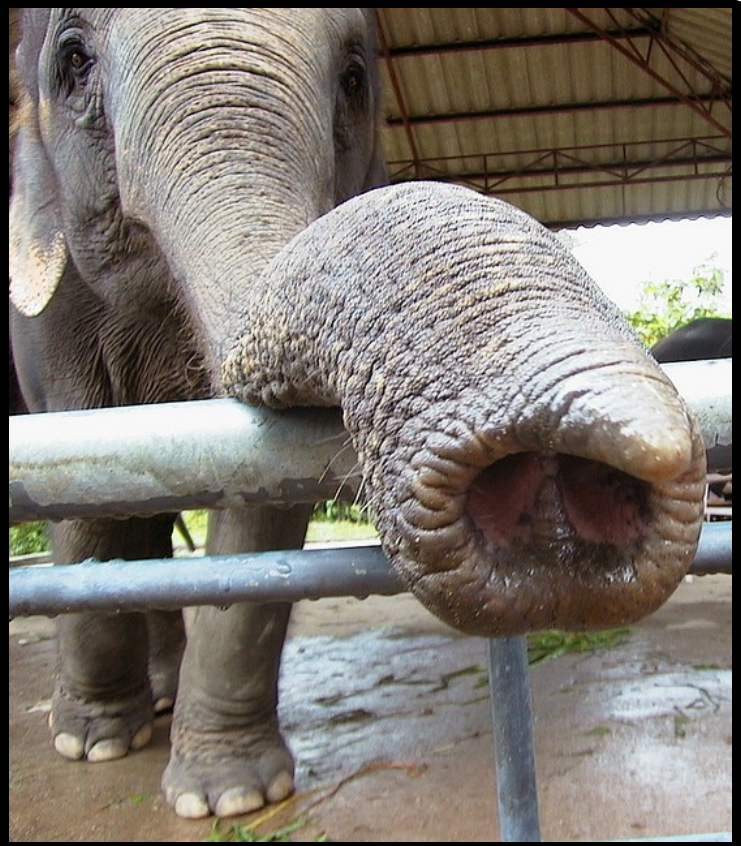
Redpill Linpro

A look at the elephants trunk



http://www.flickr.com/photos/aussy_greg/255942923/

Open Source Days 2012
Copenhagen, Denmark

Magnus Hagander
magnus@hagander.net

PostgreSQL 9.2

- Is not yet done!
- Feature set still uncertain
- Many things are done
  - Some completed
  - Some partial
- We give no guarantees, sorry...
- Many “invisible” changes, not included here

Development Schedule

- June 11, 2011: 9.1 branched, HEAD opened
- July 2011: Commitfest #1
- Sep 2011: Commitfest #2
- Nov 2011: Commitfest #3
- **Jan 2012-: Commitfest #4** → Alpha 4?
- Beta releases
- Release candidates (before summer?)
- Release (after summer?)

Current state of tree

- CF4 in progress
- `2051 files changed, 236771 insertions(+), 80419 deletions(-)`
- Already more than 9.1!

Many new features

- DBA and developer
- Replication and backup
- Performance
- Scalability

Many new features

- DBA and developer
- Replication and backup
- Performance
- Scalability

Many new features

- **DBA and developer**
- Replication and backup
- Performance
- Scalability

pg_stat_activity restructured

- Explicit state field
  - Running, idle, idle in transaction etc
- current_query removed
- Instead we have query
  - Current query when state=running
  - Otherwise, last query

pg_cancel_backend()

- A user can cancel his/her own queries
- Even when connecting in a different session
- Previously required superuser

Security barrier views

- Ability to create row-level security enforcing VIEWs
- Performance cost due to optimization barrier

```
CREATE VIEW sec
WITH (security_barrier=yes)
AS SELECT foo
FROM bar WHERE x=y
```

Range Datatypes

- Arbitrary range datatypes
  - Generalized version of e.g. period
- Store start/stop values
- Indexed lookups, exclusion constraints

```
CREATE TABLE bookings(room int,
during tsrange);
INSERT INTO bookings VALUES (1,
 '[2012-03-09 10:00, 2012-03-09 11:00]');
```

JSON datatype

- Native JSON datatype
- Currently only does JSON validation
  - Future Improvements Expected (TM)

```
CREATE TABLE mytable (
  id int,
  j JSON
);
```

PL/V8

- Not actually in PostgreSQL core
- Extension works with previous versions as well
- Integrates well with JSON datatype
  - E.g. expressional indexes on JSON extraction

Many new features

- DBA and developer
- **Replication and backup**
- Performance
- Scalability

Cascading replication

- Ability to use a replication slave as a relay
- Master doesn't need to talk to all replicas
- Off-loading processing or network
- Does *not* support synchronous mode

New sync mode: “write”

- synchronous_commit=”on” means “release transaction when data is on disk on slave”
- synchronous_commit=”write” means “release transaction when data is in memory on slave”
- Data loss if both master and slave crashes
- Higher performance (RAM > disk)

Base backups from standby

- Run backups from slave instead of master
- Off-load master
- Only backups with pg_basebackup supported

Streaming log archiver

- Create log archive using streaming replication
- Avoids archive_command tradeoffs:
  - No half-empty 16Mb blocks
  - No long delays before shipping

pg_streamrecv -h server -D archivedir

Streaming log archiver

- Can also run during base backups
- Avoid requirement for high wal_keep_segments just for backups
- In non-archived scenarios

pg_basebackup --xlog=stream

Many new features

- DBA and developer
- Replication and backup
- **Performance**
- Scalability

Index Only Scans

- If query contains only indexed columns, avoid lookup in heap
- Can avoid lots of I/O
- Only works on pages that are 100% all-visible
  - Uses visibility map
  - Partial index only scan often used

Better group commit

- When one transaction commits while another is waiting for disk
- Wake up multiple queued processes at once
  - Previous versions would do one by one
- Hopefully they can avoid flushing...
- Decreased lock contention

Faster sorting

- Inline sort operators for in-memory sorting
- Specialized fixed-size/structure aware version of quicksort

Space Partitioned GiST

- Current GiST are all balanced trees
- SP-GiST supports non-standard trees
  - K-D tree, Quadtree
    - CAD, GIS, multimedia etc
  - Suffix trees
    - Substring, IP networks etc

Many new features

- DBA and developer
- Replication and backup
- Performance
- **Scalability**

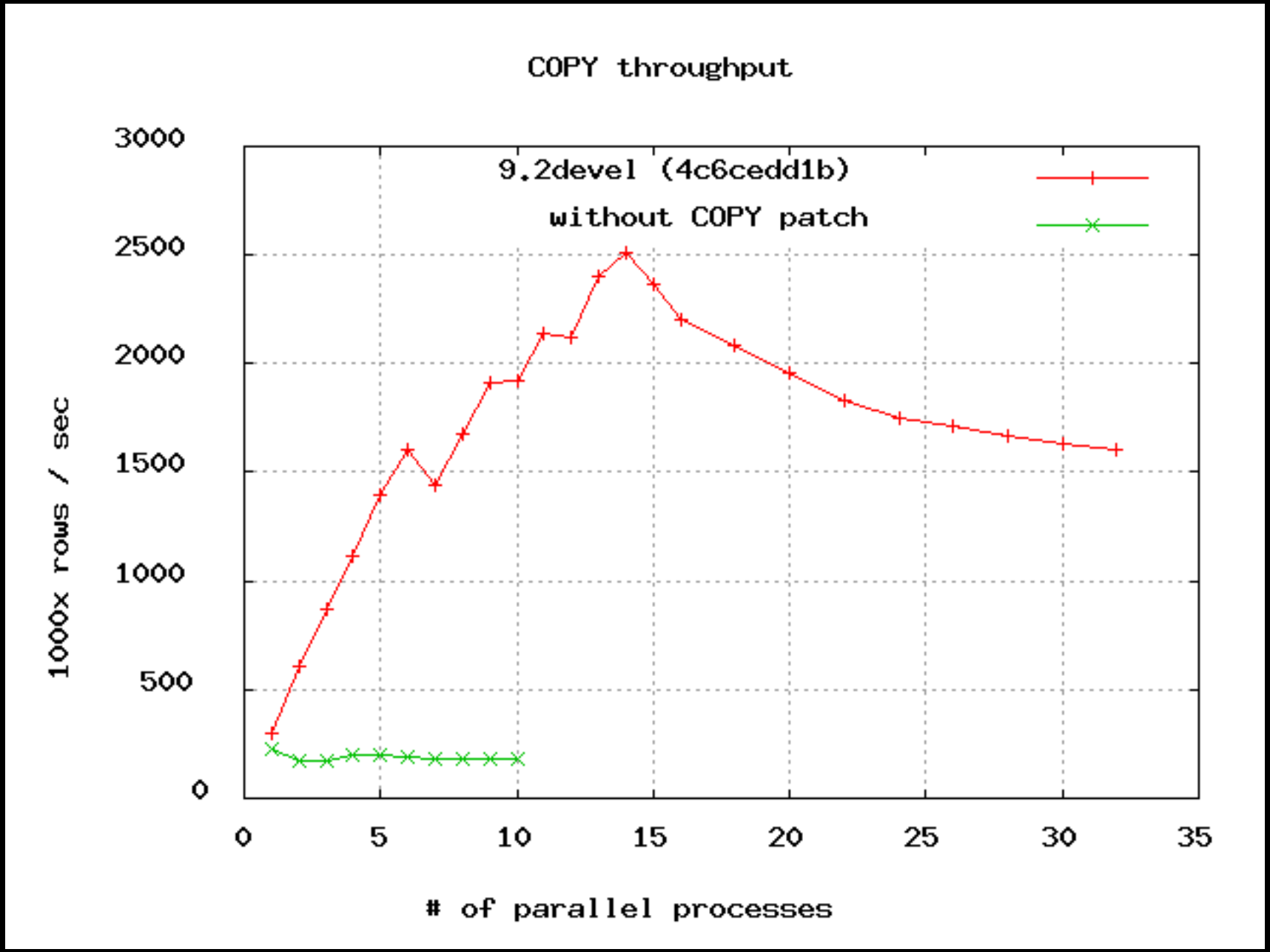
Scalability challenges

- Many core machines
- Many concurrent transactions
- *Short* transactions
  - (PostgreSQL already does very well on long transactions to >64 cores)
- Benchmark box:
  - 32 cores, ia64
  - ~380Gb RAM

COPY batch insert

- Tuples received by COPY are batched
- Reduced WAL logging
- Reduced locking
- Much better scalability to multiple loaders

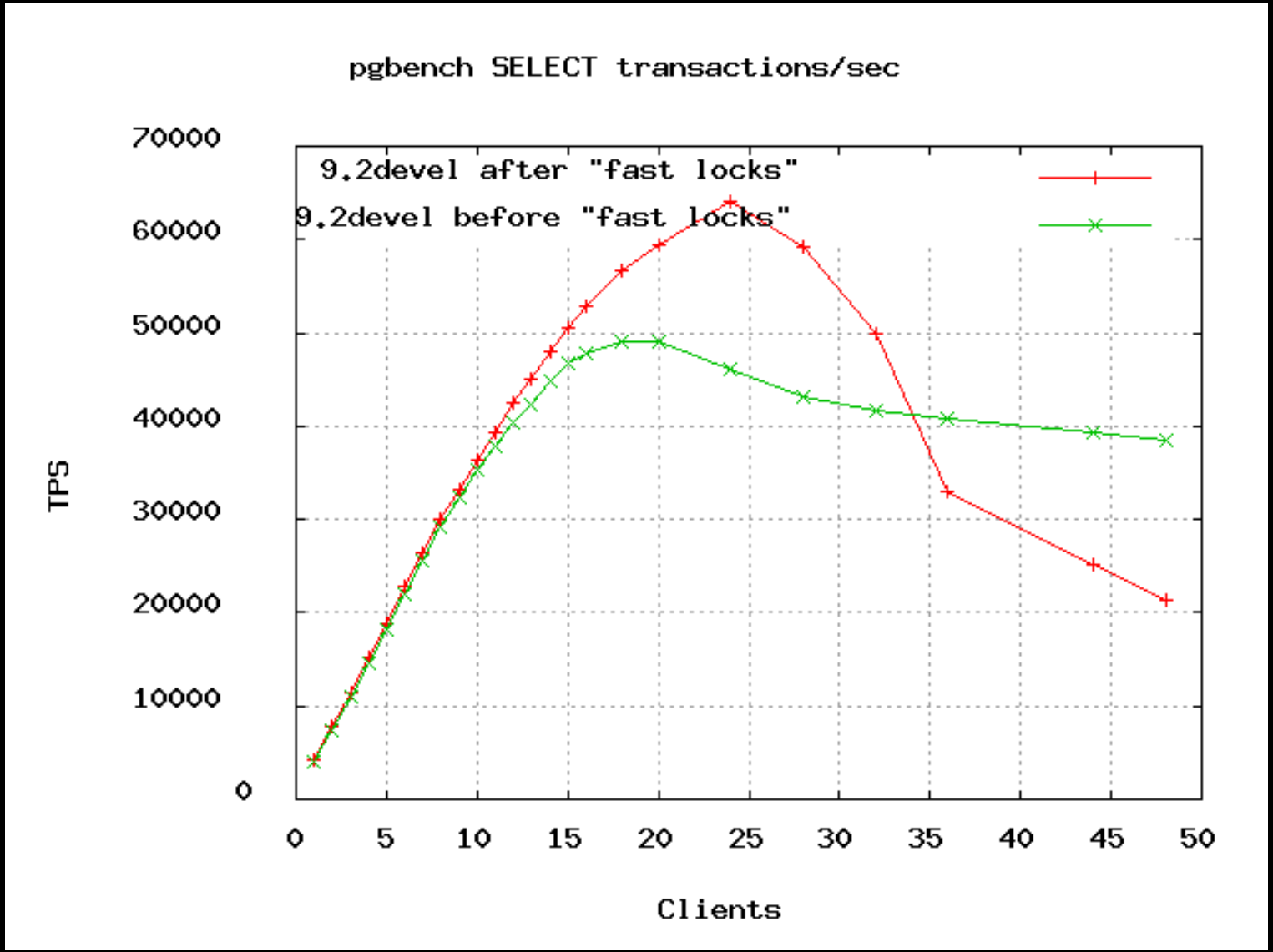
COPY batch insert



Fast path locking

- “Cheat” on non-exclusive locks
- “probably” nobody else will conflict
- Store locks locally instead of globally
- Only checked when someone tries to get exclusive lock
  - Exclusive-lock session pays the cost

Fast path locking



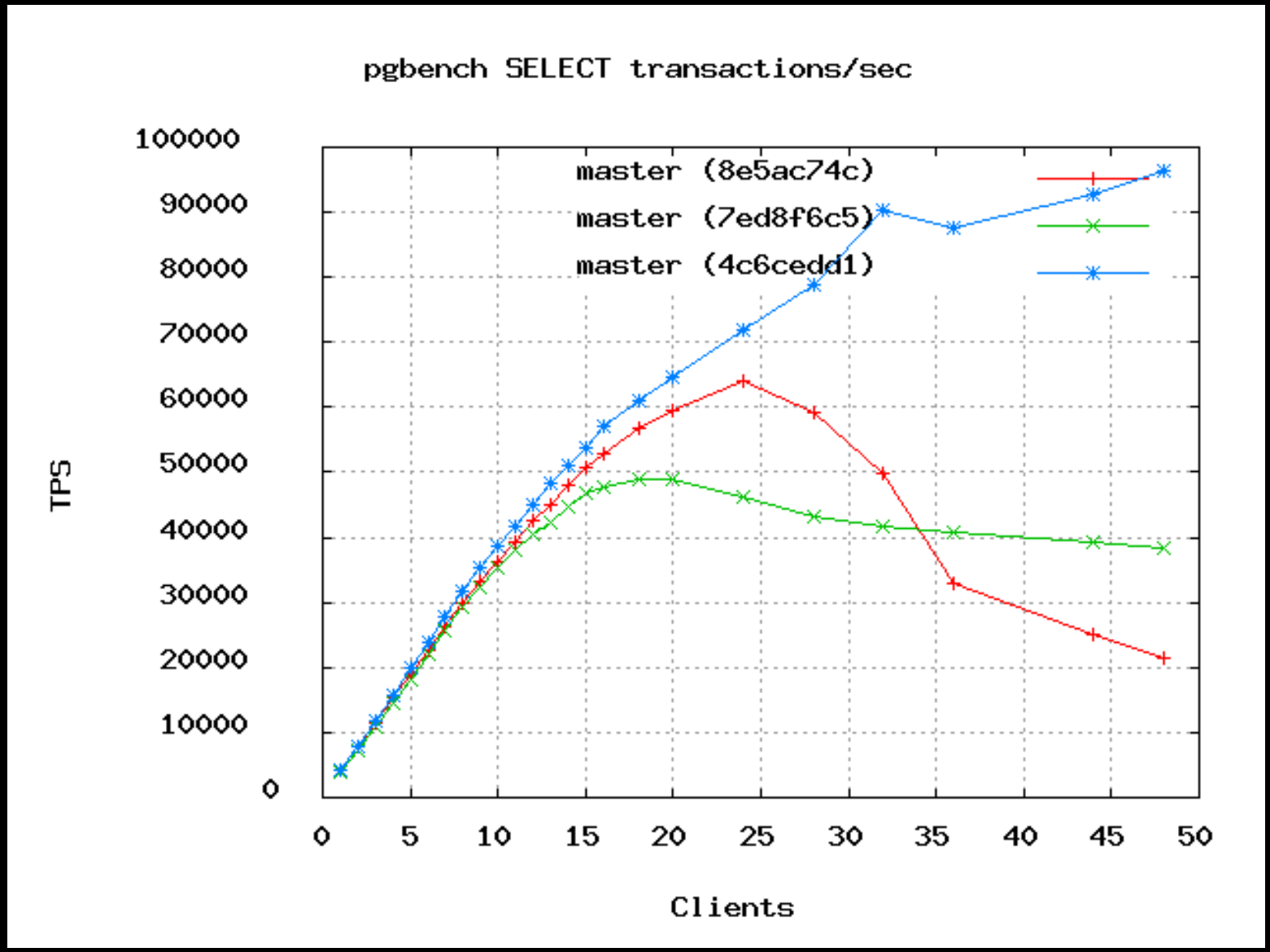
Split ProcArray

- All sessions have an entry in ProcArray
- Single global lock
- Shared lock whenever snapshots are taken
- Exclusive lock whenever transactions commit
- Split ProcArray into one array with “hot” elements and one with “cold”

More small stuff

- Lazy VXID creation
- Spinlock improvements
- Sinval sync overhead

Total scalability results



More to come?

- Some features queued up
- Unknown if they'll make it

Better write-scaling

- Refactoring and enhancement of XLogInsert
- Hold locks for shorter time
- More processing outside of locked sections

Command Triggers

- Assign trigger to utility commands
  - CREATE TABLE
  - ALTER TABLE
  - CREATE INDEX
  - Etc etc

FDW for pgsql

- Access remote PostgreSQL servers
- Should've had this from the start...
- Better optimizations
  - Join push-down etc

Foreign table statistics

- Collect frequency statistics on foreign tables
- Used for optimizing queries accessing remote tables
- Number of distinct values, MVCs, LCVs etc

Parallel pg_dump

- Based on snapshot exporting
- Get transactionally consistent dump across parallel sessions
- Increased performance in multi core systems

pg_stat_statements

- Better normalization
- Based on internal query tree representation

Even more?

- Several other things still discussed
- Feature freeze for new submissions!

Thank you!

Questions?

Twitter: @magnushagander
http://blog.hagander.net/
magnus@hagander.net

Thanks to Heikki Linnakangas, Greg Smith and Nathan Boley
for benchmarks, tools and graphs