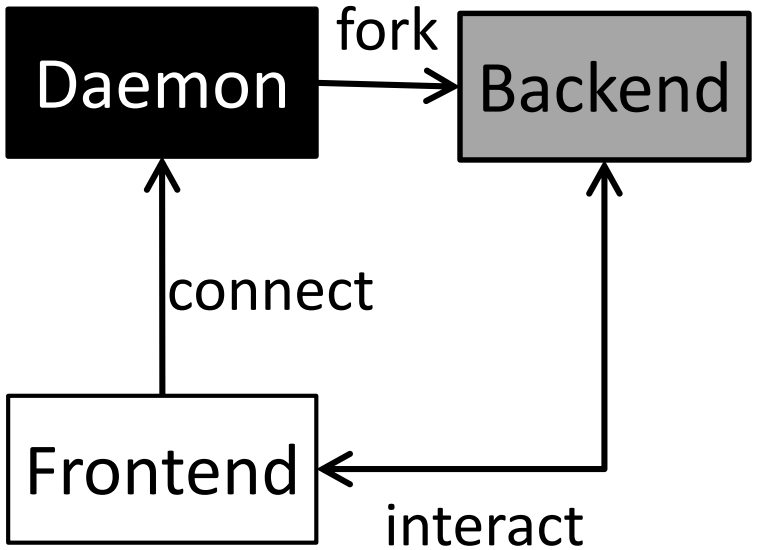
```
SELECT Name
FROM S
WHERE City='Prague'
```





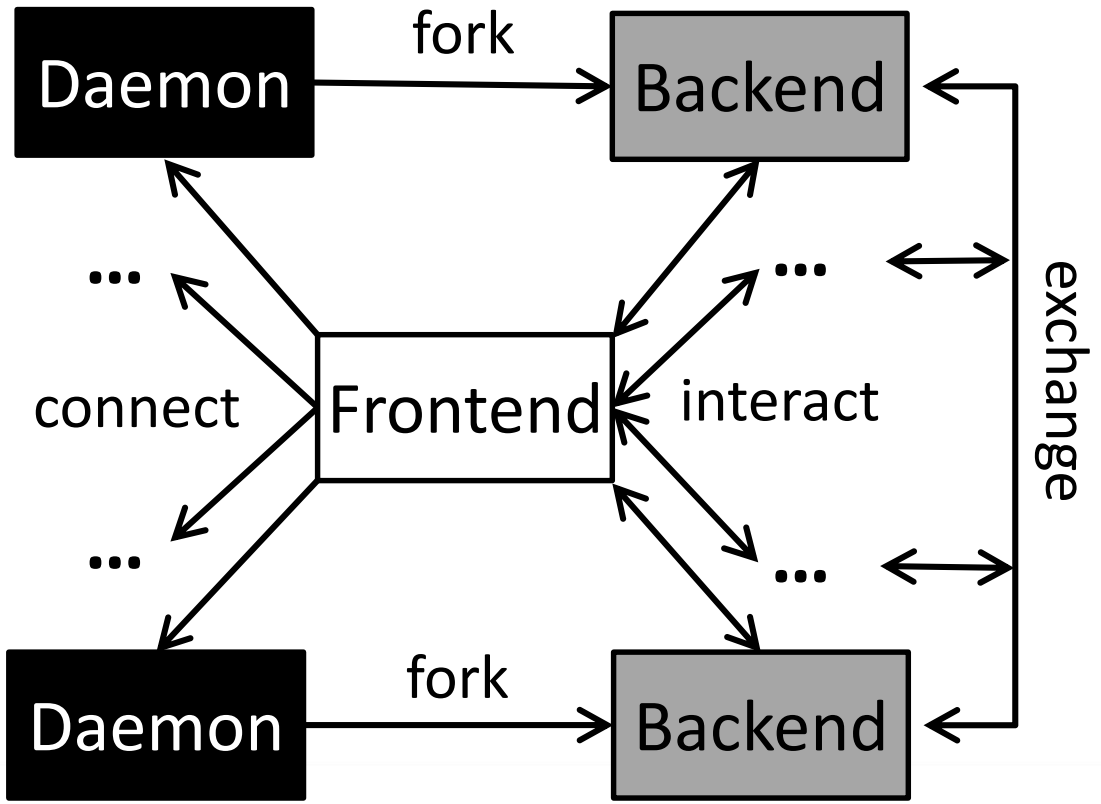



# DBMS processes



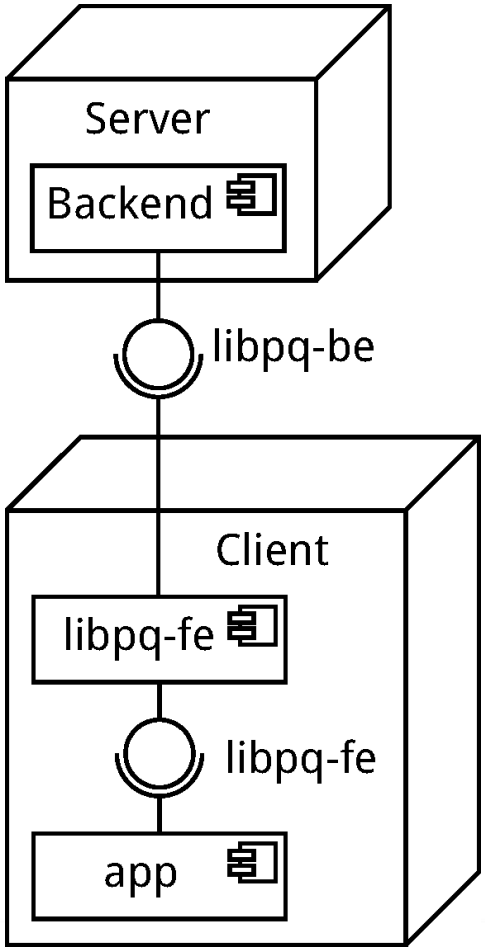

**PostgreSQL**

**PargreSQL**

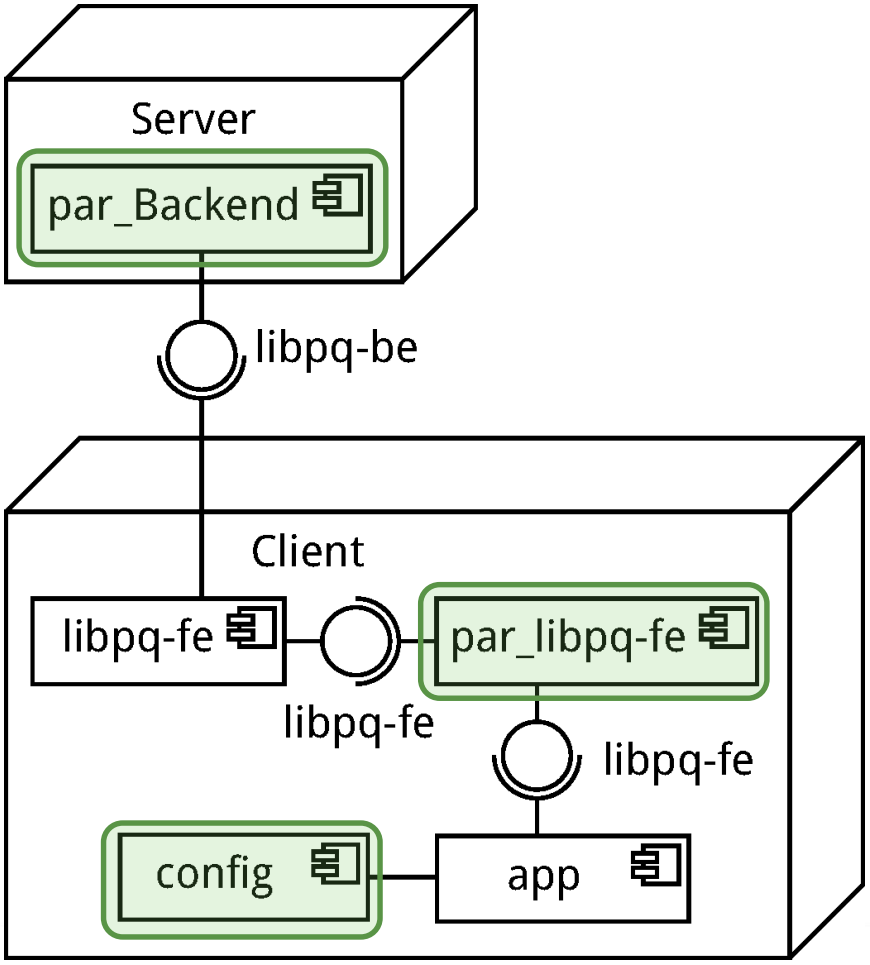


# Deployment scheme

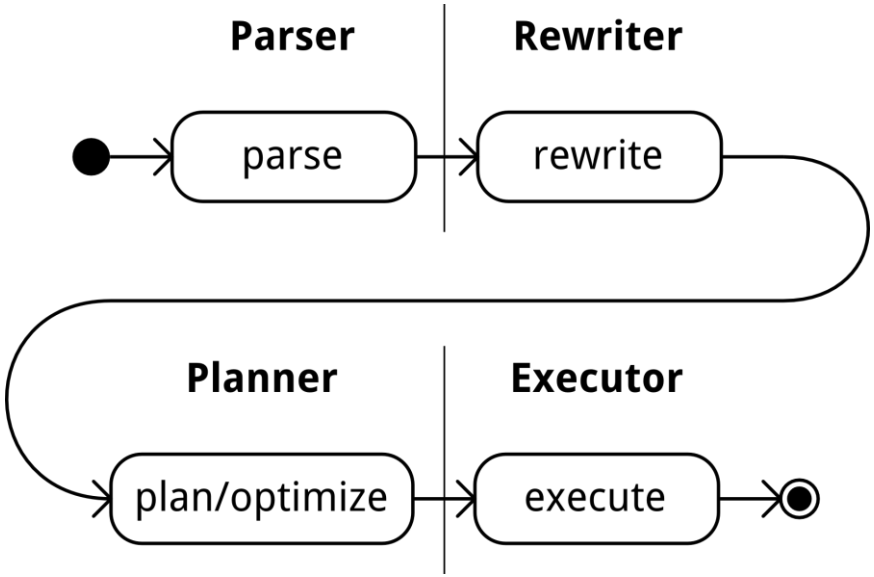
## PostgreSQL




## PargreSQL

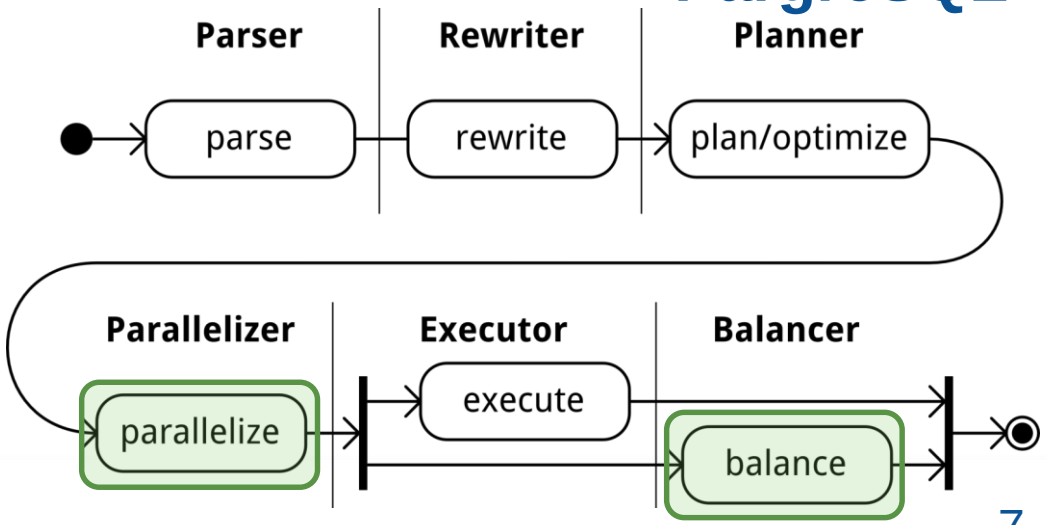


# Query processing



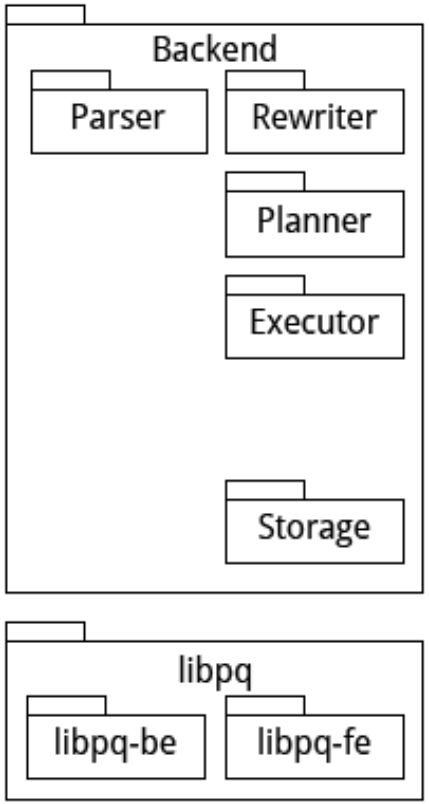

**PostgreSQL**

**PargreSQL**

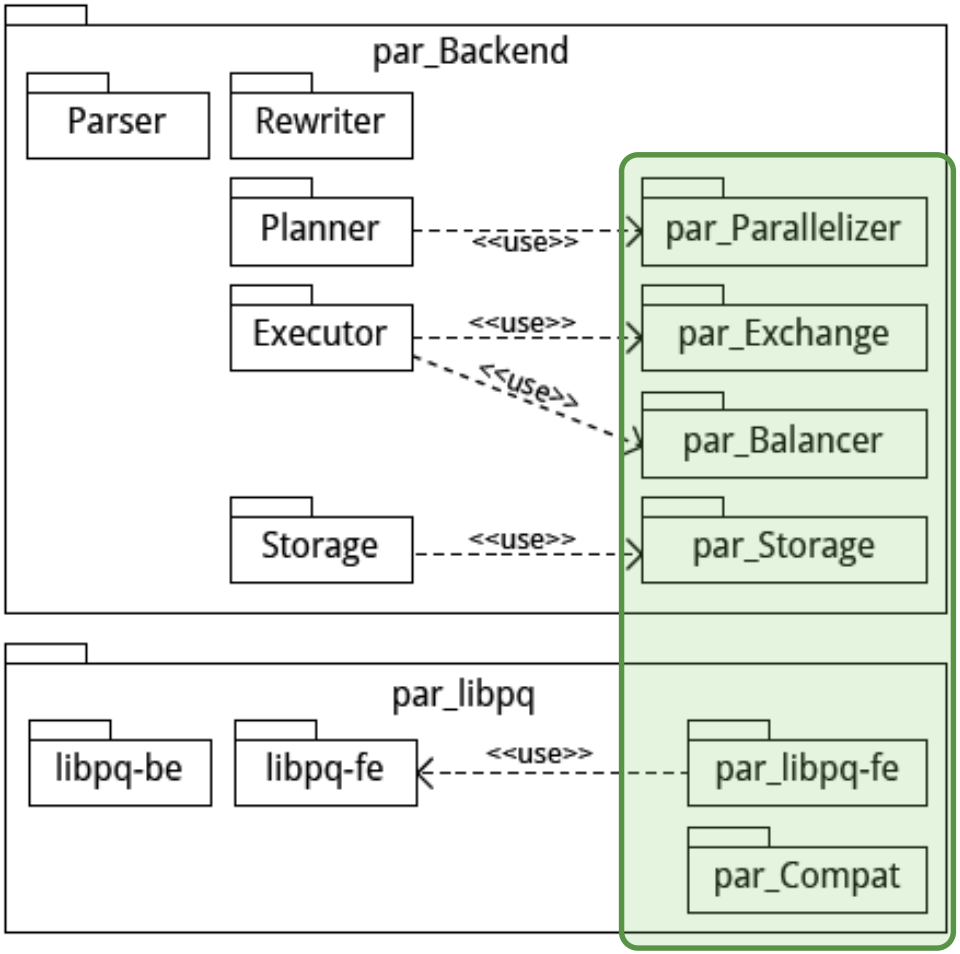


# DBMS subsystems

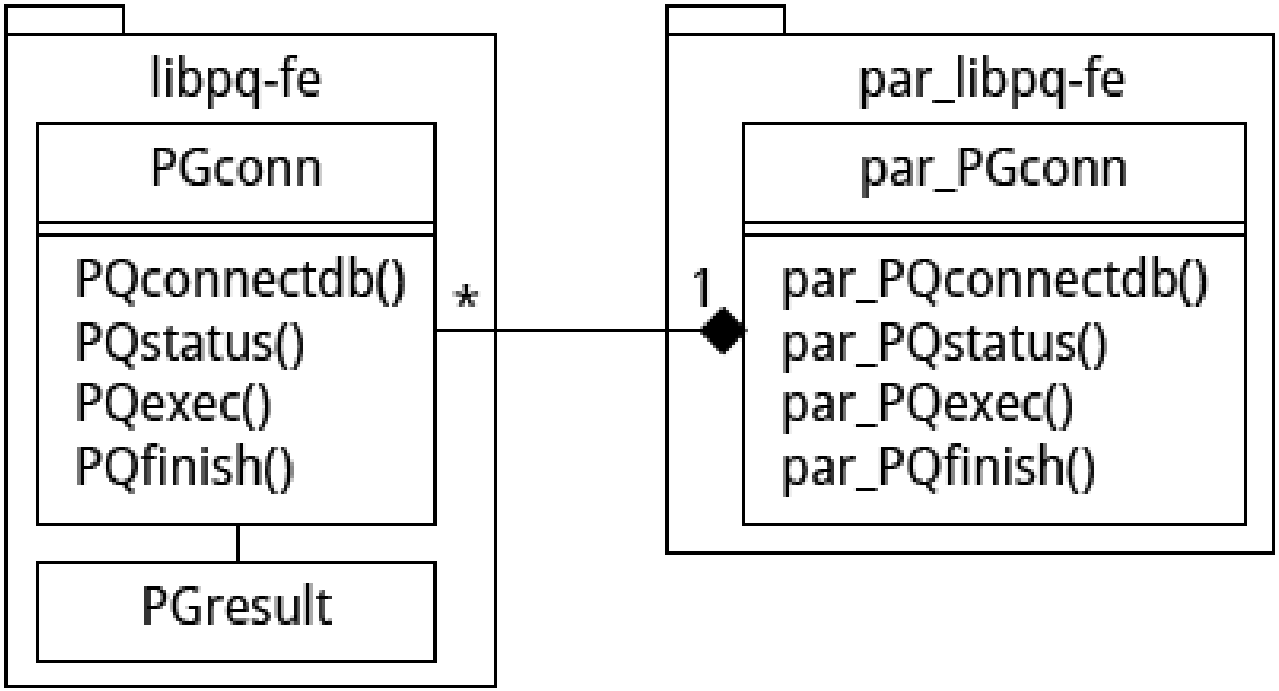
## PostgreSQL




## PargreSQL



# par_libpq-fe

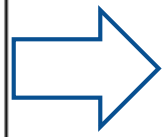

libpq-fe
PGconn
PQconnectdb()
PQstatus()
PQexec()
PQfinish()
PGresult
*
1
par_libpq-fe
par_PGconn
par_PQconnectdb()
par_PQstatus()
par_PQexec()
par_PQfinish()

# par_Compat

```
#define PGconn par_PGconn
#define PQconnectdb(X) par_PQconnectdb()
#define PQfinish(X) par_PQfinish(X)
#define PQstatus(X) par_PQstatus(X)
#define PQexec(X,Y) par_PQexec(X,Y)
```

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

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



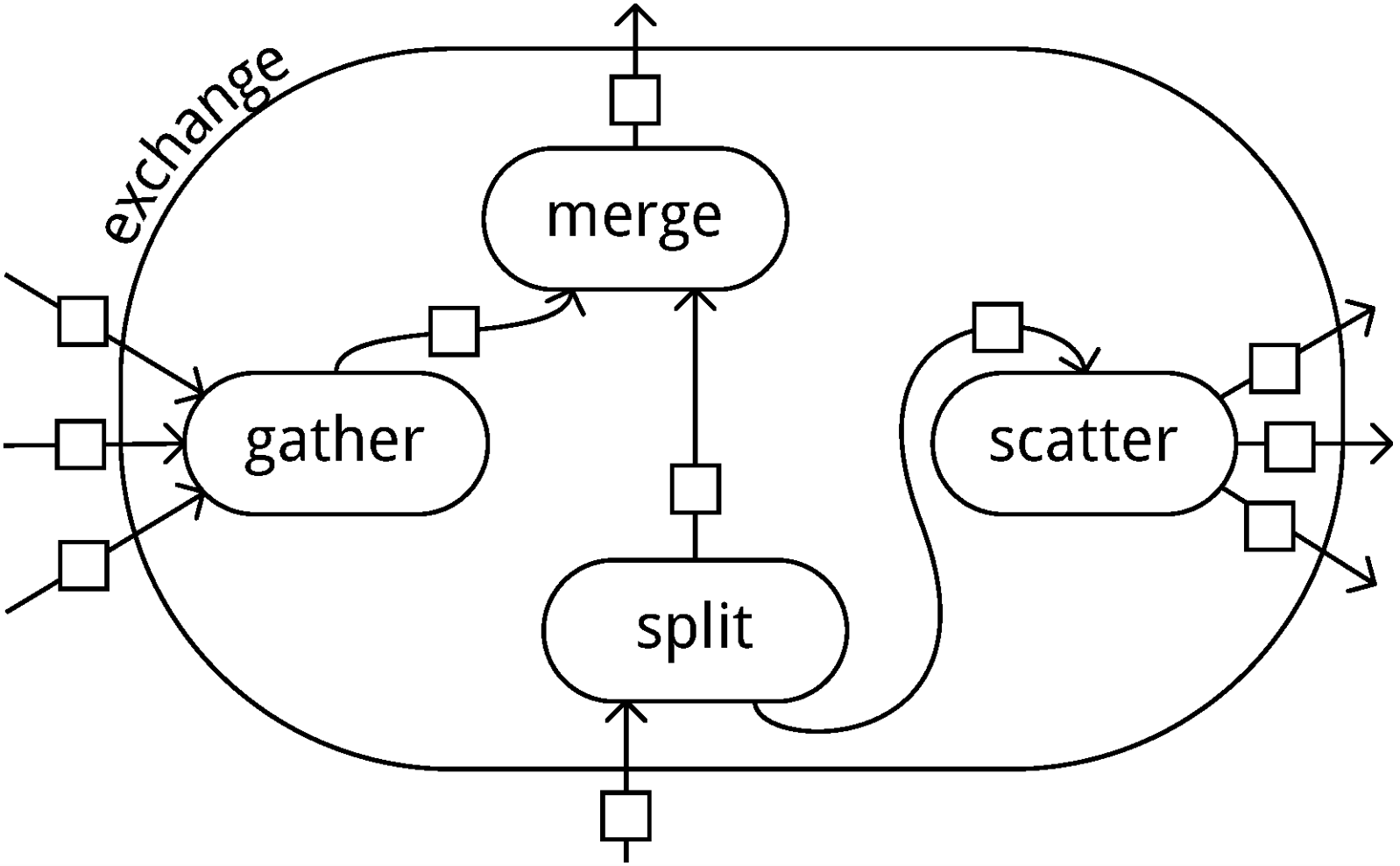
```
// par_app.c
#include <par_libpq-fe.h>

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

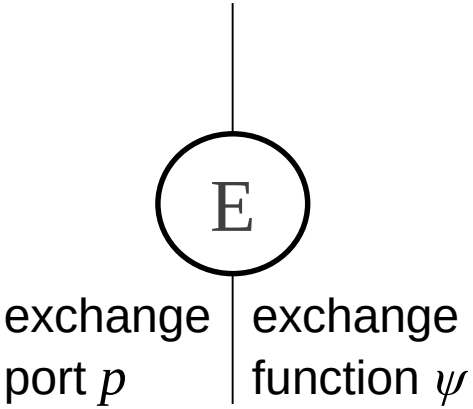
**PostgreSQL application**

**PargreSQL application**

# EXCHANGE operator



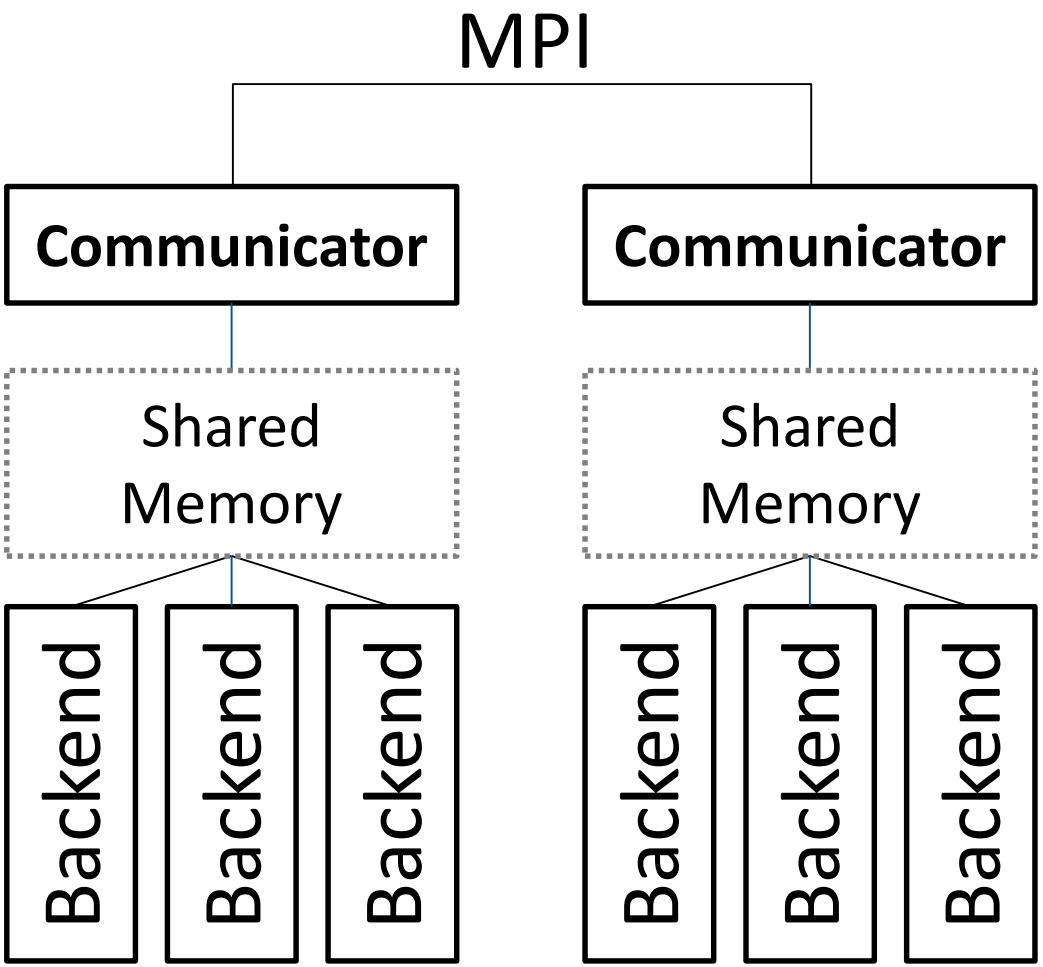
# EXCHANGE operator




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

```
if (ψ(tuple)==mynode())
   Put(tuple, this_output_buffer);
else {
   Send(tuple, ψ(tuple));
   Put(tuple, that_output_buffer);
}
```

# Message passing system




- Why not plain MPI?
  - Because of a fork() inside the PostgreSQL daemon
- MPI-like interface
  - ISend()
  - IRecv()
  - Test()

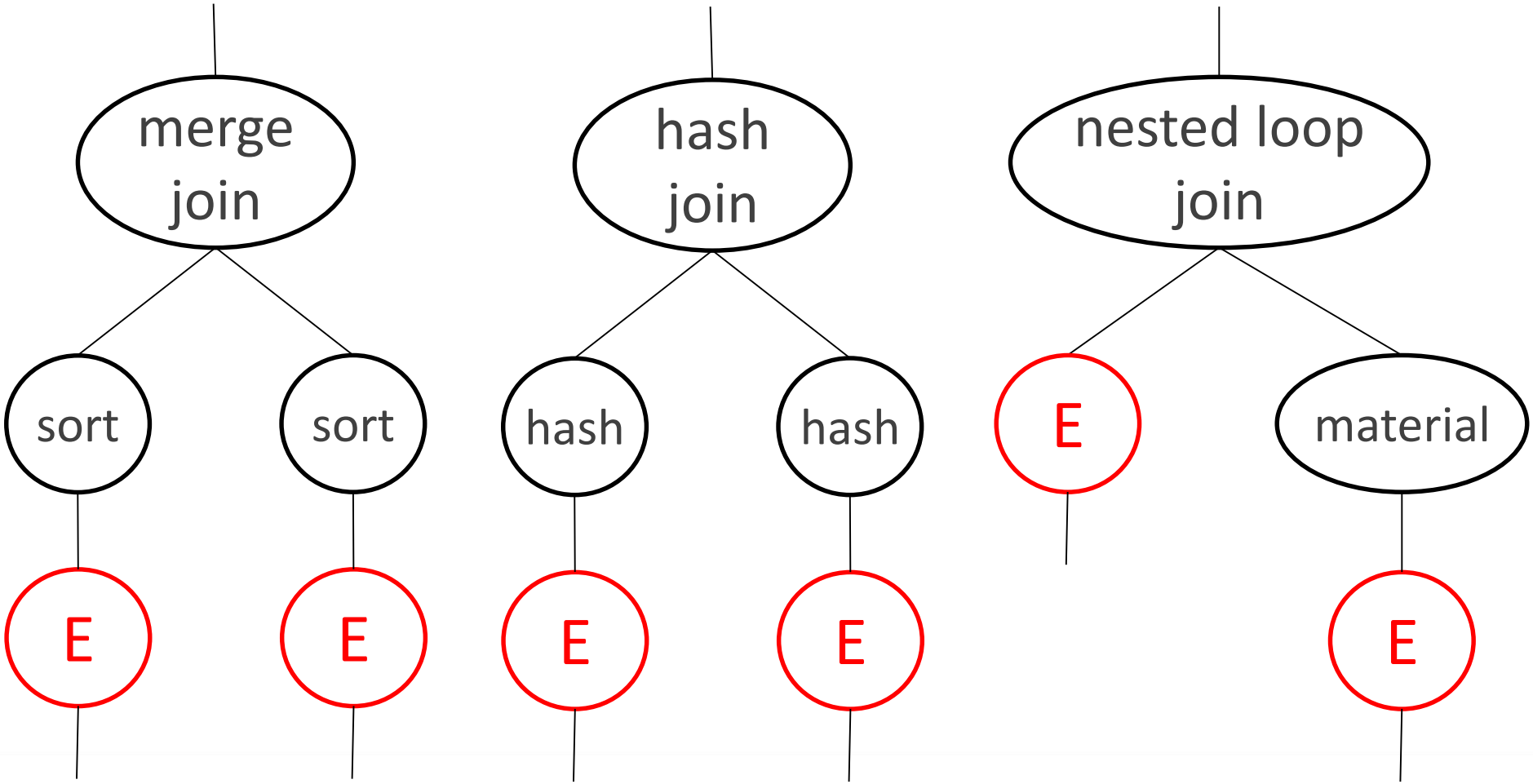
# par_Storage

- create table R (
  ID integer primary key,
  Attr1 …
  …)
  **with (fragattr = ID)**;

-- Set ID as fragmentation attribute
-- with fragmentation function ID % N,
-- where number of nodes N
-- is a number of lines in config file.

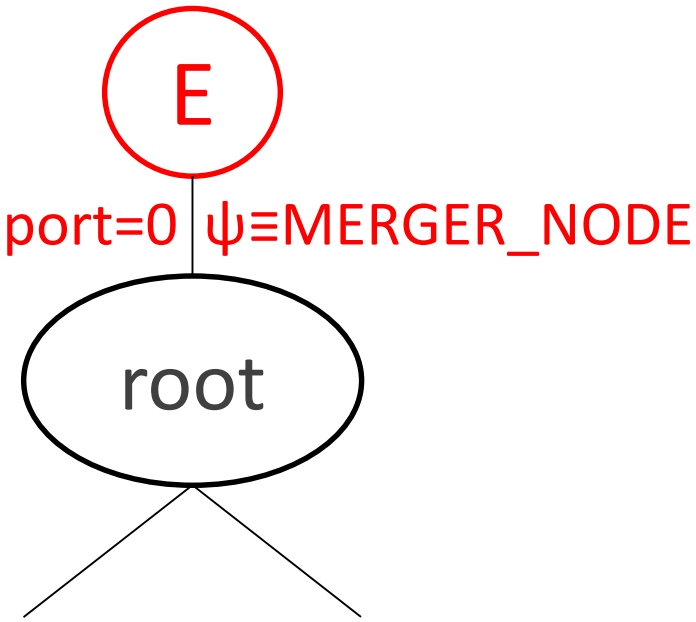
# Parallelizer: joins

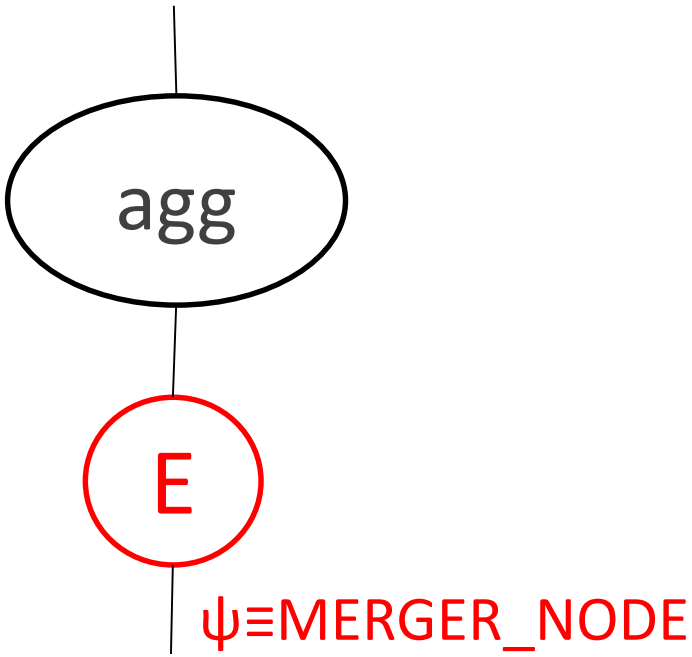
select smth from T1, T2



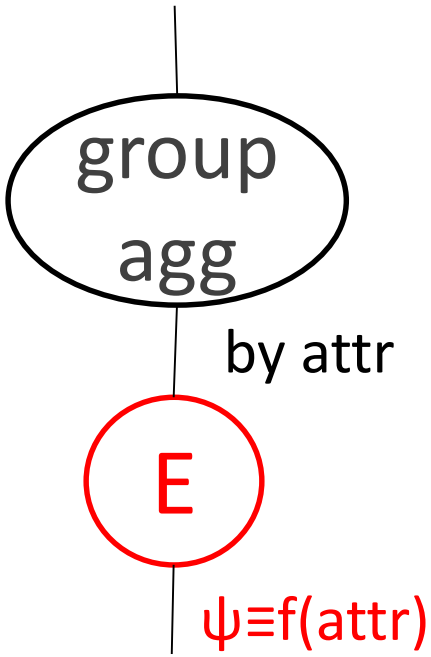
# Parallelizer: merging and aggregation

select sum(a) from T



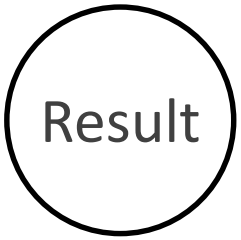




select a, sum(b) from T
group by a



# Parallelizer: INSERT queries

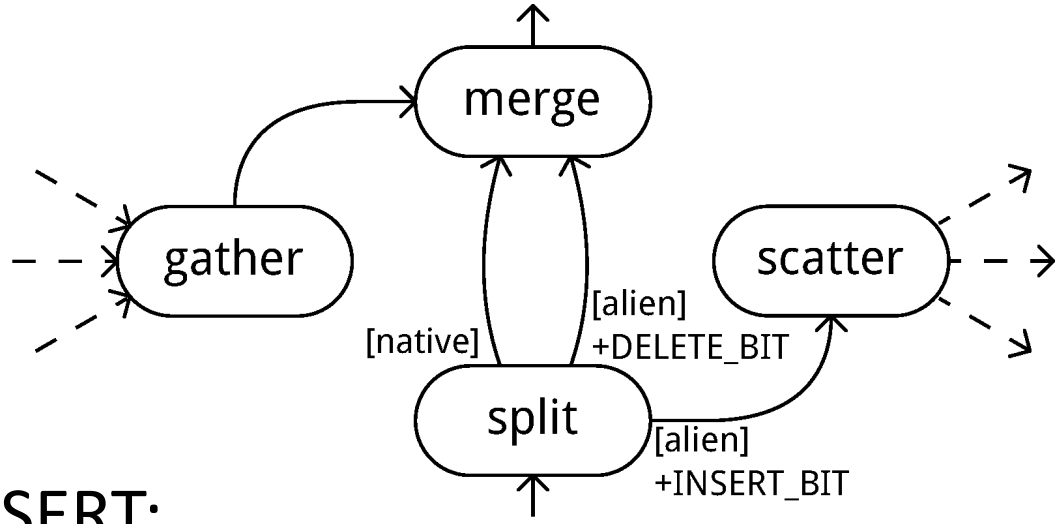
insert into T values (...);




filter(t.fragattr % n = mynode)

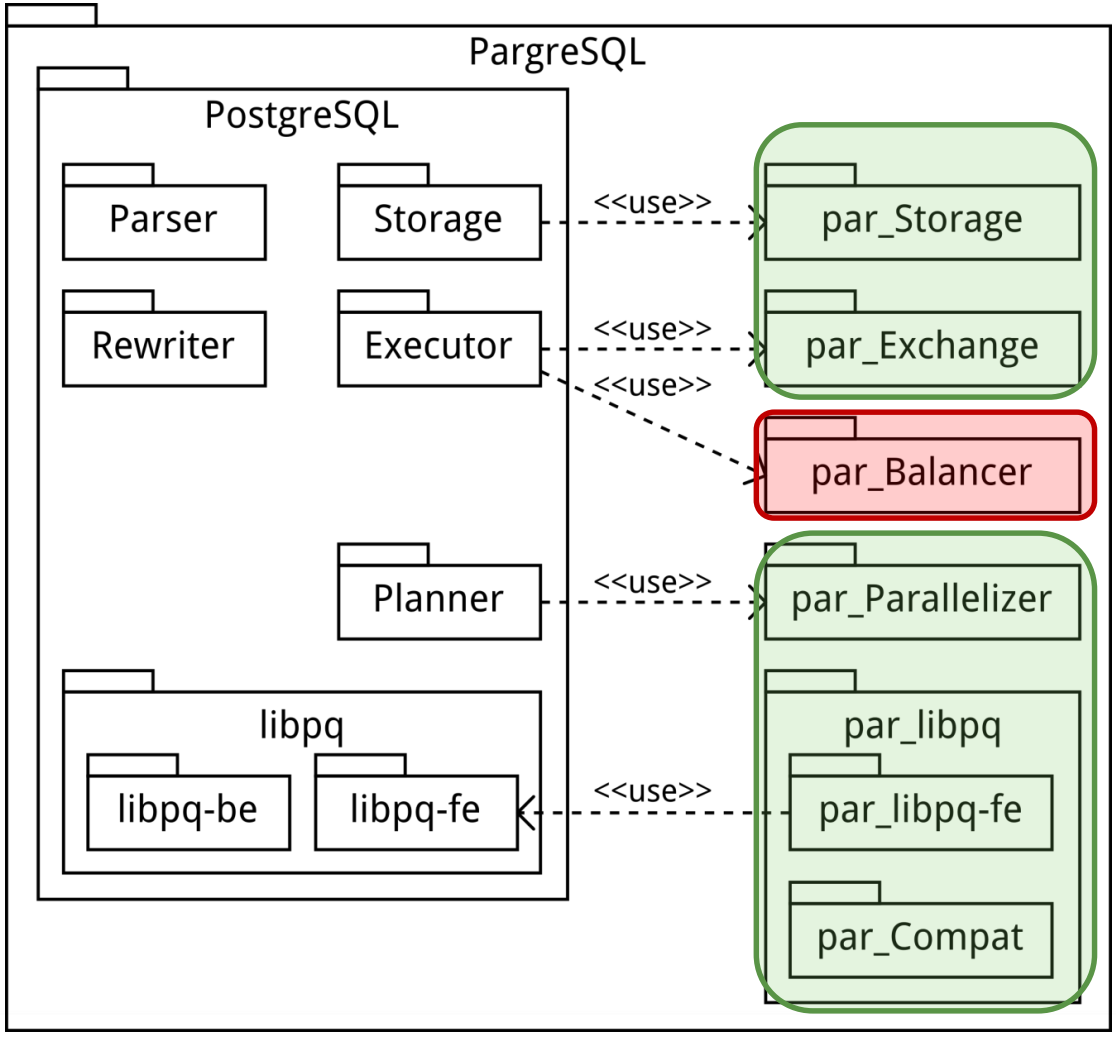
# Parallelizer: UPDATE queries

update T set …




```
if (!IsNative(t)) {
  // for the SCATTER
  dup=Duplicate(t);
  dup.SystemFlag=DO_INSERT;
  Send(t, ψ(t));
  // for the MERGE
  t.SystemFlag=DO_DELETE;
  return (t);
} else do as usual
```

# Current results




- Implemented
- To Do

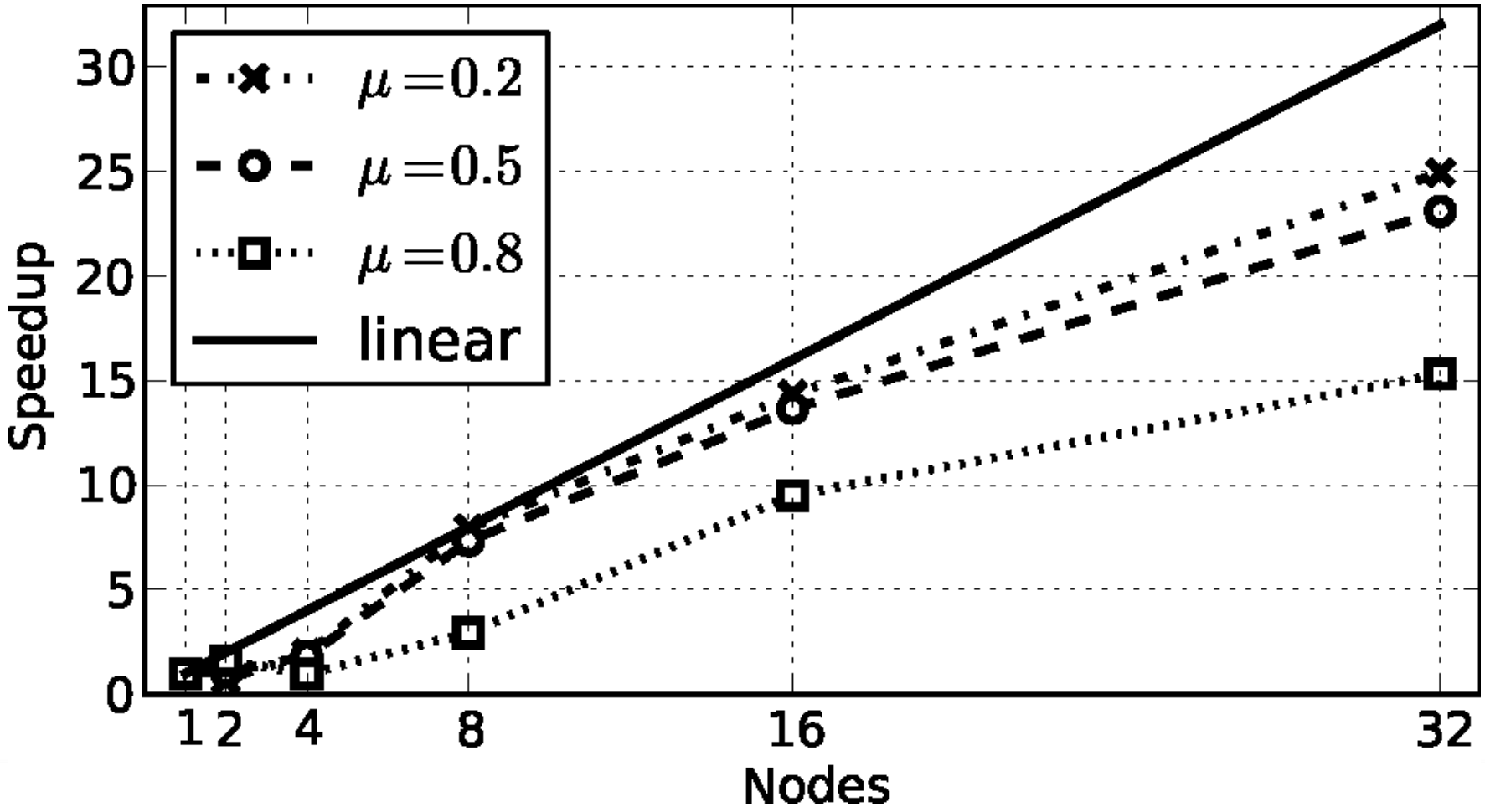
- Source code size: 5K lines

# Experiments

$\mathbf{R} \bowtie \mathbf{S}$

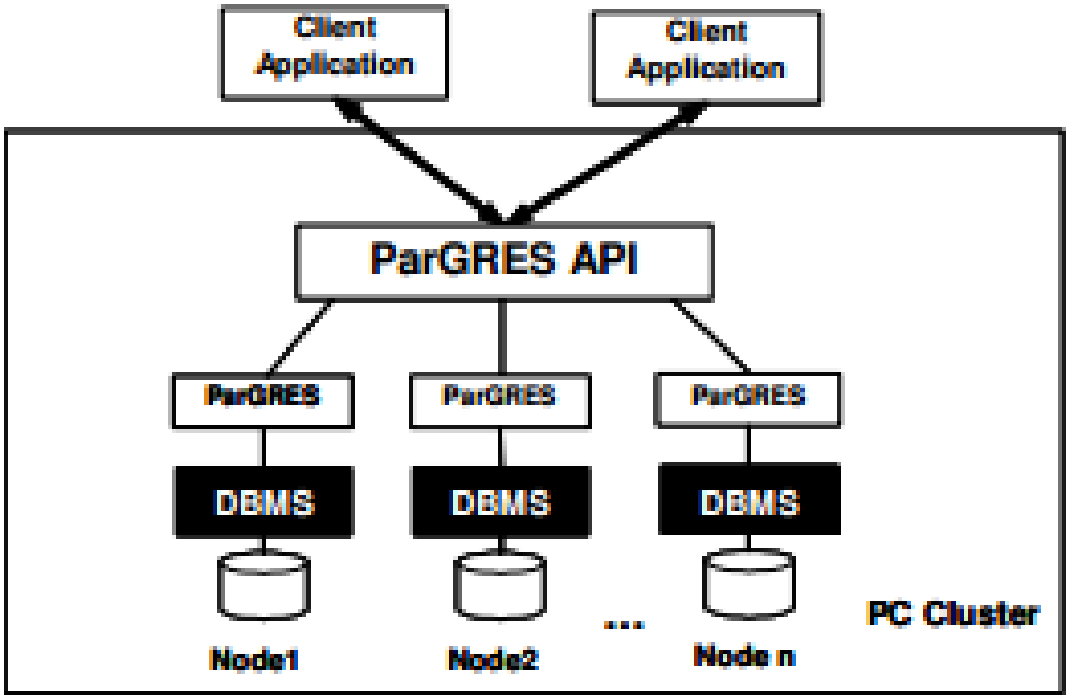
$|\mathbf{R}|=6 \cdot 10^7$

$|\mathbf{S}|=1.5 \cdot 10^6$



# Related work

- *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.

# Conclusion

- Design and implementation of PargreSQL parallel DBMS for cluster systems has been presented.
- PargreSQL is based upon PostgreSQL open-source DBMS and exploits partitioned parallelism.
- This approach is applicable to other open-source relational DBMSes (e.g. MySQL).

# Thank you for paying attention!

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