Locomotor
Transparent Migration of Client-Side Database Code

Michael Mior – University of Waterloo

Stored Procedures

- Complex database operations often require queries and updates to be mixed with application code
- Using stored procedures can reduce latency by cutting back on round trips

Stored Procedures

- Server-side scripting languages differ from the language used to develop the application
- Independent maintenance is required
- Required data needs to be carefully transferred in both directions

Locomotor

- Automatically convert annotated Python functions into server-side Lua scripts in Redis
- Patch the Python code at runtime to call the dynamically-generated script
- Updated code is automatically re-translated and deployed

Shipping Example

Redis Data Model

Key-value store where values can be complex types (e.g. maps, lists)

```
     item:1  → { name: 'Foo', category: 'Bar' }
category:Bar → ['item:1', ...]
```

Shipping Example

Application Code

```
hmset('item:' + str(key), {'name': 'Foo'})
lpush('category:Books', 'item:1')

ids = lrange('category:' + category, 0, -1)
items = []
for id in ids:
    items.append(hget(id, 'name'))
return items
```

Shipping Example

Application Code

```
hmset('item:' + str(key), {'name': name})
lpush('category:Books', 'item:1')

ids = lrange('category:' + category, 0, -1)
items = []
for id in ids:
    items.append(hget(id, 'name'))
return items
```

Each of these calls requires a round trip to the server!

And once per iteration

Shipping Example

Server Script

```
local category = 'category:' .. ARGV[1]
local item_keys = redis.call('lrange',
  'category:' .. category, 0, -1)
local items = {}
for _, key in ipairs(item_keys) do
  table.insert(items,
    redis.call('hget', key, 'name'))
end
return items
```

Shipping Example

Server Script

```
local category = 'category:' .. ARGV[1]
local item_keys = redis.call('lrange',
  'category:' .. category, 0, -1)
local items = {}
for _, key in ipairs(item_keys) do
  table.insert(items,
    redis.call('hget', key, 'name'))
end
return items
```

Only one round trip needed

Auto Translate

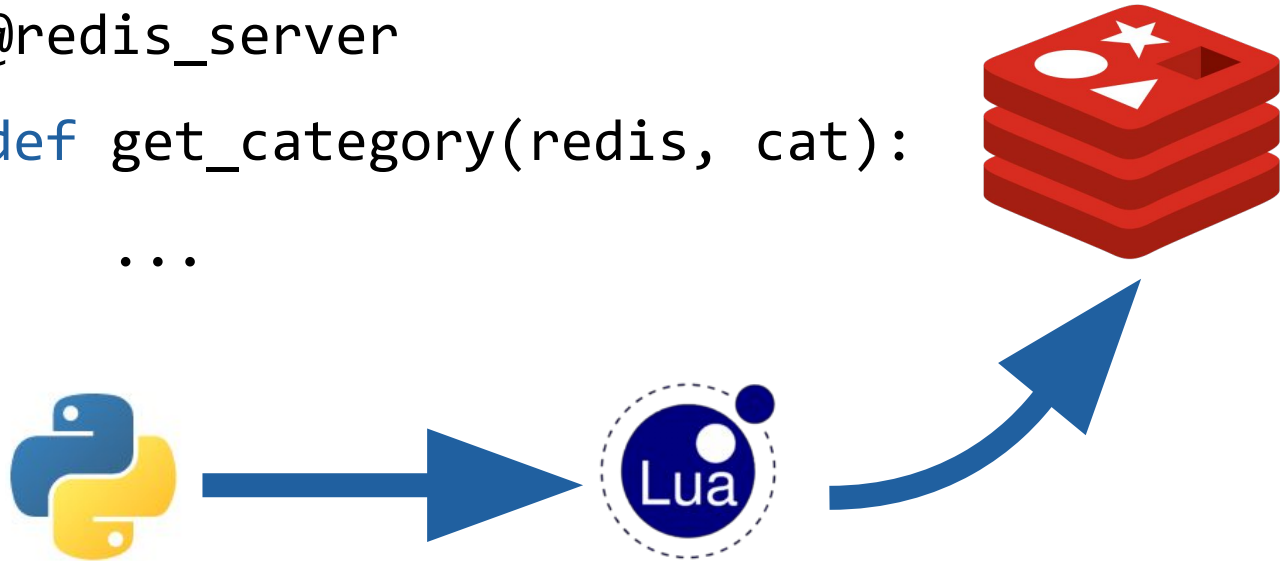
1. Data type conversions
2. Differing language semantics
3. Built-in functions
4. Loop constructs

...

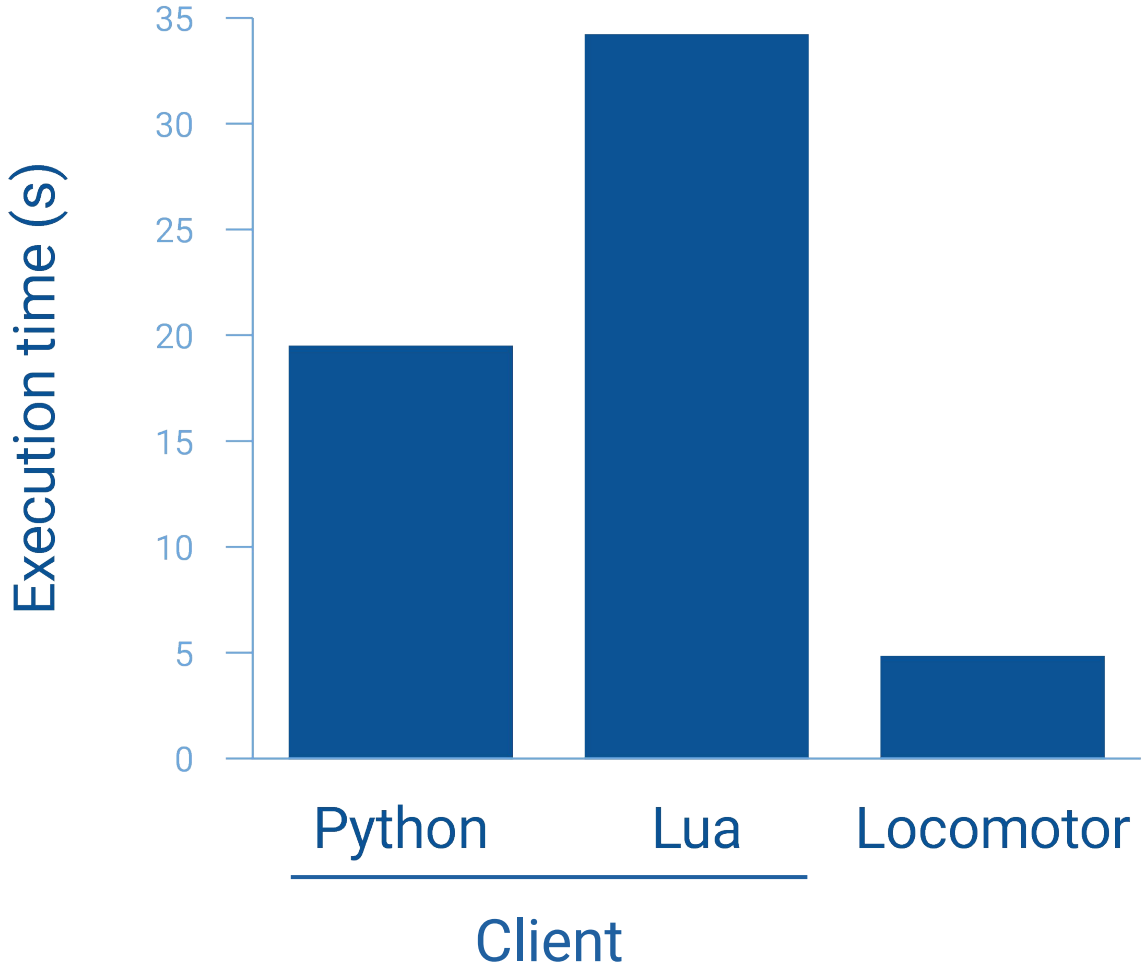
Deploy

Each time an annotated function is run, it is translated/shipped on the fly

```
@redis_server
def get_category(redis, cat):
    ...
```



Evaluation


Execution time (s)
35
30
25
20
15
10
5
0
Python
Lua
Locomotor
Client

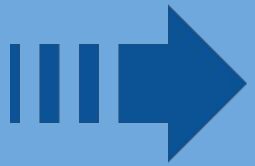
Evaluation

- Shipping code reduced round trips from 24 to 8
- With an inefficient server-side implementation, we still achieve a 4× reduction in runtime

Summary

- Translation of client code to server-side scripting languages is a viable approach to optimization
- This optimization can be automated with careful observation of language semantics

Future Work



- Explore other DB/language combos such as MongoDB and JavaScript
- Automated selection of code fragments to translate and move
- Use of low-level interfaces (e.g. Redis modules)

Questions?