Внедрение фрагментного параллелизма в СУБД с открытым исходным кодом

К.С. Пан, М.Л. Цымблер

Актуальность

- Сверхбольшие данные
- Сверхбольшие реляционные базы данных
- Параллельные СУБД, фрагментный параллелизм
- Коммерческие параллельные СУБД — высокая стоимость или специфические аппаратно-программные платформы
- Свободные СУБД — не реализуют фрагментный параллелизм

Цель

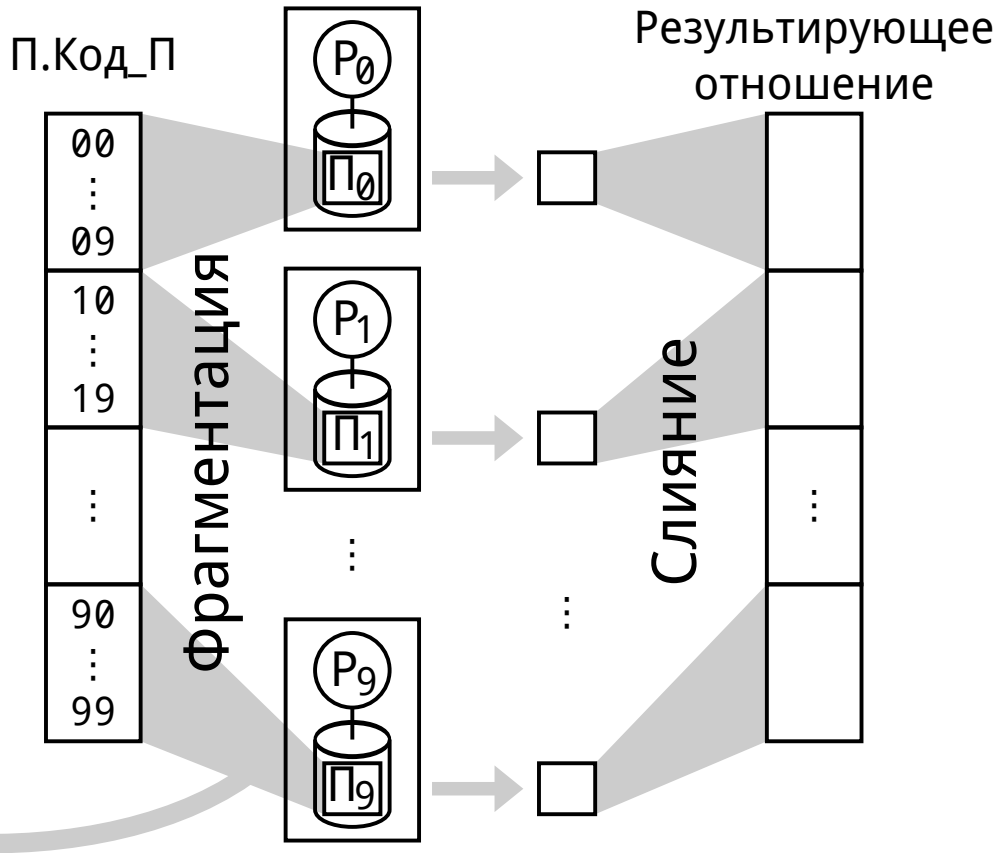
Цель работы: разработка методов, архитектурных подходов и алгоритмов, обеспечивающих внедрение фрагментного параллелизма в имеющиеся последовательные СУБД, свободно распространяемые на уровне исходных кодов, а также проверка разработанных методов путем их применения для решения задач аналитической и оперативной обработки сверхбольших баз данных.

Основные задачи

1. Разработать методы внедрения фрагментного параллелизма в последовательную СУБД с открытым исходным кодом.
2. На основе разработанных методов предложить архитектурные подходы и алгоритмы, реализующие фрагментный параллелизм в рамках последовательной СУБД с открытым исходным кодом. Выполнить параллелизацию одной из свободных последовательных СУБД.
3. Исследовать эффективность предложенных подходов.

Фрагментный параллелизм

$$\Pi_i = \{t | t \in \Pi, \phi(t) = i\}$$
$$i = 0, \ldots, 9$$



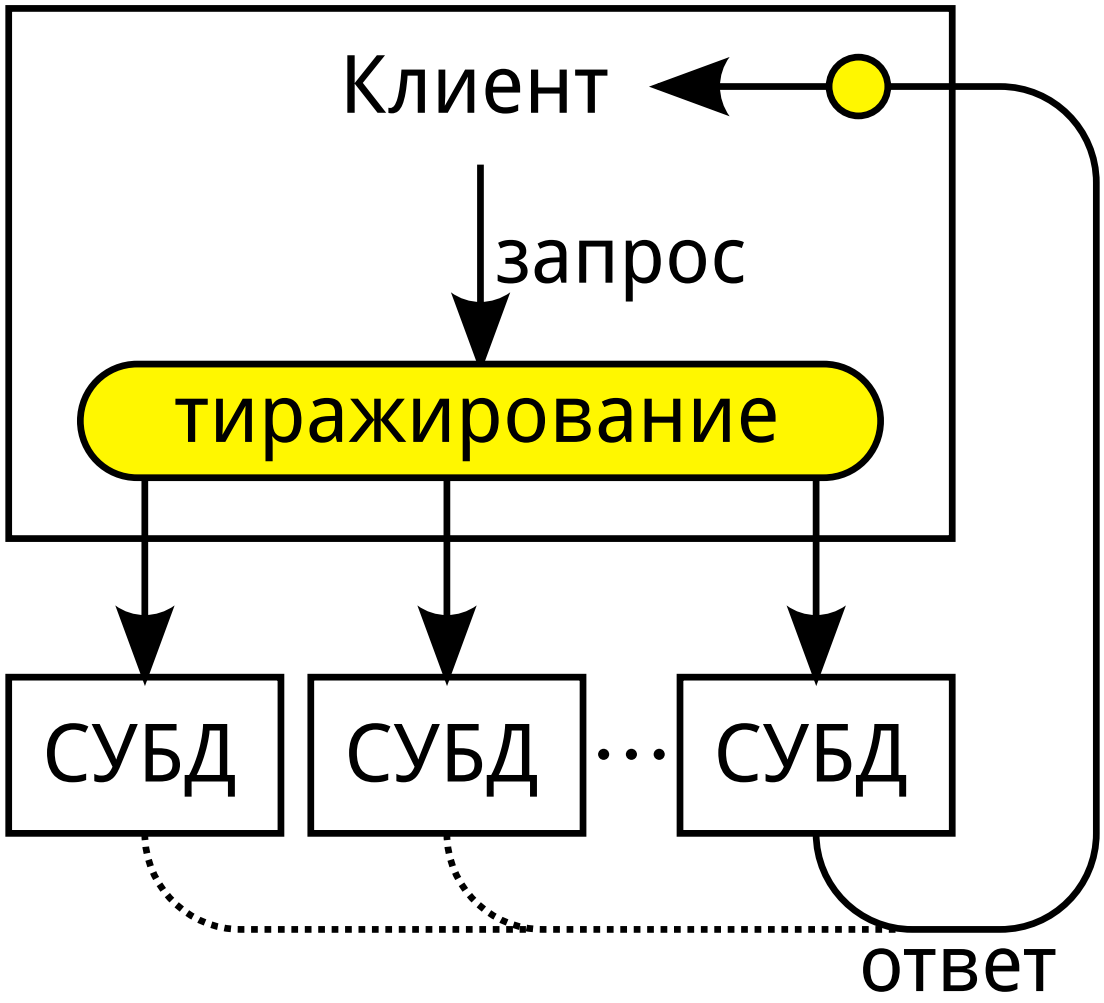
Функция фрагментации

$$\phi(t) = (t.\text{Код\_П} \text{ div } 10) \bmod 10$$

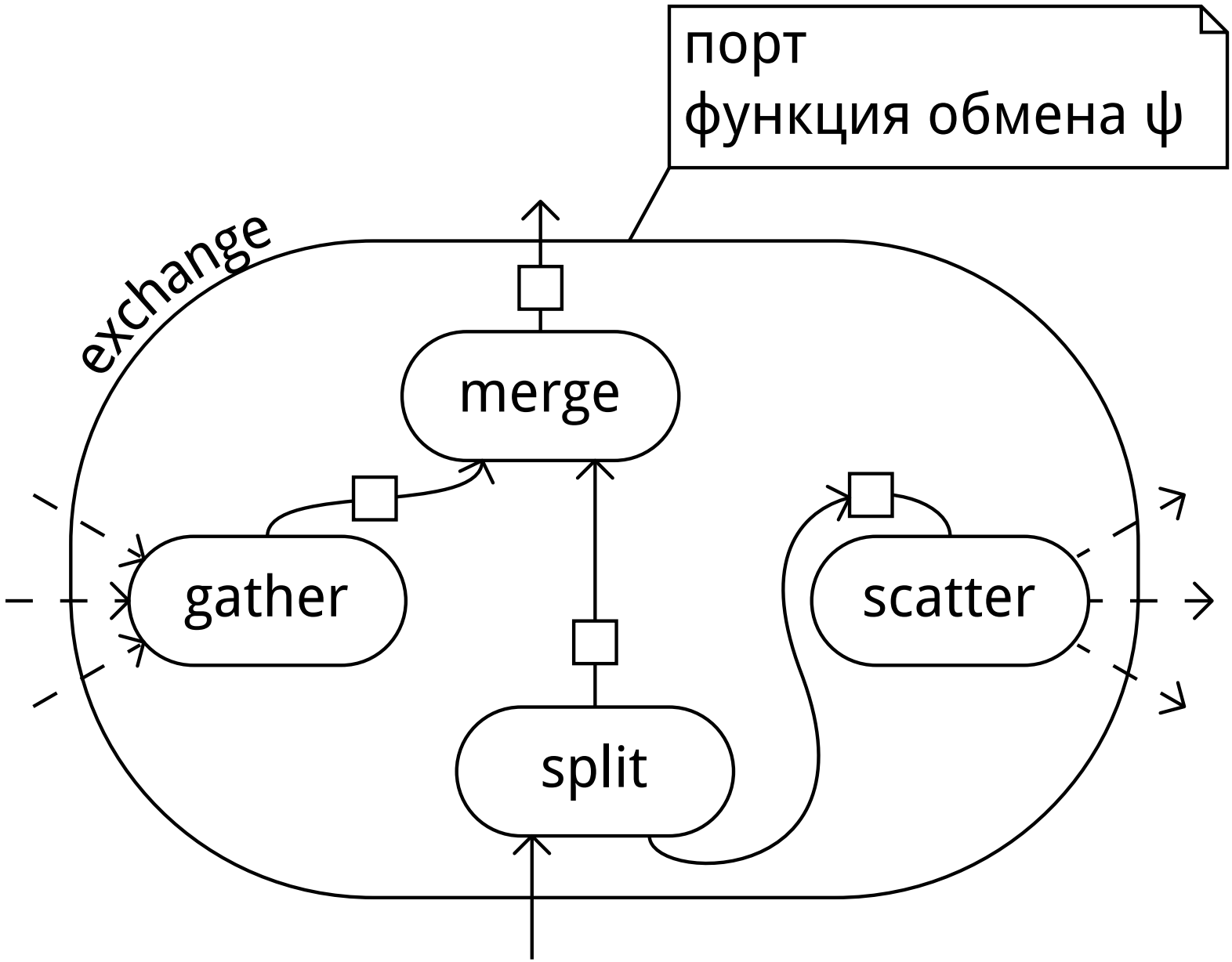
Методы внедрения фрагментного параллелизма

1. Тиражирование запроса
2. Использование оператора обмена
3. Построение параллельного плана запроса
4. Обработка запросов на изменение данных
5. Хранение метаданных о фрагментации
6. Портирование приложений последовательной СУБД
7. Модификация исходных текстов последовательной СУБД

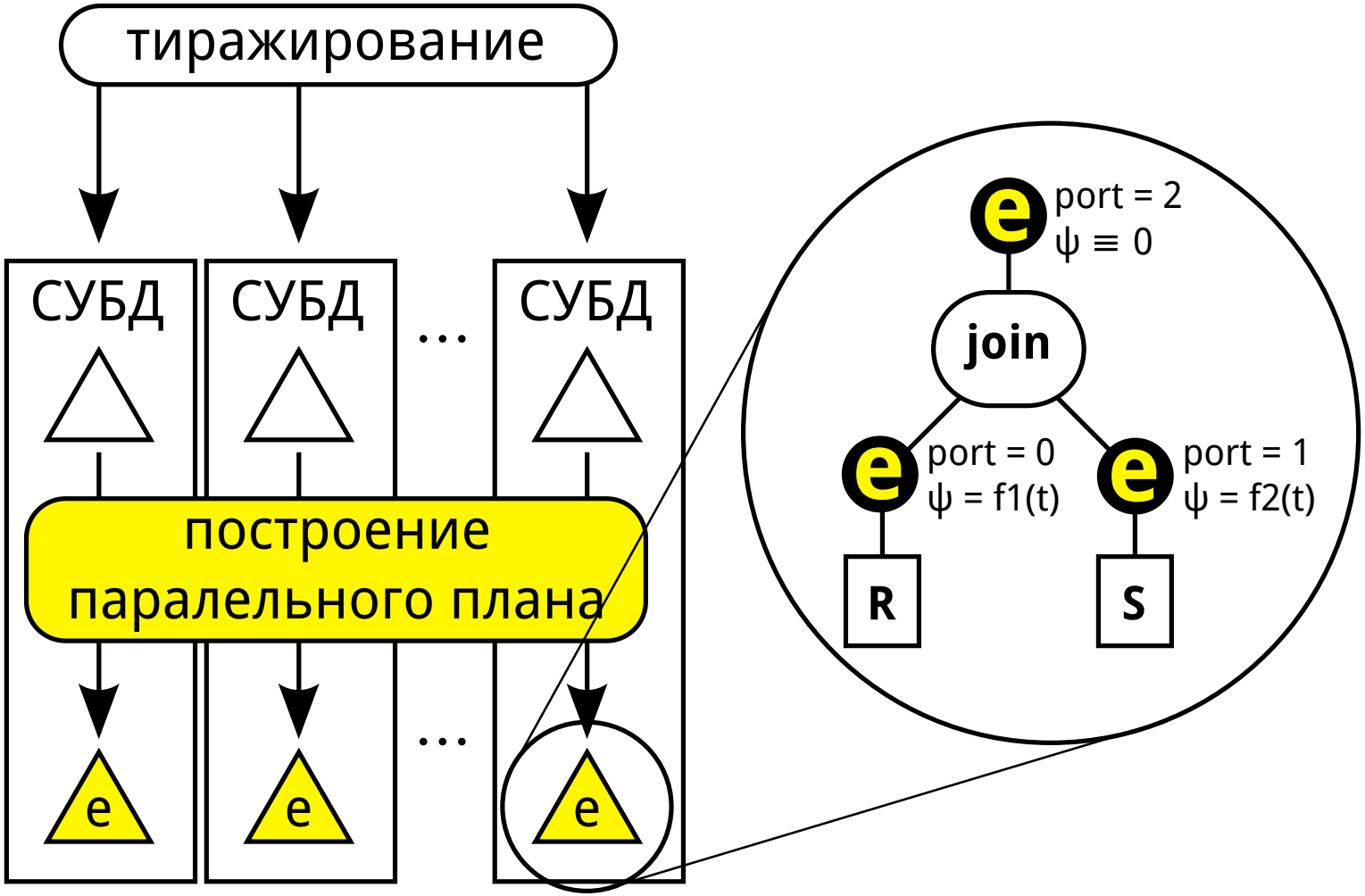
Тиражирование запроса



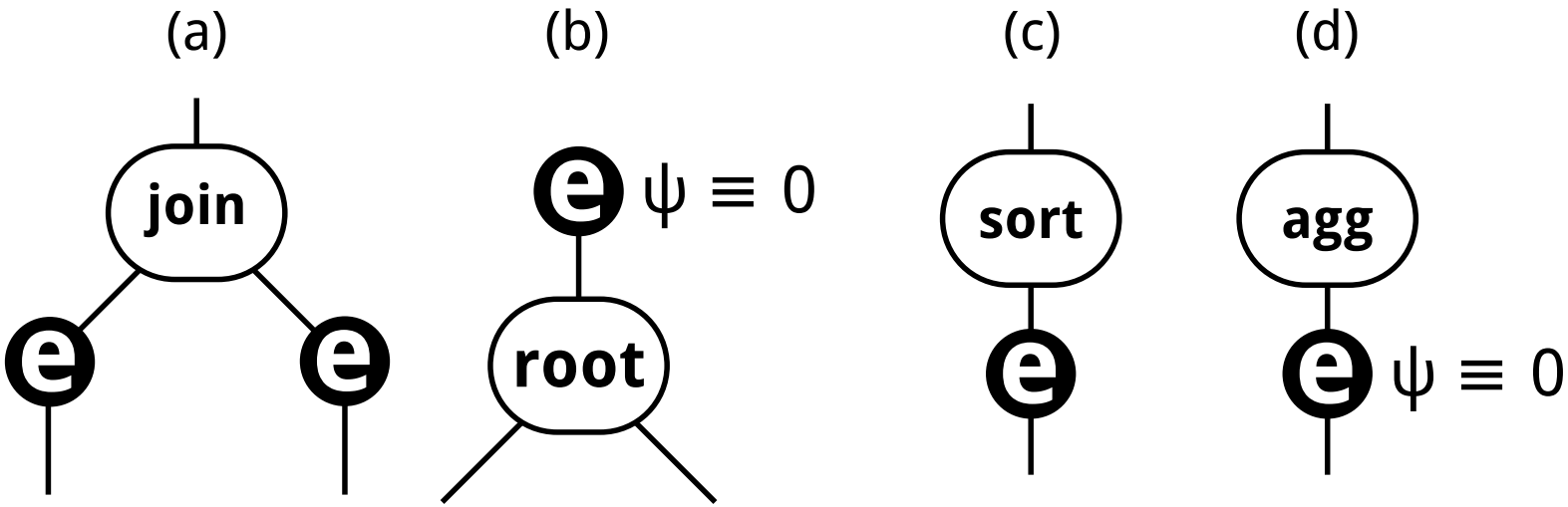
Оператор обмена (exchange)



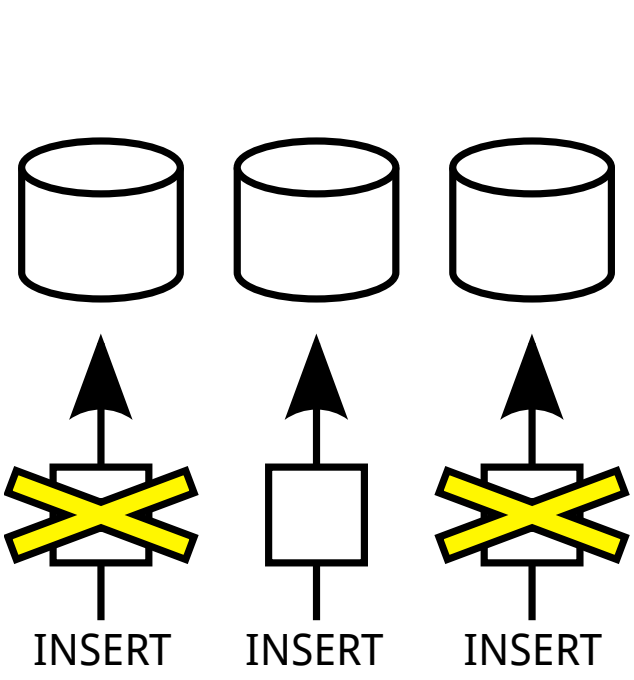
Построение параллельного плана запроса

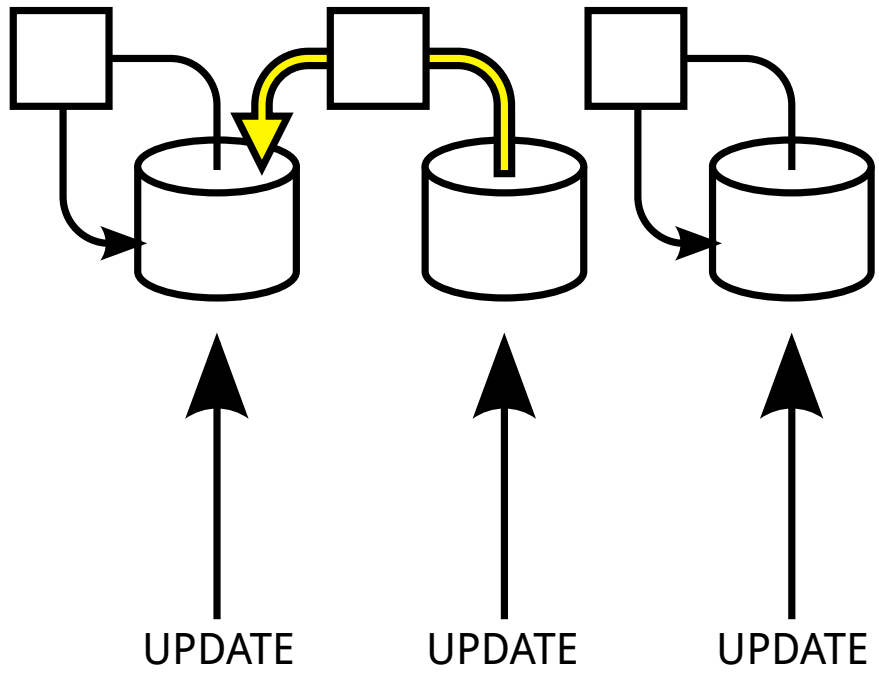


Построение параллельного плана запроса



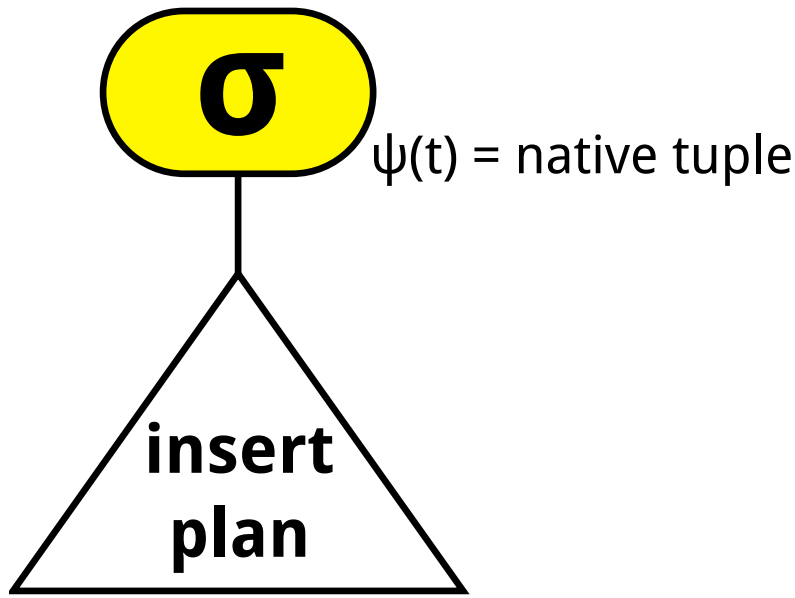
Обработка запросов на изменение данных



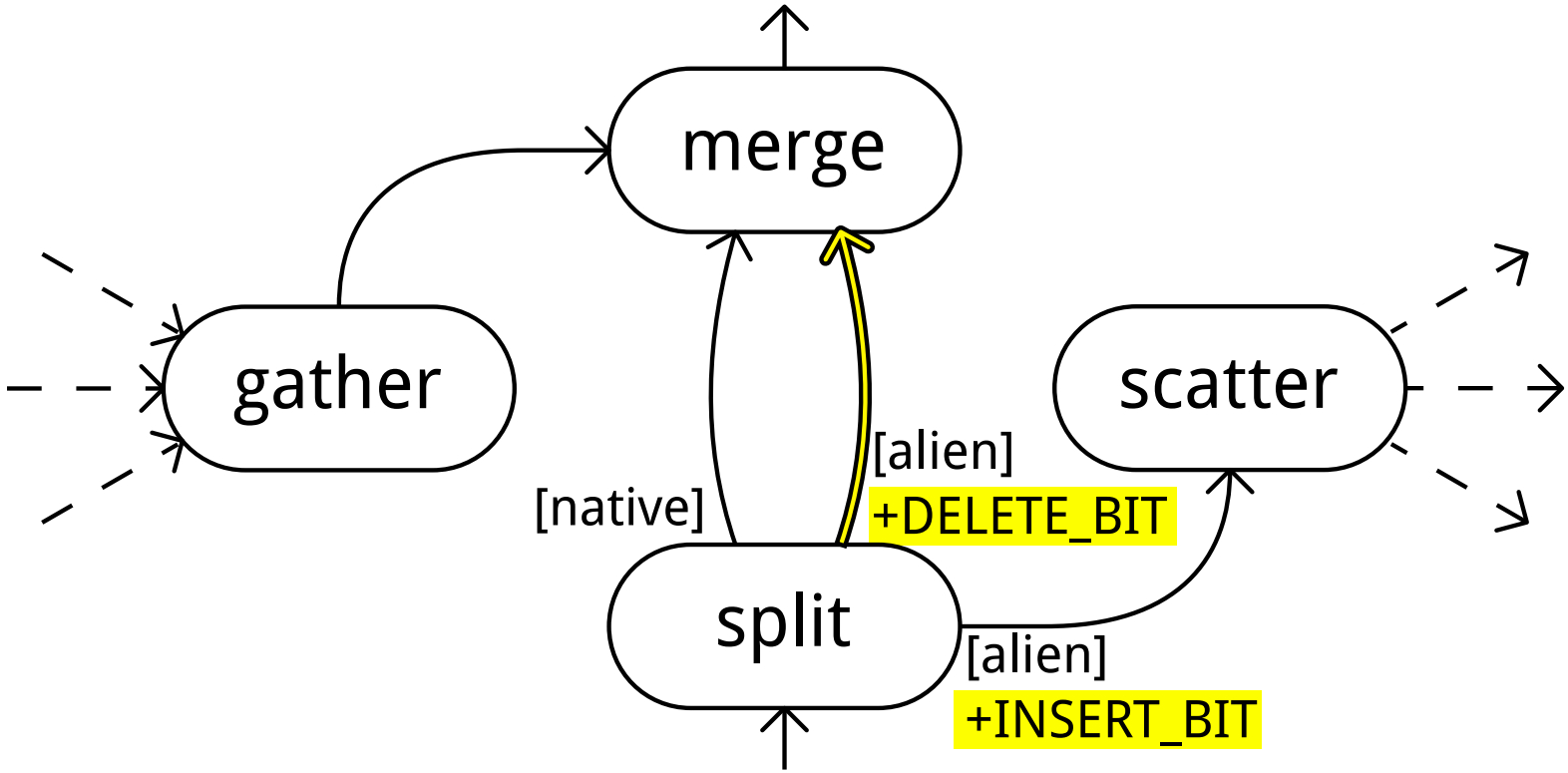


Обработка запросов INSERT

Дополнительная операция выборки в корне плана запроса обеспечивает отсечение «чужих» кортежей.



Обработка запросов UPDATE



Модифицированный алгоритм exchange позволяет перемещать кортежи, которые в результате обновления стали «чужими» (если $\varphi(t') \neq \varphi(t)$, то на узле $\varphi(t)$ кортеж t удаляется, а на узле $\varphi(t')$ вставляется t').

Хранение метаданных о фрагментации

Добавление в язык баз данных средств определения функции фрагментации для таблиц.

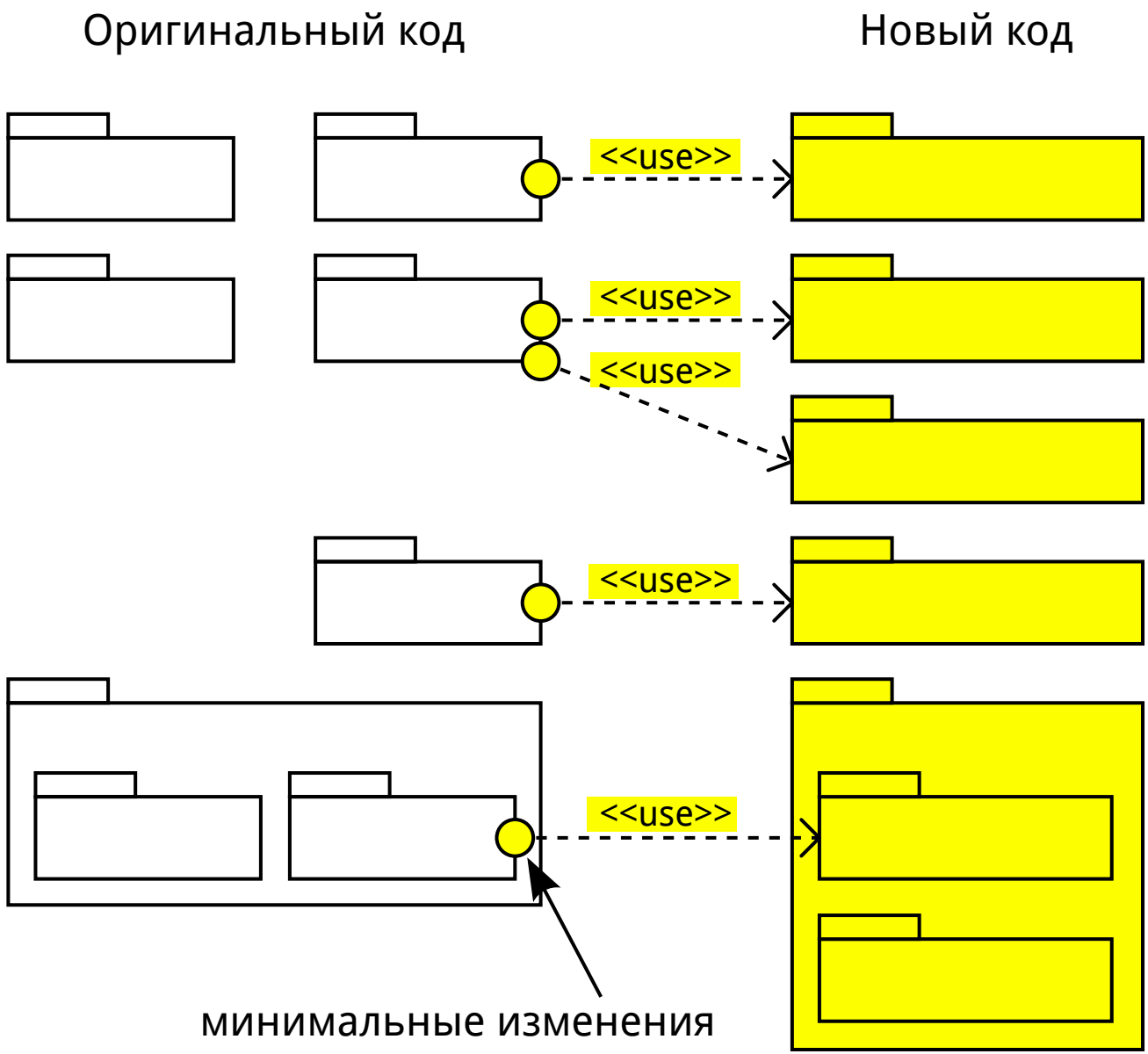
```
create table Person (
    id int,
    name char(30),
    gender char(1),
    birth date,
) with (функция фрагментации);
```

Портирование приложений последовательной СУБД

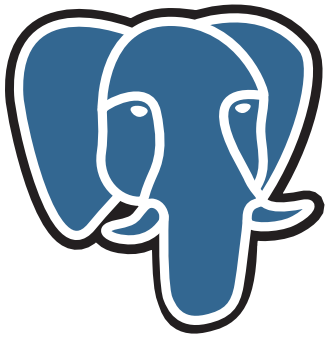
СУБД

Параллельная СУБД

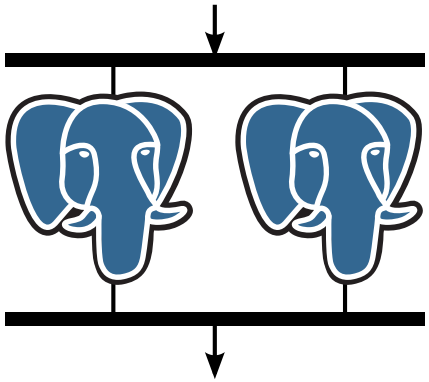
Модификация исходных текстов последовательной СУБД



Реализация методов



PostgreSQL

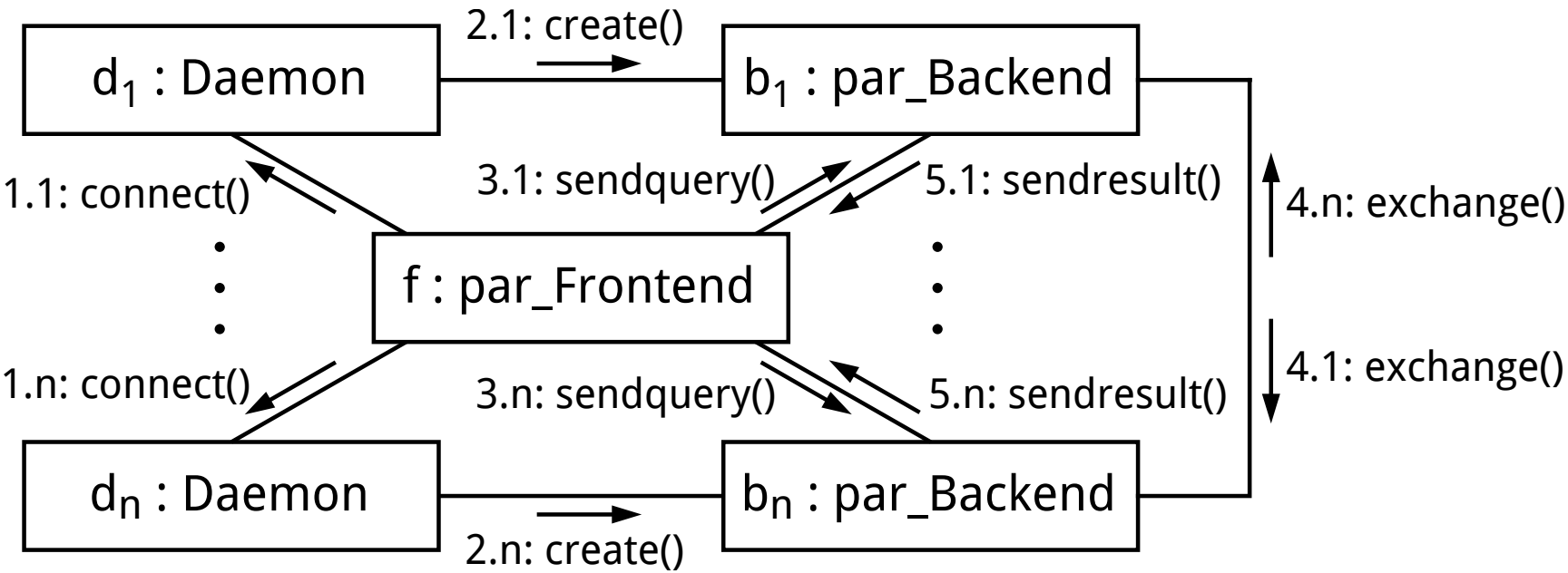


PargreSQL

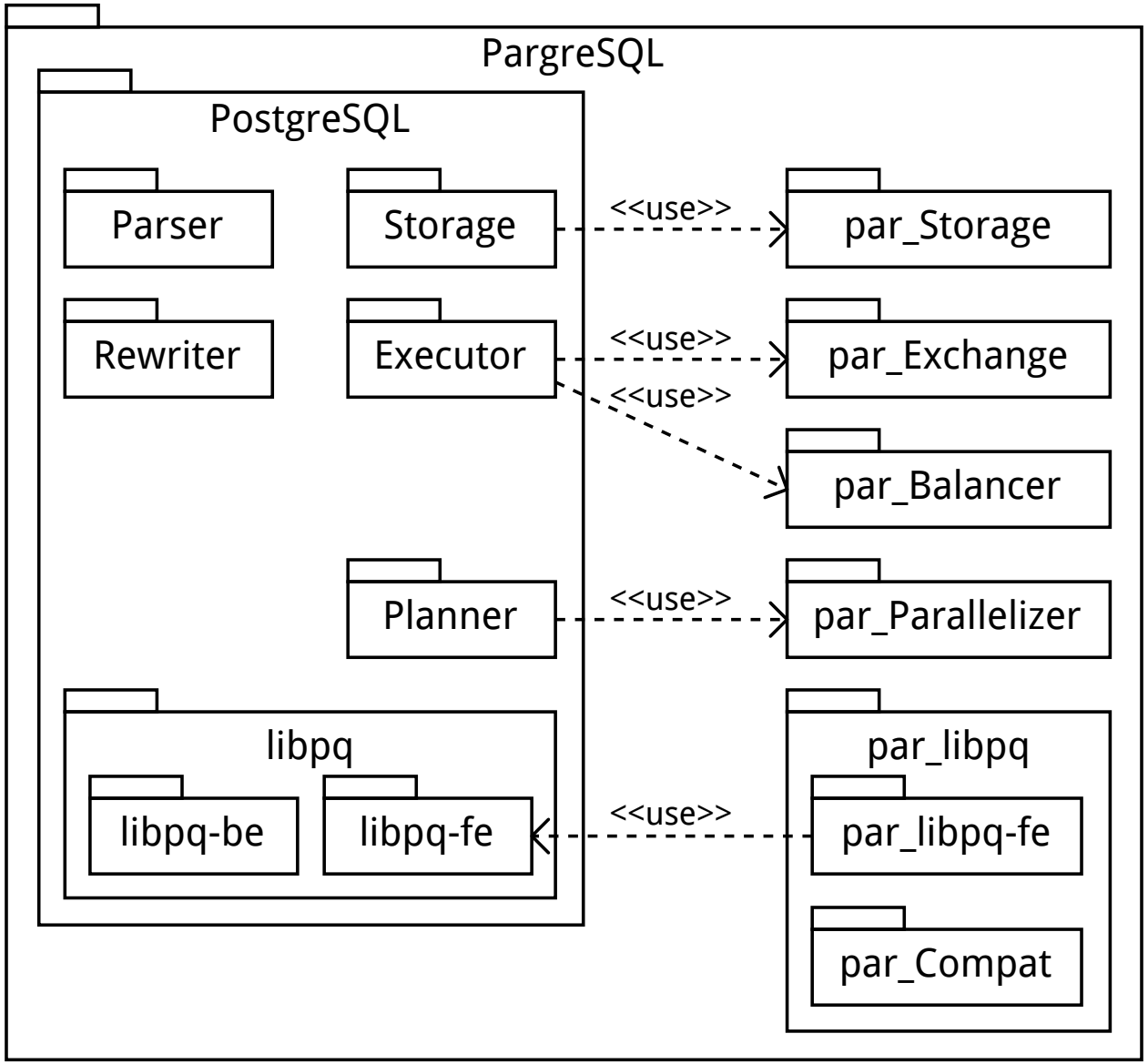
PostgreSQL

- Свободная объектно-реляционная СУБД с открытым исходным кодом для различных ОС
- Разрабатывается группой энтузиастов с 1995 г. (ответвление проекта POSTGRES М. Стоунбрейкера)
- Основные особенности
  - Поддержка SQL:2011 и ACID-транзакций
  - Хранимые процедуры SQL, pgSQL, Perl, Python, C и др.
  - Макс размеры: таблица – 32 Тб, поле – 1 Гб
  - Качественная документация исходного кода
- Надежная альтернатива коммерческим СУБД
- Широкое применение
  - Корпорации: Apple, Sun, Cisco, Fujitsu, Red Hat, ...
  - Госструктуры: U.S. State Department, United Nations Industrial Development Organization, ...

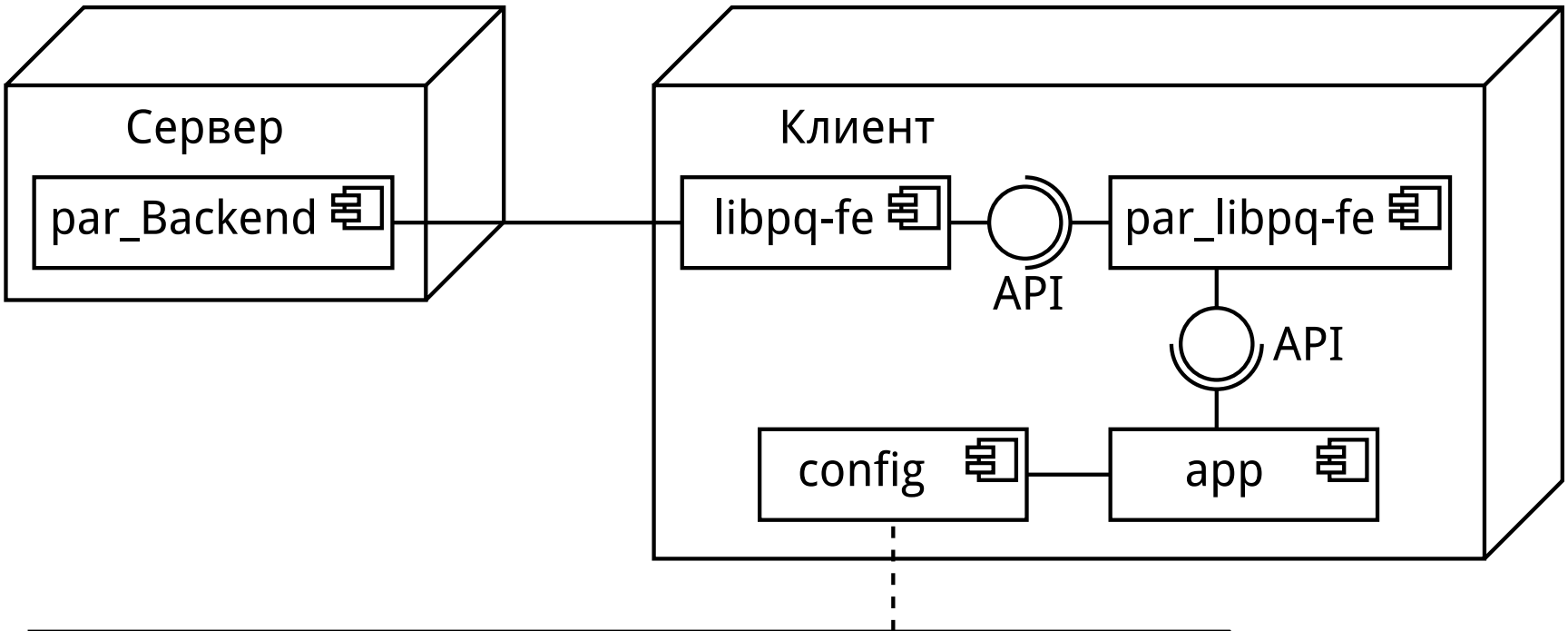
Клиент-серверное взаимодействие



Архитектура PargreSQL



Тиражирование

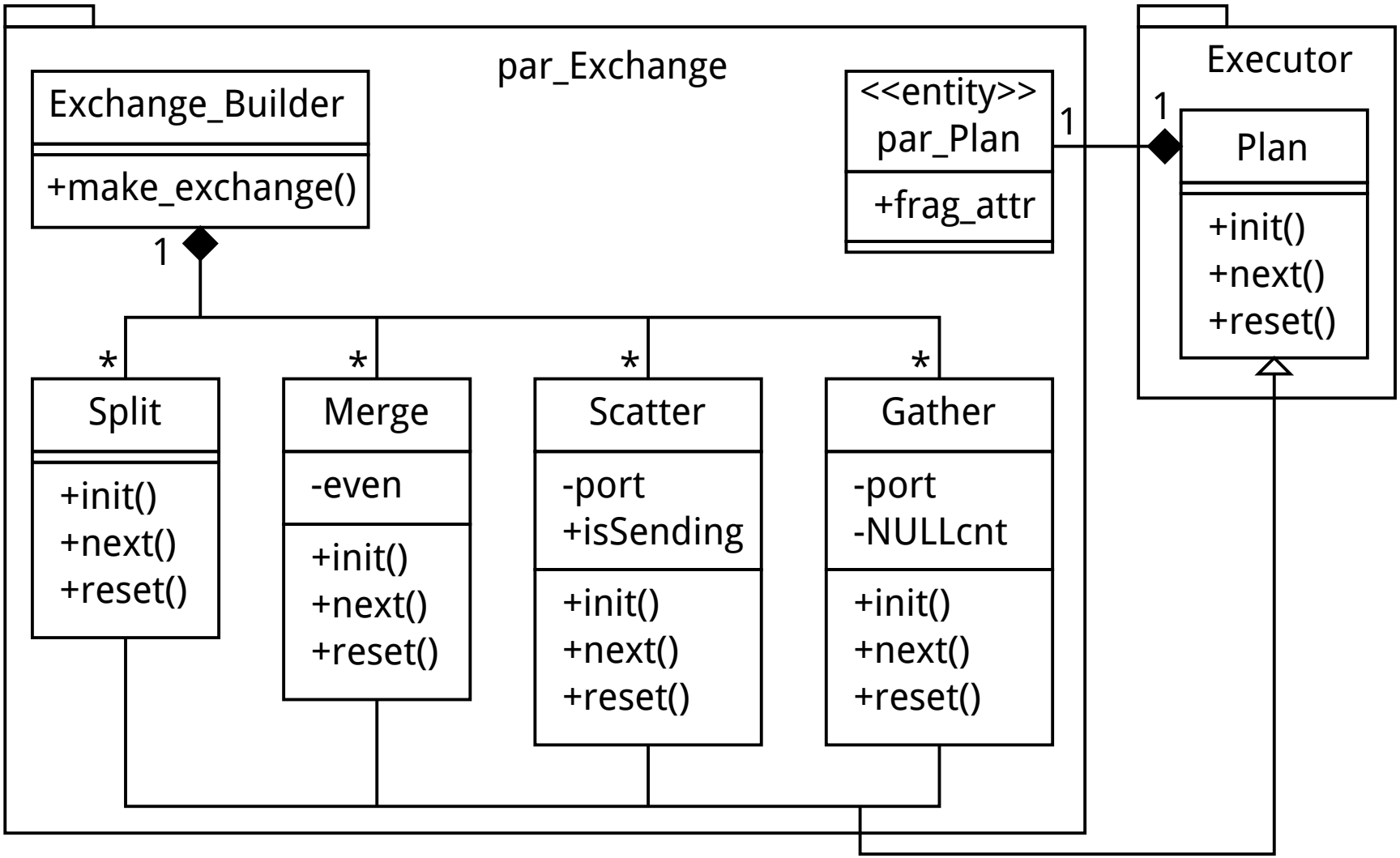



```
# Список строк подключения к экземплярам СУБД
# (по одной строке на каждый узел кластера)
dbname=postgres hostaddr=10.4.5.204 port=5432
dbname=postgres hostaddr=10.4.5.205 port=5432
```

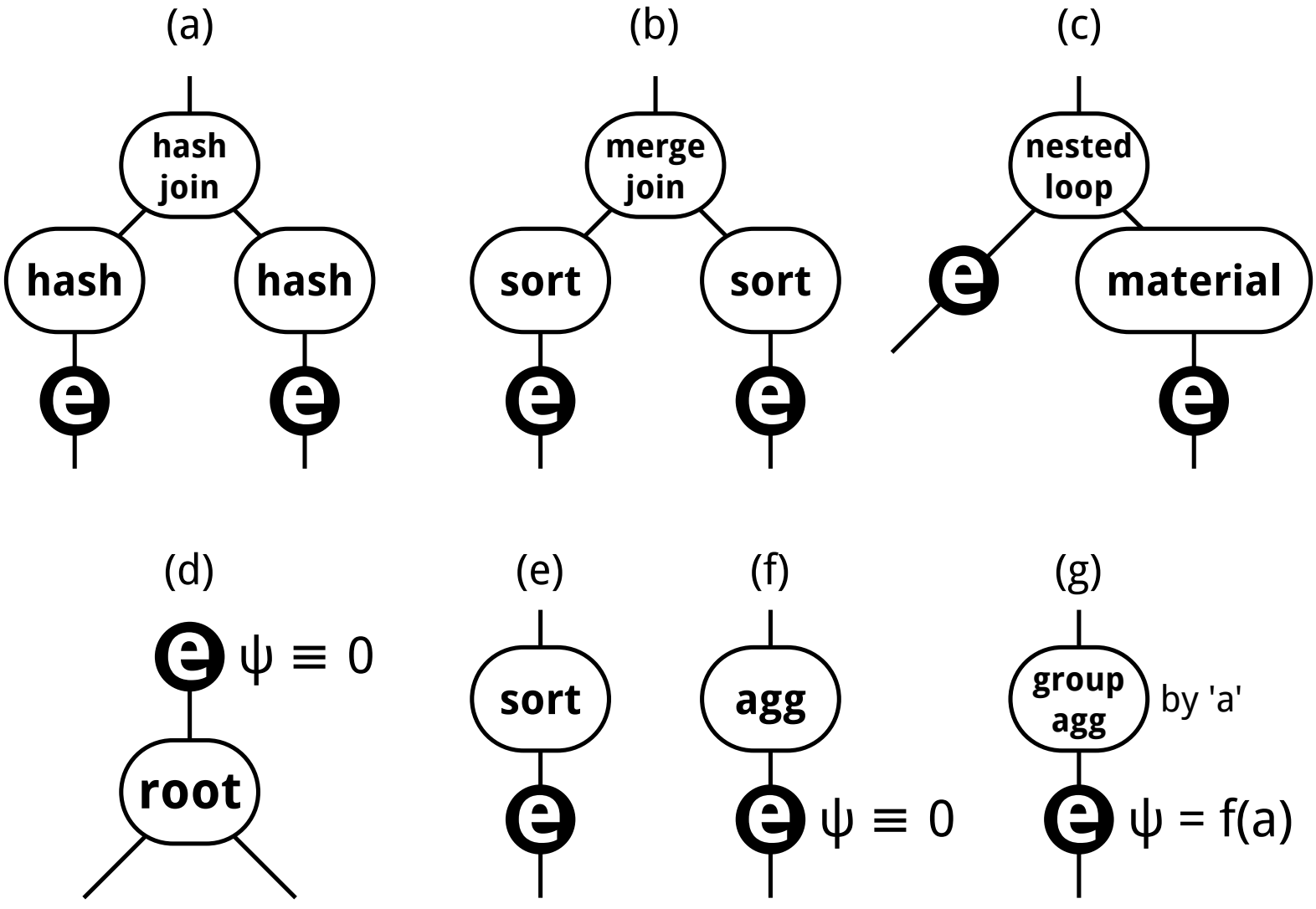
Портирование

```
#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)
```

Реализация exchange



Построение параллельного плана запроса



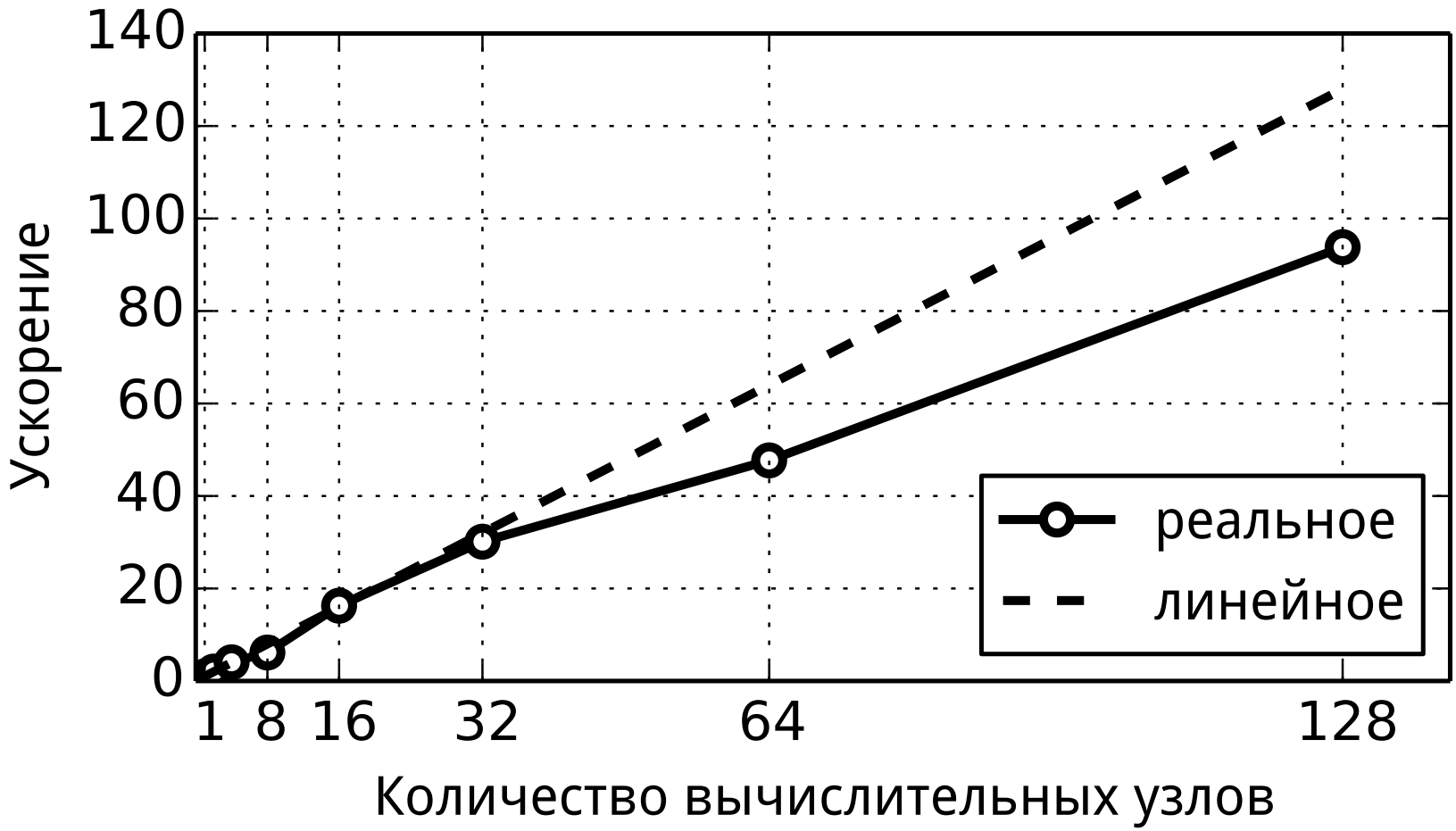
Запросы на определение функции фрагментации

```
create table Person (
    id int,
    name varchar(30),
    gender char(1),
    birth date
) with (fragattr = id);
```

Определит функцию фрагментации
для таблицы Person как $\varphi(t) = t.id \mod n$,
где n — количество вычислительных узлов.

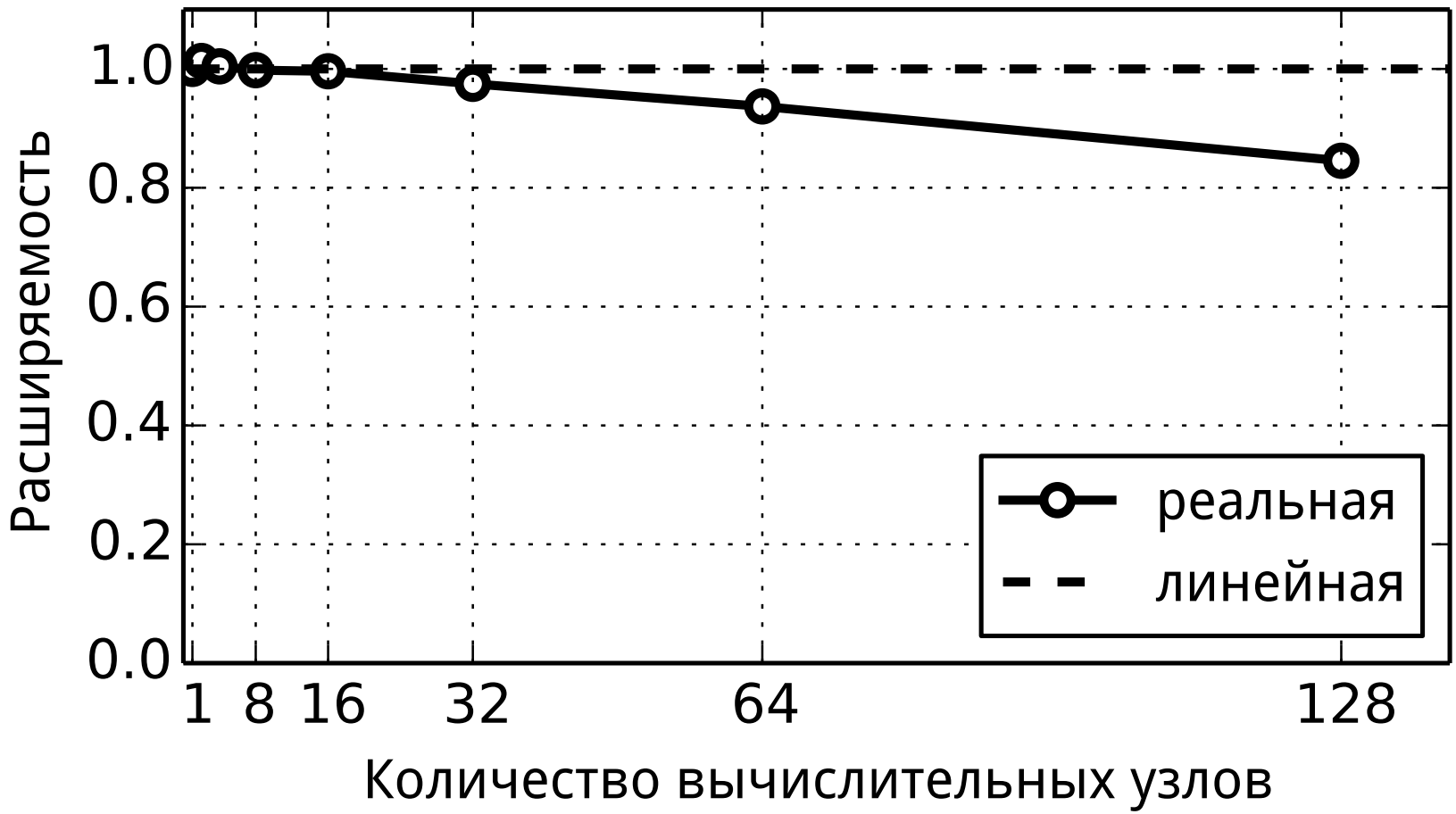
Ускорение параллельной СУБД

$$R \bowtie S, \qquad |R| = 3 \cdot 10^8, \quad |S| = 7.5 \cdot 10^6$$



Расширяемость параллельной СУБД

$R \bowtie S, \qquad |R_i| = i \cdot 12 \cdot 10^6, \quad |S_i| = i \cdot 0.3 \cdot 10^6$



Производительность на тесте TPC-C (моделирование складского учета)

К-во клиентов	tpm-C ↓
29	2202531
26	2197183
30	2195122
32	2194285
27	2189189
31	2188235
28	2181818
25	2173913

К-во клиентов	tpm-C ↓
24	2165413
23	2156250
22	2146341
20	2068965
19	2054054
18	2037735
21	2016000
17	1961538

К-во клиентов	tpm-C ↓
16	1882353
15	1747572
14	1647058
13	1529411
12	1358490
11	1346938
10	1290322
9	1270588

К-во клиентов	tpm-C ↓
8	1156626
7	1150684
5	857142
6	847058
4	657534
3	444444
2	328767
1	150000

Производительность на тесте TPC-C

№ п/п	Кластер	СУБД	К-во узлов	К-во клиентов	tpm-C
1	SPARC SuperCluster with T3-4 Servers	Oracle 11g R2 Enterprise Edition w/RAC w/Partitioning	108	81	30 249 688
2	IBM Power 780 Server Model 9179-MHB	IBM DB2 9.7	24	96	10 366 254
3	Sun SPARC Enterprise T5440 Server Cluster	Oracle 11g Enterprise Edition w/RAC w/Partitioning	48	24	7 646 486
	Торнадо ЮУрГУ	PargreSQL	12	29	2 202 531
4	HP Integrity rx5670 Cluster Itanium2/1.5 GHz-64p	Oracle 10g Enterprise Edition	64	80	1 184 893

Основные результаты

1. Разработаны новые методы внедрения фрагментного параллелизма в свободную СУБД с открытым исходным кодом.
2. На основе разработанных методов предложены архитектурные подходы и алгоритмы, реализующие фрагментный параллелизм в рамках последовательной СУБД с открытым исходным кодом, на базе которых выполнена параллелизация СУБД PostgreSQL.
3. Проведены вычислительные эксперименты с СУБД PargreSQL, которые показали успешное применение данной СУБД для решения задач, связанных с обработкой сверхбольших баз данных.

Публикации в журналах из списка ВАК

1. *Пан К.С.* Разработка параллельной СУБД на основе PostgreSQL // Труды Института системного программирования РАН. 2011. Т. 21. С. 357–370.
2. *Пан К.С., Цымблер М.Л.* Разработка параллельной СУБД на основе последовательной СУБД PostgreSQL с открытым исходным кодом // Вестник ЮУрГУ. Серия «Математическое моделирование и программирование». 2012. № 18(277). Вып. 12. С. 112–120.
3. *Pan C., Zymbler M.* Taming Elephants, or How to Embed Parallelism into PostgreSQL // Database and Expert Systems Applications – 24th International Conference, DEXA 2013, Prague, Czech Republic, August 26–29, 2013. Proceedings, Part I. Springer, 2013. Lecture Notes in Computer Science. Vol. 8055. P. 153–164.
4. *Pan C., Zymbler M.* Very Large Graph Partitioning by Means of Parallel DBMS // Proceedings of the 17th East-European Conference on Advances in Databases and Information Systems, ADBIS'2013, September 1–4, 2013, Genoa, Italy. Lecture Notes in Computer Science. Springer, 2013. Vol. 8133. P. 388–399.

Апробация работы — 8 выступлений

1. Международная научная конференция DEXA'2013 (The 24th International Conference on Database and Expert Systems Applications) (Чешская Республика, Прага, 26–30 августа 2013 г.).
2. Международная научная конференция ADBIS'2013 (The 17th East-European Conference on Advances in Databases and Information Systems) (Италия, Генуя, 1–4 сентября 2013 г.).
3. Международная научная конференция SYRCoDIS'2011 (The 7th Spring Researchers Colloquium on Databases and Information Systems) (Москва, 2–3 июня 2011 г.).
4. Международная суперкомпьютерная конференция «Научный сервис в сети Интернет: все грани параллелизма» (Новороссийск, 23–28 сентября 2013 г.).
5. Международная научная конференция «Параллельные вычислительные технологии (ПаВТ'2011)» (Москва, 28 марта – 1 апреля 2011 г.).

Свидетельства о регистрации программ

Соколинский Л.Б., Цымблер М.Л., Пан К.С., Медведев А.А.
Свидетельство Роспатента о государственной регистрации программы для ЭВМ «Параллельная СУБД PargreSQL»
№ 2012614599 от 23.05.2012.

Гранты

Работа выполнялась при поддержке *Российского фонда фундаментальных исследований* (проекты 12-07-31217 мол_а, 12-07-00443, 09-07-00241) и *Президента Российской Федерации* (стипендия СП-5427.2013.5).

Применение методов к другим СУБД

- Реляционные СУБД основаны на реляционной алгебре, поэтому применимы методы обмена и построения параллельного плана.
- Любая СУБД поддерживает словарь данных, поэтому применим метод хранения метаданных.
- Любая СУБД предполагает наличие прикладной библиотеки, поэтому применимы методы тиражирования и портирования.

Зачем нужны обмены?

Фрагмент 0

Поставщики

пID	пName
0	Поставщик0
2	Поставщик2

Детали

дID	дName
0	Деталь0
2	Деталь2

Поставки

пID	дID
0	1
2	3

Фрагмент 1

Поставщики

пID	пName
1	Поставщик1
3	Поставщик3

Детали

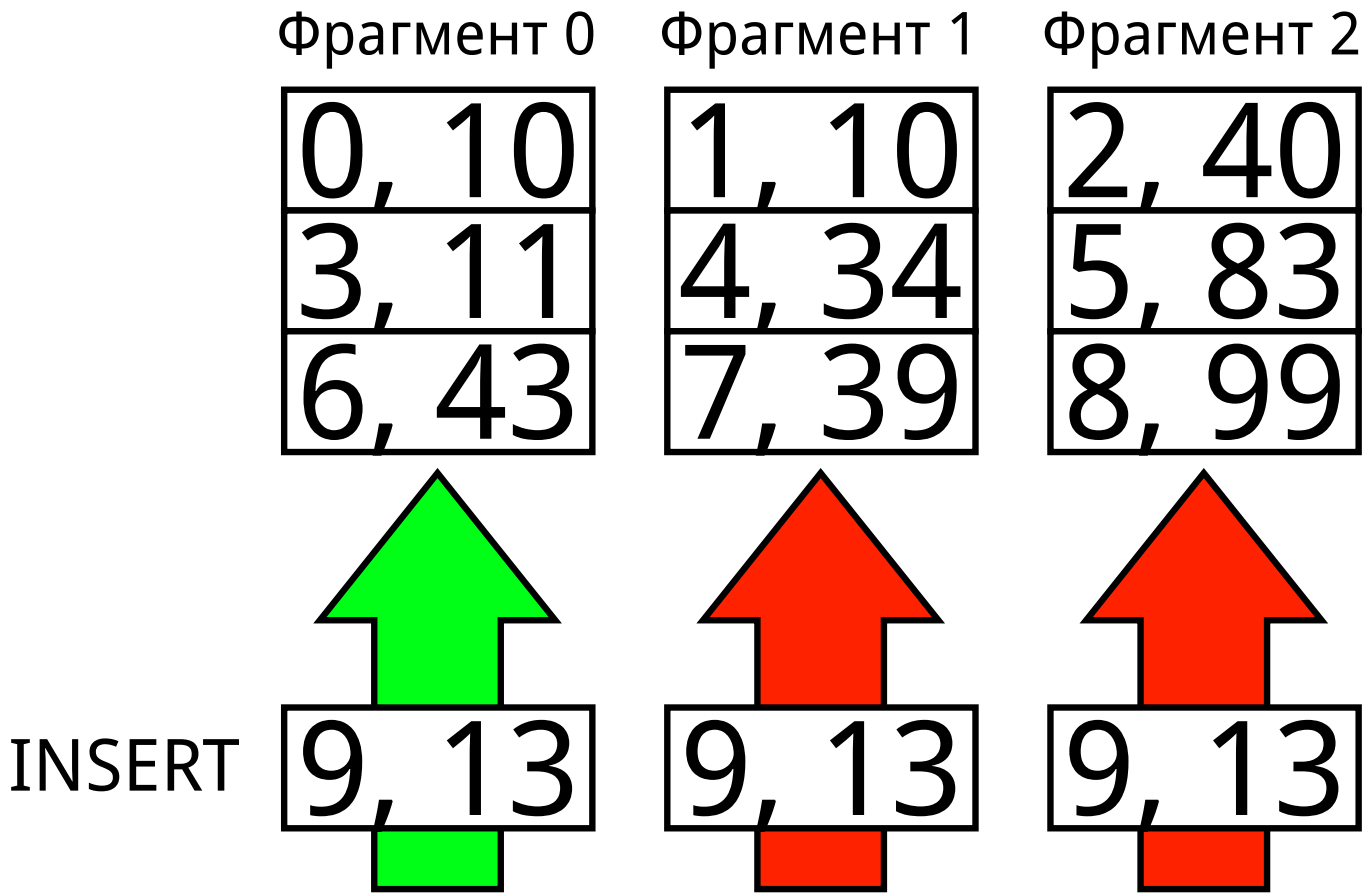
дID	дName
1	Деталь1
3	Деталь3

Поставки

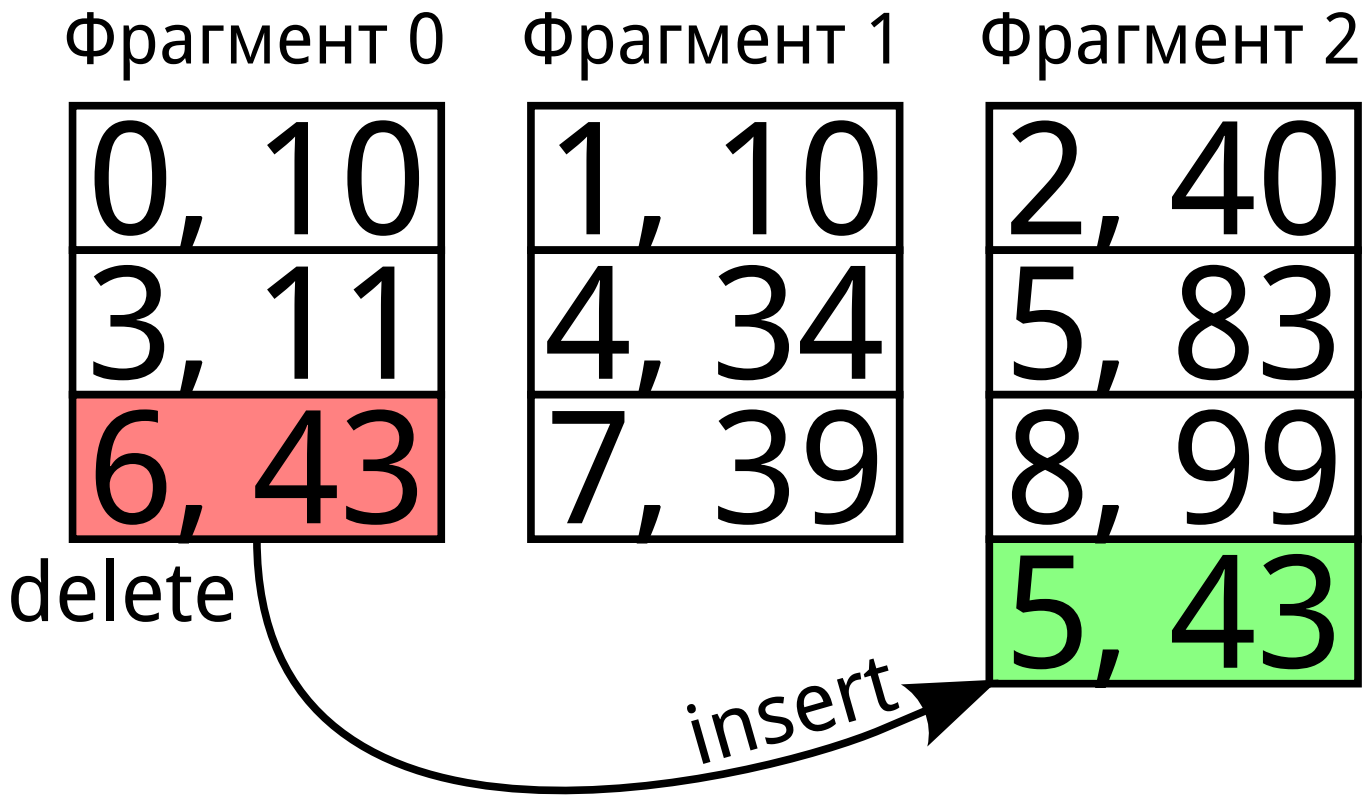
пID	дID
1	0
3	2

```
select пName, дName
from Поставщики, Детали, Поставки
where Детали.дID = Поставки.дID
and Поставщики.пID = Поставки.пID;
```

Пример вставки кортежа



Пример обновления кортежа



UPDATE t SET a = 5 WHERE a = 6;