one.. two.. three.. postgres go

Murat Kabilov
Adjust GmbH
2018-09-27

query cycles

- Simple
- Extended
  - Parse
  - Bind
  - Execute
  - Describe
  - Flush
  - Sync

```
demo=# \d+ my_table
                        Table "public.my_table"
 Column |          Type           | Collation | Nullable | ...
--------+-------------------------+-----------+----------+----------
 id     | integer                 |           |          |
 str    | character varying(10) |           |          |


demo=# select * from my_table;
 id |  str
----+-------
  1 | text1
  2 | [null]
```

simple query

```
select * from my_table where id < 3
```

Frontend → Backend:

  query string

Backend → Frontend:

  row description

  data row for the 1st row

  data row for the 2nd row

  command complete

  ready for query

simple query

Frontend → Backend:

byte1	int32	string
`Q`	`40`	`select * from my_table where id < 3`
simple query	message length	null-terminated query string

response

Backend → Frontend:
row description

byte1	int32	int16
T	49	2
row description	message length	number of fields

...

response

Backend → Frontend:
row description (description for column “id”)

	string	int32	int16	int32	int16	int32	int16	
...	id	627291	1	23	4	-1	0	...
	field name	table OID	Att num	type OID (int4)	typlen	atttypmod	format code*	

* — 0: text, 1: binary

response

Backend → Frontend:
row description (description for column “str”)

...

string	int32	int16	int32	int16	int32	int16
str	627291	2	1043	-1	14	0
field name	table OID	Att num	type OID (varchar)	typlen	atttypmod (varchar(10))	format code*

* — 0: text, 1: binary

response

Backend → Frontend:
data row (1 row)

byte1	int32	int16	int32	byteN	int32	byteN
D	20	2	1	1	5	text1
data row	message length	fields	length	value	length	value
			field 1		field 2	

response

Backend → Frontend:
data row (2 row)

byte1	int32	int16	int32	byteN	int32
D	15	2	1	2	-1
data row	message length	fields	length	value	length (null)
			field 1		field 2

response

Backend → Frontend:
command complete

byte1	int32	string
C	13	SELECT 2
command complete	message length	command tag

Number of rows retrieved

response

Backend → Frontend:
ready for query

byte1	int32	byte1
Z	5	I
ready for query	message length	status*

* — `I`: idle (not in a tx block), `T`: idle (in tx block), `E`: failed tx block

extended query

```
select * from my_table where id < $1
```

Frontend → Backend:

- parse
- describe (optional)
- bind
- execute
- sync

Backend → Frontend:

- row description
- data row for the 1st row
- data row for the 2nd row
- command complete
- ready for query

extended query

Frontend → Backend:
parse

byte1	int32	string	string	int16	int32
P	44	“”	select * from my_table where id < $1	1	23
parse	message length	name	query string	params	param OIDs

extended query

Frontend → Backend:
describe

byte1	int32	byte1	string
D	6	S	“”
describe	message length	S - statement P - portal*	name of the prepared statement/portal

* — i.e. cursor

extended query

Frontend → Backend:
flush

`byte1`	`int32`
`H`	`4`
flush	message length

extended query

Frontend → Backend:
bind

byte1	int32	string	string	
B	28	“”	“”	
bind	message length	destination portal	source prepared	…

extended query

Frontend → Backend:
bind

...

int16	int16	int16	int32	byteN
1	1	1	1	3
number of parameter format codes	format code	number of parameter values	length	value
	parameter 1		parameter 1	

...

extended query

Frontend → Backend:
bind

...

int16	int16	int16	int16
3	1	1	1
number of result-column format codes	format code	format code	format code
	field 1	field 2	field 3

extended query

Frontend → Backend:
execute

byte1	int32	string	int32
E	9	“”	0
execute	message length	name of the portal	maximum number of rows to return

extended query

Frontend → Backend:
sync

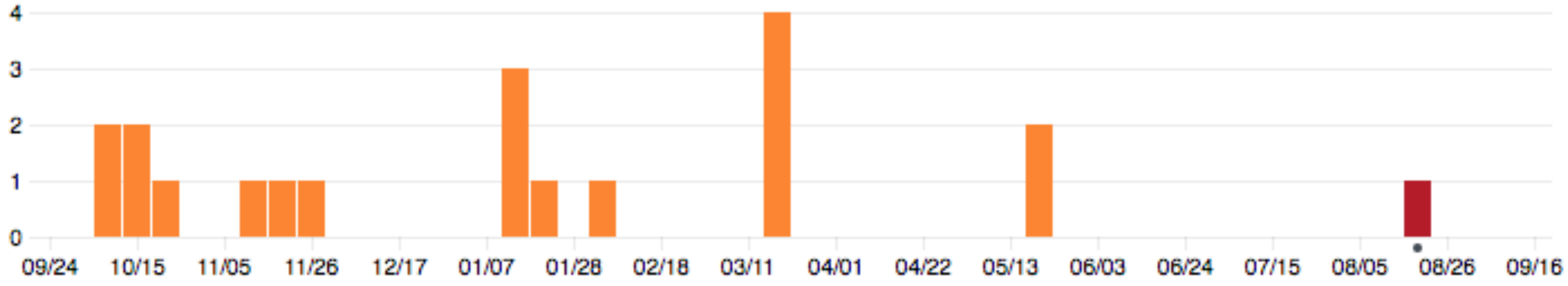
`byte1`	`int32`
`S`	`4`
sync	message length

pure golang libraries

- github.com/lib/pq
- github.com/go-pg/pg
- github.com/jackc/pgx
- ?

lib/pq

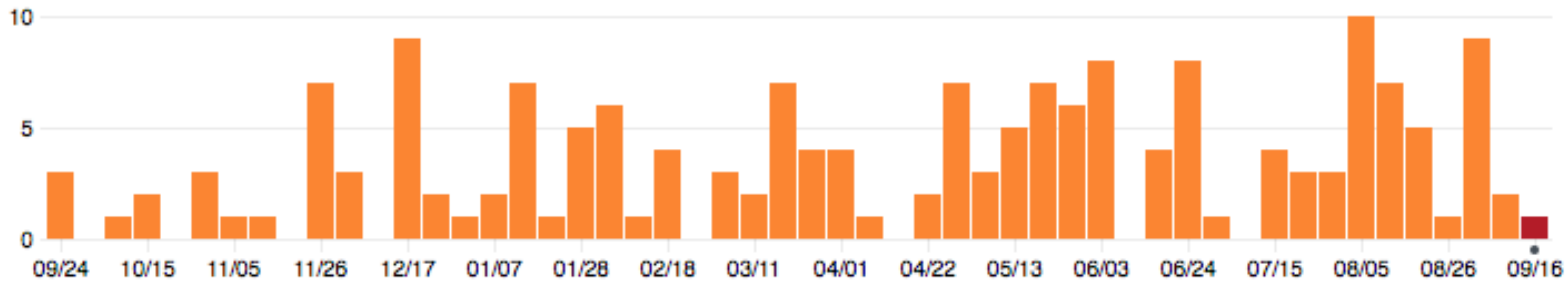
- Stars: 4,184
- Issues: 130
- Pull requests: 60



as on September 16, 2018

go-pg/pg

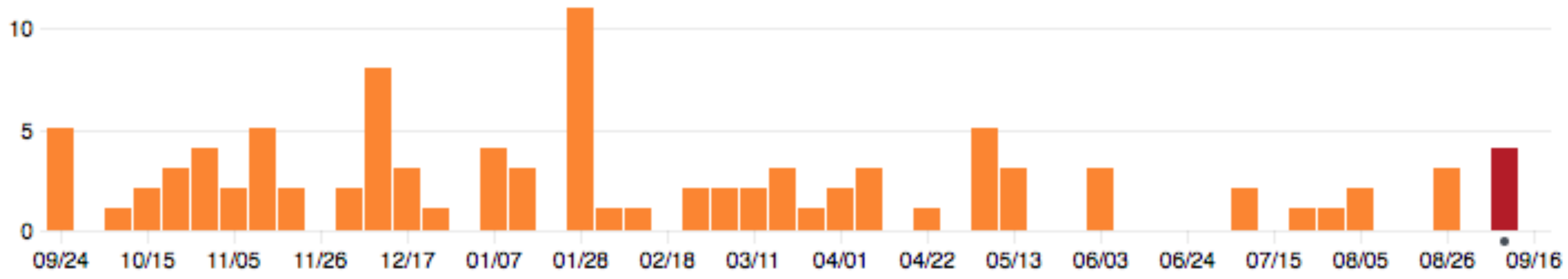
- Stars: 1,993
- Issues: 20
- Pull requests: 0



as on September 16, 2018

jackc/pgx

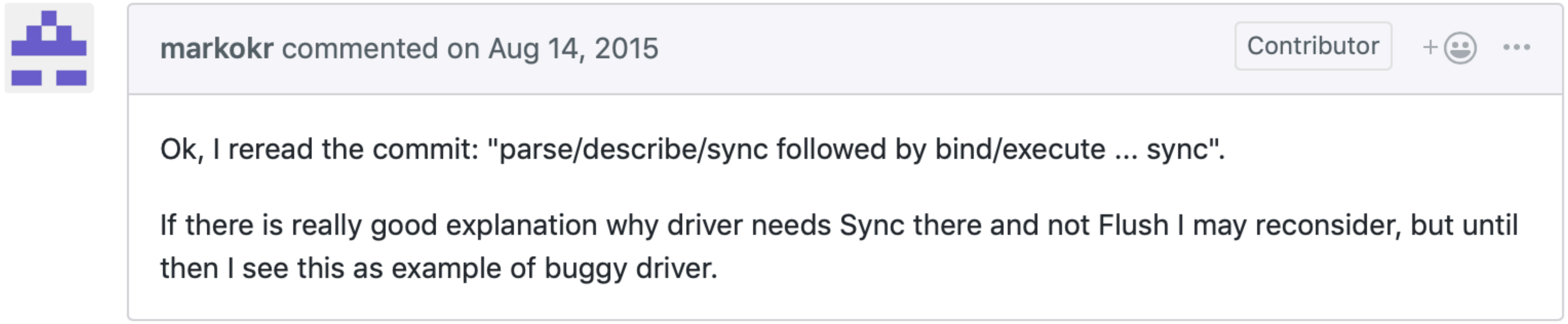
- Stars: 1,490
- Issues: 64
- Pull requests: 9



as on September 16, 2018

lib/pq

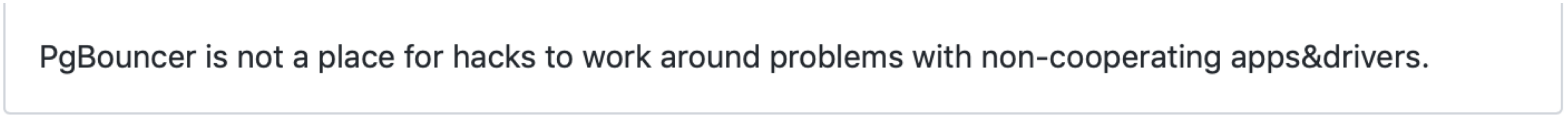
- Query execution:
  - Prepare unnamed statement;
  - Describe statement;
  - Sync;
  - Bind;
  - Execute;
  - Sync.


markokr commented on Aug 14, 2015

Ok, I reread the commit: "parse/describe/sync followed by bind/execute ... sync".

If there is really good explanation why driver needs Sync there and not Flush I may reconsider, but until then I see this as example of buggy driver.

...


PgBouncer is not a place for hacks to work around problems with non-cooperating apps&drivers.

lib/pq

- Query execution with `binary_parameters=yes`:
  - Prepare unnamed statement;
  - Bind;
  - Describe portal;
  - Execute;
  - Sync.

lib/pq

- `binary_parameters=yes:`
  - if parameter is []byte: sent in binary format, otherwise — text
  - all result column are in text format (unnamed prep stmnt)

- `binary_parameters=no:`
  - all parameters sent in text format
  - result columns are in text/binary format

go-pg/pg

- all queries sent via simple query cycle
- all parameters sent in text format
- all result columns are in text format

jackc/pgx

- `preferSimpleQuery` - forces using simple query cycle;
- with param OIDs specified:
  - Parse; Describe; Bind; Execute — Query method
  - Parse; Bind; Execute — Exec method

highlights

lib/pq highlights

- copyFrom (text format)
- `binary_parameters`
- returned ParameterStatuses are <u>not exposed</u>

go-pg/go highlights

- ORM
- count estimate (using EXPLAIN)
- copyTo, copyFrom (text format)
- returned ParameterStatuses are fetched but <u>ignored</u>

jackc/pgx highlights

- replication protocol support
- `PreferSimpleProtocol`
- copyTo, copyFrom (binary/text format)
- fetches type OIDs on connect
- returned ParameterStatuses are exposed

Thank you!
Questions?