

Monolith vs. Microservices: CPU, Latency & Throughput Analysis

Jotinha

1. Pacotes e utilitários

```
library(dplyr)
library(tidyr)
library(ggplot2)
library(readr)
library(here)
library(lubridate)
```

2. Carregamento dos dados

```
# ajuste o path dos results
df_cpu <- read_csv(here("merged_all.csv"))
df_locust <- read_csv(here("merged_locust.csv"))

# nomes fixos das colunas
cpu_metric_col <- "cores"      # ajuste para o nome correto no merged_all
latency_col    <- "total_median_response_time" # ajuste para o nome correto no merge_locust
throughput_col <- "req_s"
```

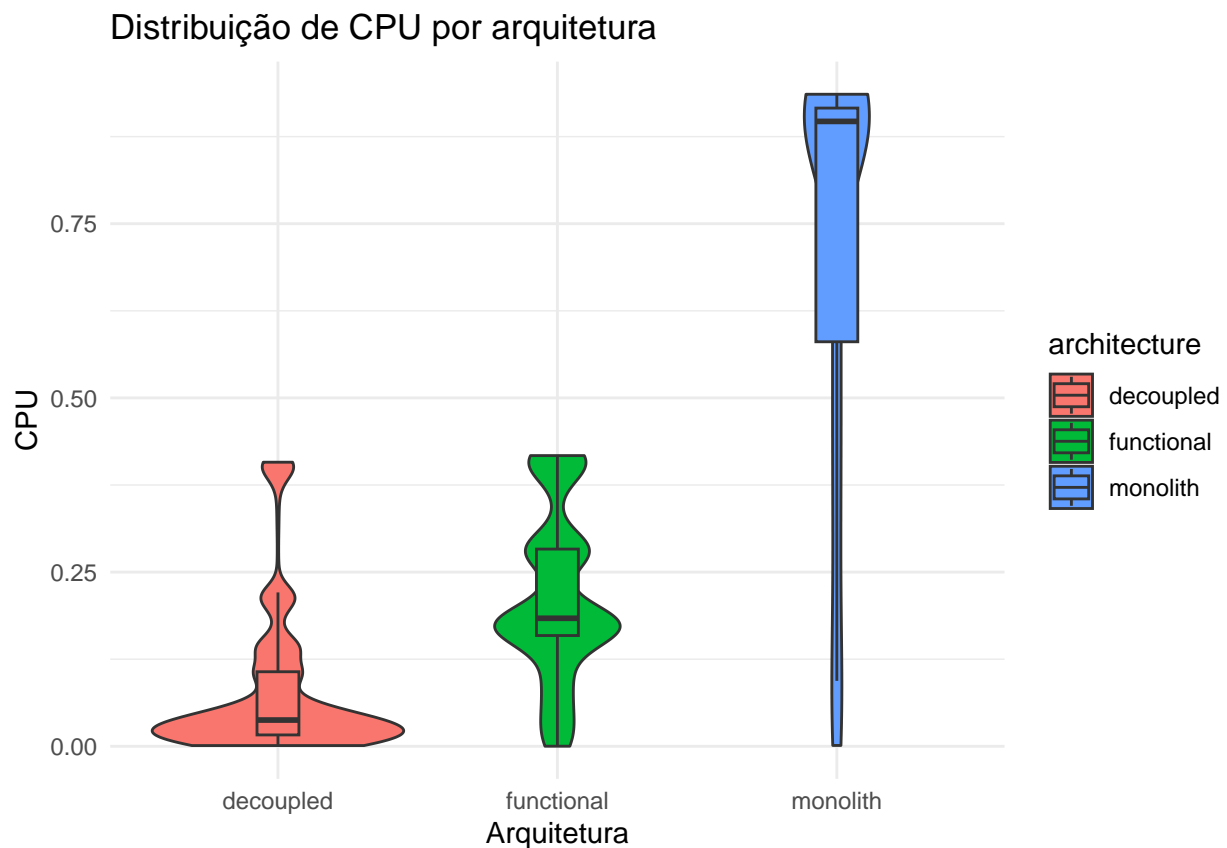
3. CPU

3.1 Entre arquiteturas

```
df_cpu %>%
  group_by(architecture) %>%
  summarise(
    cpu_mean    = mean(.data[[cpu_metric_col]], na.rm = TRUE),
    cpu_median  = median(.data[[cpu_metric_col]], na.rm = TRUE),
    cpu_p95     = quantile(.data[[cpu_metric_col]], 0.95, na.rm = TRUE)
  )
```

```
## # A tibble: 3 x 4
##   architecture cpu_mean cpu_median cpu_p95
##   <chr>         <dbl>     <dbl>   <dbl>
## 1 decoupled     0.0816     0.0376  0.393
## 2 functional    0.213      0.184   0.414
## 3 monolith      0.721      0.897   0.930
```

```
df_cpu %>%
  ggplot(aes(x = architecture, y = .data[[cpu_metric_col]], fill = architecture)) +
  geom_violin(trim = TRUE) +
  geom_boxplot(width = 0.15, outlier.shape = NA) +
  labs(title = "Distribuição de CPU por arquitetura",
       x = "Arquitetura", y = "CPU") +
  theme_minimal()
```



3.2 Entre serviços (somente microarquitecturas)

```
df_cpu %>%
  filter(architecture %in% c("decoupled", "functional")) %>%
  group_by(architecture, service) %>%
  summarise(
```

```

    cpu_mean = mean(.data[[cpu_metric_col]], na.rm = TRUE),
    .groups = "drop"
) %>%
arrange(architecture, desc(cpu_mean))

```

```

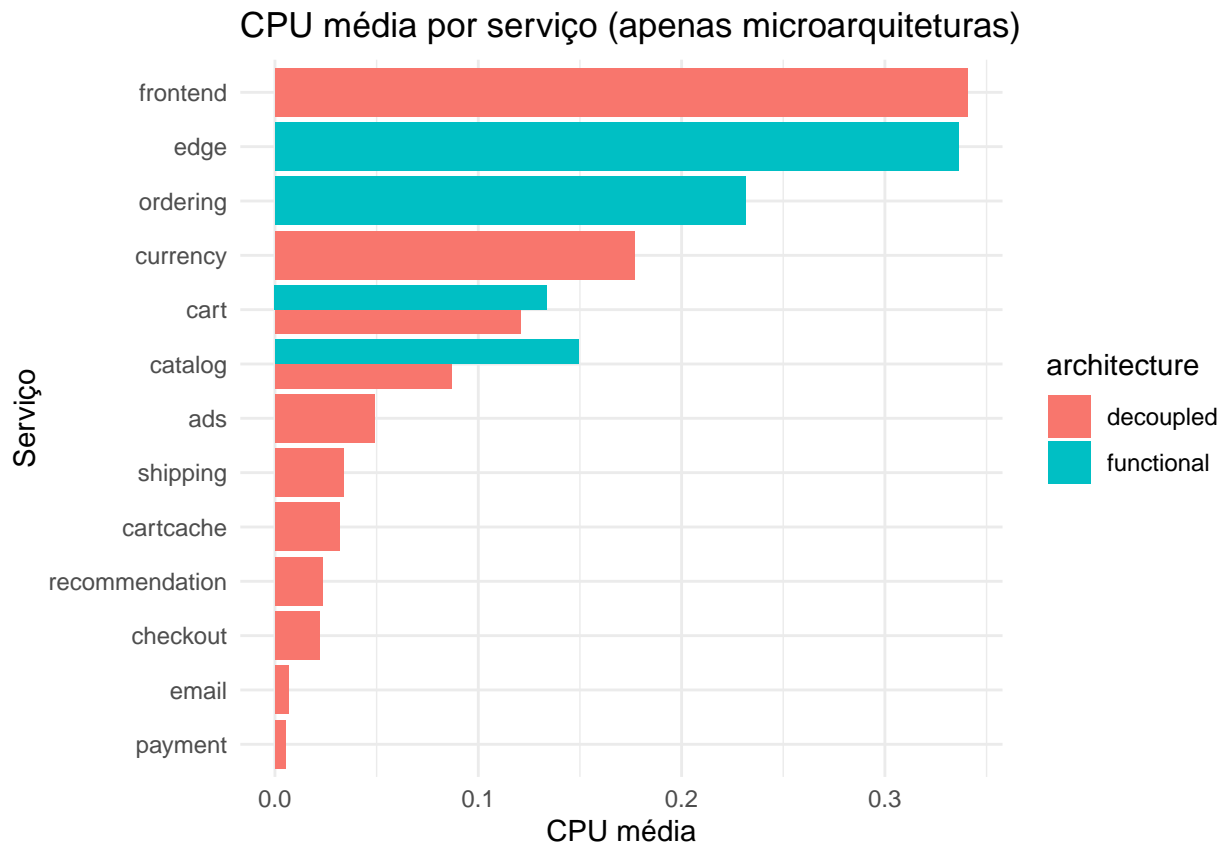
## # A tibble: 15 x 3
##   architecture service      cpu_mean
##   <chr>          <chr>      <dbl>
## 1 decoupled     frontend    0.341
## 2 decoupled     currency    0.177
## 3 decoupled     cart        0.121
## 4 decoupled     catalog     0.0869
## 5 decoupled     ads         0.0491
## 6 decoupled     shipping    0.0338
## 7 decoupled     cartcache   0.0319
## 8 decoupled     recommendation 0.0236
## 9 decoupled     checkout    0.0218
## 10 decoupled    email       0.00676
## 11 decoupled    payment     0.00508
## 12 functional   edge        0.336
## 13 functional   ordering     0.231
## 14 functional   catalog     0.149
## 15 functional   cart        0.134

```

```

df_cpu %>%
  filter(architecture %in% c("decoupled", "functional")) %>%
  group_by(architecture, service) %>%
  summarise(cpu_mean = mean(.data[[cpu_metric_col]], na.rm = TRUE), .groups = "drop") %>%
  ggplot(aes(x = reorder(service, cpu_mean), y = cpu_mean, fill = architecture)) +
  geom_col(position = "dodge") +
  coord_flip() +
  labs(title = "CPU média por serviço (apenas microarquiteturas)",
       x = "Serviço", y = "CPU média") +
  theme_minimal()

```



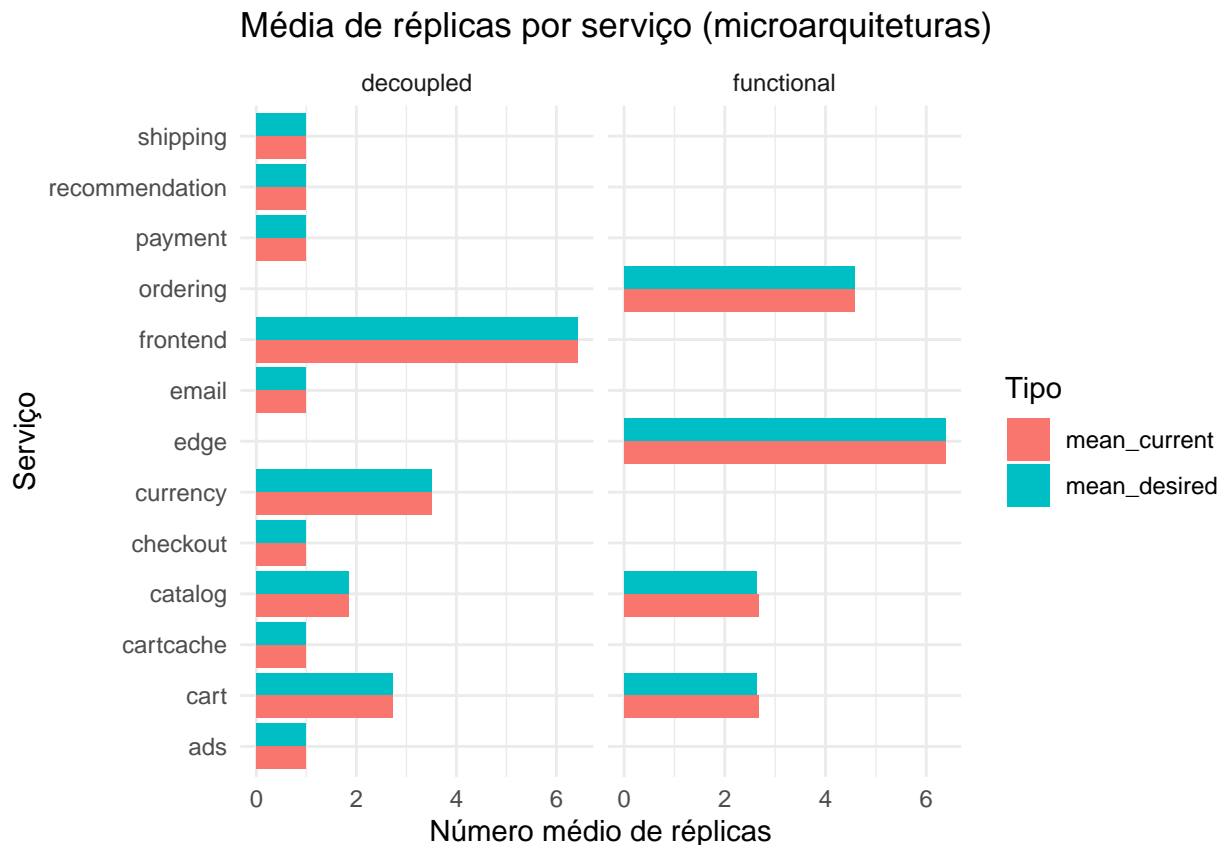
3.3 Réplicas — comparação entre serviços (microarquitecturas)

```
df_cpu %>%
  filter(architecture %in% c("decoupled", "functional")) %>%
  group_by(architecture, service) %>%
  summarise(
    max_repl      = max(max_replicas, na.rm = TRUE),
    mean_desired  = mean(desired_replicas, na.rm = TRUE),
    mean_current  = mean(current_replicas, na.rm = TRUE),
    .groups = "drop"
  ) %>%
  arrange(architecture, desc(mean_current))
```

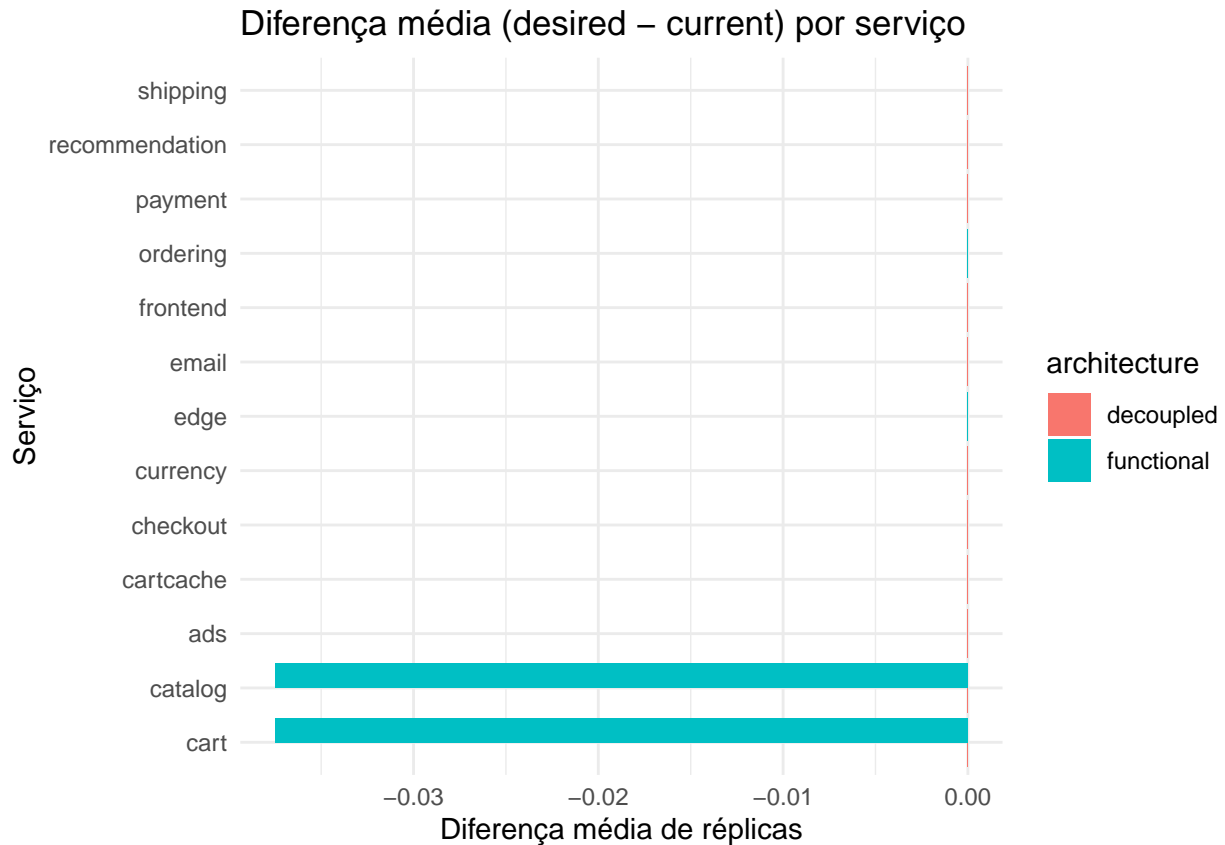
```
## # A tibble: 15 x 5
##   architecture service      max_repl mean_desired mean_current
##   <chr>          <chr>      <dbl>      <dbl>      <dbl>
## 1 decoupled    frontend      10        6.42        6.42
## 2 decoupled    currency      10        3.52        3.52
## 3 decoupled    cart          10        2.73        2.73
## 4 decoupled    catalog       10        1.85        1.85
```

## 5 decoupled	ads	10	1	1
## 6 decoupled	cartcache	10	1	1
## 7 decoupled	checkout	10	1	1
## 8 decoupled	email	10	1	1
## 9 decoupled	payment	10	1	1
## 10 decoupled	recommendation	10	1	1
## 11 decoupled	shipping	10	1	1
## 12 functional	edge	10	6.38	6.38
## 13 functional	ordering	10	4.58	4.58
## 14 functional	cart	10	2.64	2.68
## 15 functional	catalog	10	2.64	2.68

```
df_cpu %>%
  filter(architecture %in% c("decoupled", "functional")) %>%
  group_by(architecture, service) %>%
  summarise(
    mean_desired = mean(desired_replicas, na.rm = TRUE),
    mean_current = mean(current_replicas, na.rm = TRUE),
    .groups = "drop"
  ) %>%
  pivot_longer(cols = c(mean_desired, mean_current),
               names_to = "replica_type", values_to = "replicas") %>%
  ggplot(aes(x = service, y = replicas, fill = replica_type)) +
  geom_col(position = "dodge") +
  facet_wrap(~ architecture, scales = "free_x") +
  coord_flip() +
  labs(title = "Média de réplicas por serviço (microarquiteturas)",
       x = "Serviço", y = "Número médio de réplicas",
       fill = "Tipo") +
  theme_minimal()
```



```
df_cpu %>%
  filter(architecture %in% c("decoupled", "functional")) %>%
  mutate(diff_desired_current = desired_replicas - current_replicas) %>%
  group_by(architecture, service) %>%
  summarise(
    mean_diff = mean(diff_desired_current, na.rm = TRUE),
    .groups = "drop"
  ) %>%
  ggplot(aes(x = reorder(service, mean_diff), y = mean_diff, fill = architecture)) +
  geom_col(position = "dodge") +
  coord_flip() +
  labs(title = "Diferença média (desired - current) por serviço",
       x = "Serviço", y = "Diferença média de réplicas") +
  theme_minimal()
```



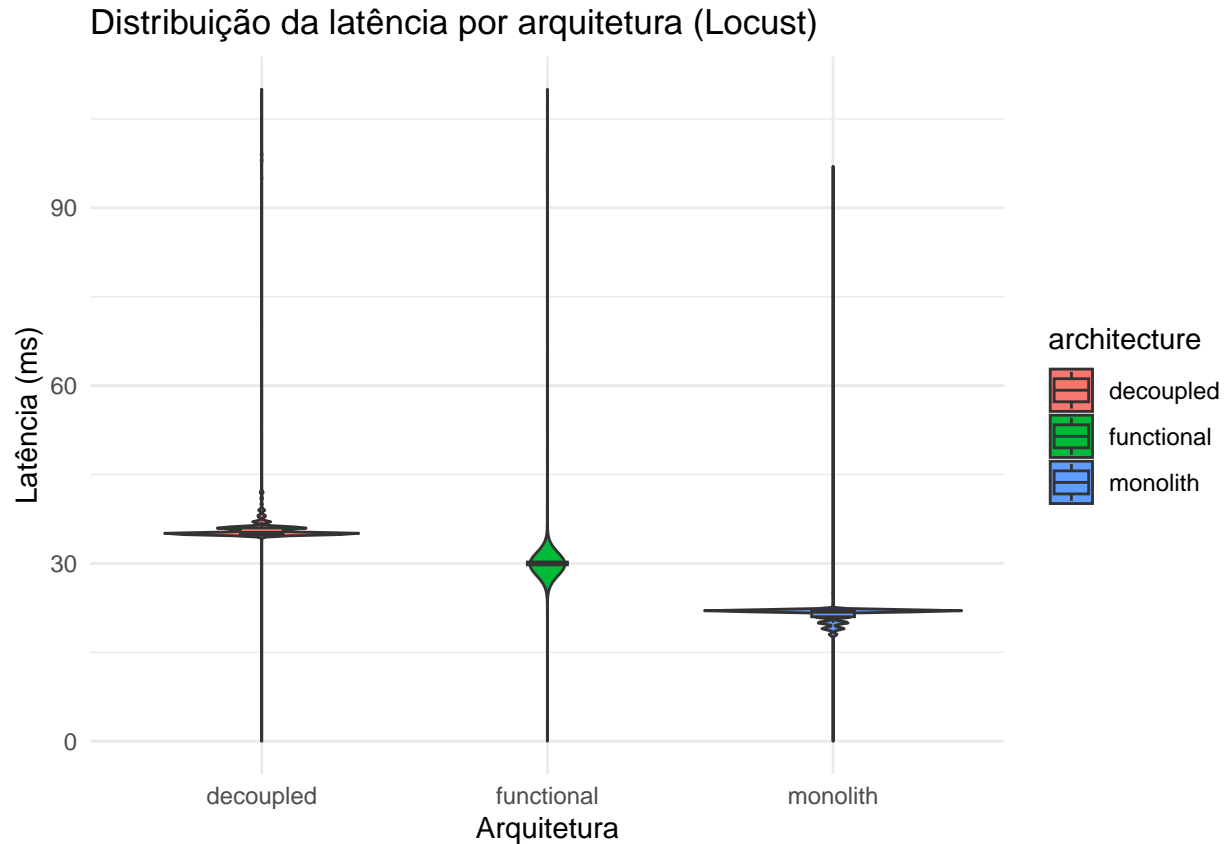
4. Latência (Locust)

```
df_locust %>%
  group_by(architecture) %>%
  summarise(
    latency_mean = mean(.data[[latency_col]], na.rm = TRUE),
    latency_median = median(.data[[latency_col]], na.rm = TRUE),
    latency_p95 = quantile(.data[[latency_col]], 0.95, na.rm = TRUE)
  )
```

```
## # A tibble: 3 x 4
##   architecture latency_mean latency_median latency_p95
##   <chr>          <dbl>          <dbl>          <dbl>
## 1 decoupled      36.6            35            39
## 2 functional     31.0            30            30
## 3 monolith       22.0            22            22
```

```
df_locust %>%
  ggplot(aes(x = architecture, y = .data[[latency_col]], fill = architecture)) +
  geom_violin(trim = TRUE) +
```

```
geom_boxplot(width = 0.15, outlier.shape = NA) +
labs(title = "Distribuição da latência por arquitetura (Locust)",
     x = "Arquitetura", y = "Latência (ms)") +
theme_minimal()
```



5. Throughput (Locust)

```
df_locust %>%
  group_by(architecture) %>%
  summarise(
    thr_mean = mean(.data[[throughput_col]], na.rm = TRUE),
    # thr_median = median(.data[[throughput_col]], na.rm = TRUE),
    thr_p95 = quantile(.data[[throughput_col]], 0.95, na.rm = TRUE)
  )
```

```
## # A tibble: 3 x 3
##   architecture thr_mean thr_p95
##   <chr>         <dbl>   <dbl>
## 1 decoupled     59.3    61.5
## 2 functional    59.6    61.7
## 3 monolith      59.9    61.9
```



```
df_locust %>%
  ggplot(aes(x = architecture, y = .data[[throughput_col]], fill = architecture)) +
  geom_violin(trim = TRUE) +
  geom_boxplot(width = 0.15, outlier.shape = NA) +
  labs(title = "Distribuição do throughput por arquitetura (Locust)",
       x = "Arquitetura", y = "Requests/s") +
  theme_minimal()
```

