Backend Size Matters

Serge Rielau
Principal Architect

srielau@salesforce.com

salesforce

Vital Stats

> 30,000 PLpgSQL functions

> 1000 tables

> 2000 PLpgSQL triggers
        ++ types, indices, views, ...

> 200 connections

No client session affinity to DB sessions across transactions (spraying)

The FUD

"The database is *leaking* memory like a sieve!"

"We *must* recycle connections every 100 txns!"

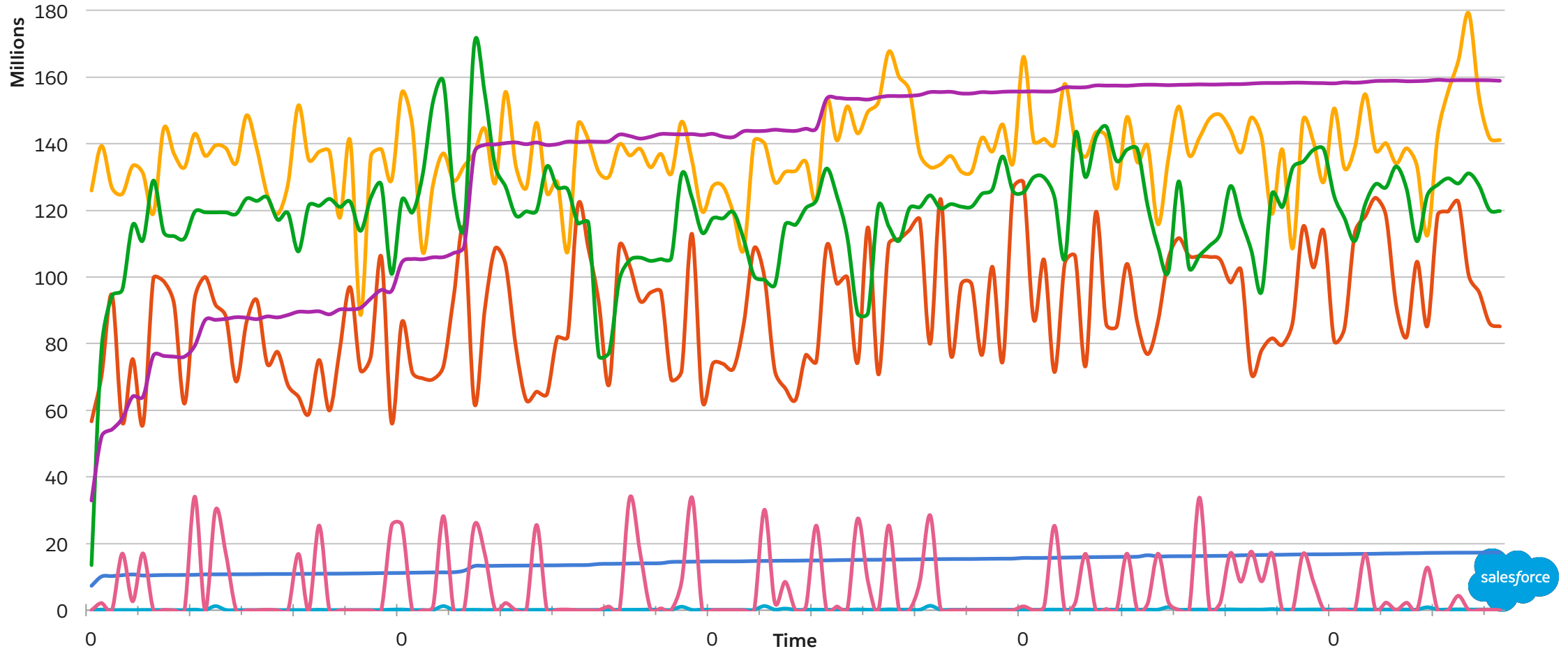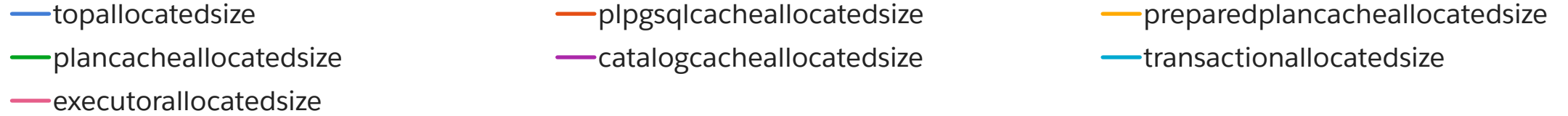"Recycling connections doesn't cost *anything*!"

The facts

> 30% time spent in compiling queries and PLpgSQL
(with frequent backend recycling)

Per backend memory foot print
(without recycling connections)
- 140MB Catalog cache
- 550MB PLpgSQL and embedded SQL cache
- 50MB "other"

→**750MB/backend * 200 backends => 150GB**

Allocated Memory

Good news!

It's just bloating.....

Limiting PLpgSQL function cache

- Set max number of cached function and LRU evict
- Reduced cache by half with no performance impact

Limiting catalog cache

- Set max number of cached catalog entries
- Cut size by 3 with no performance impact

More robust custom/generic plan decisions

- Rather than only compare cost also compare plan shape
  Switch to generic plan if shape (plan-id) remains unchanged

Plan source cache for dynamic SQL

- In conjunction with plan-id reduce compile time by 90%

salesforce

Memory estimator

	GUC	Setting	Measure	Multiplier	
Backend	function_cache_size	1000	Elements	272,000	272,000,000
	plan_source_cache_size	1000	Elements	127,000	127,000,000
	catalog_cache_size	10000	Elements	2,400	24,000,000
	Fixed				
	Other caches (relcache, ...)	36MB			37,748,736
	Other	30MB			31,457,280
				Total per Backend	469.40MB
App Settings	maxconnections	100	Connections	**Total Backends**	45.84GB

Food for thought

Due to spraying each backend holds mostly the same cached plans, …

Any improvement in footprint / backend yields incremental saving

Shared caches would drop footprint by 100x!

salesforce

Thank you!

srielau@salesforce.com