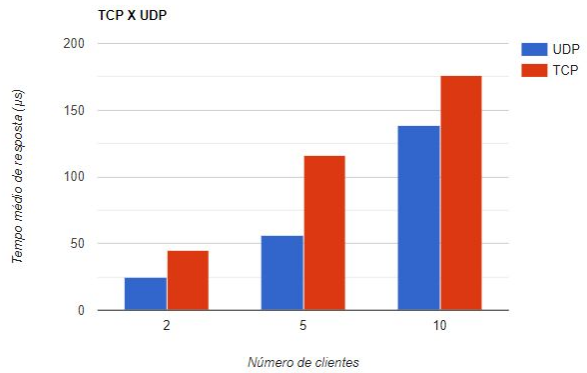
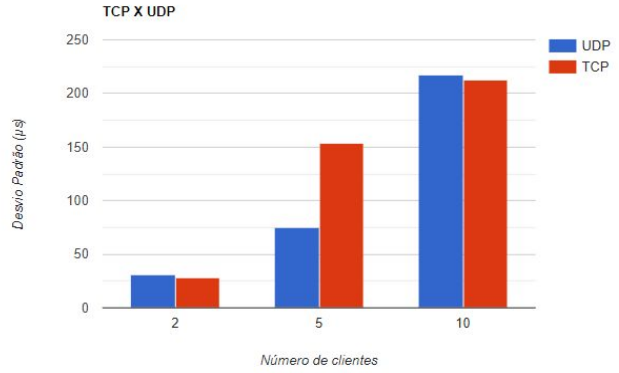
Victor Nicéas & Lucas Mota

middleware - lista 3

Índice

Resultados Anteriores





1. Códigos importantes e proto

Proto

Mecanismo neutro para serializar dados estruturados. Um xml melhorado.

```
syntax = "proto3";

package calculadora;

service Calculator {
  rpc Add (Request) returns (Reply) {}
  rpc Sub (Request) returns (Reply) {}
  rpc Div (Request) returns (Reply) {}
  rpc Mul (Request) returns (Reply) {}
}

// Mensagem de Request
message Request {
  string Op = 1;
  int32 P1 = 2;
  int32 P2 = 3;
}

//Mensagem de resposta
message Reply {
  int32 N = 1;
}
```

Cliente & Servidor

```go
// contacta o server
ctx, cancel := context.WithTimeout(context.Background(), time.Second*5)
defer cancel()

for idx = 0; idx < shared.SAMPLE_SIZE; idx++ {

    t1 := time.Now()
    // invoca operação remota
    rep, msgErr := calc.Add(ctx, &calculadora.Request{Op: "add", P1: idx, P2: idx})

    if msgErr != nil {
        fmt.Println(idx, msgErr)
    } else {
        x := float64(time.Since(t1))

        fmt.Printf("%f %d\n", x, rep.N)
    }
}
```

```go
func (s *servidorCalculadora) Add(ctx context.Context, in *calculadora.Request) (*calculadora.Reply, error) {
    return &calculadora.Reply{N: in.P1 + in.P2}, nil
}
```

```go
conn, err := net.Listen("tcp", ":"+strconv.Itoa(shared.CALCULATOR_PORT))
shared.ChecaErro(err, "Não foi possível criar o listener")

servidor := grpc.NewServer()
calculadora.RegisterCalculatorServer(servidor, &servidorCalculadora{})

fmt.Println("Servidor pronto ...")

// Register reflection service on gRPC server.
reflection.Register(servidor)

err = servidor.Serve(conn)
shared.ChecaErro(err, "Falha ao servir")
```

2. Um novo cliente: ghz.

```
./ghz --insecure --proto ./greeter.proto --call helloworld.Greeter.SayHello -d '{"name":"Joe"}' 0.0.0.0:50051

Summary:
  Count:    200
  Total:    235.93 ms
  Slowest:  85.68 ms
  Fastest:  25.65 ms
  Average:  39.85 ms
  Requests/sec: 847.70

Response time histogram:
  25.652 [1]    |▮
  31.655 [43]   |▮▮▮▮▮▮▮▮▮▮▮▮▮▮▮▮▮▮▮▮▮▮▮▮▮
  37.657 [37]   |▮▮▮▮▮▮▮▮▮▮▮▮▮▮▮▮▮▮▮▮▮▮▮
  43.660 [66]   |▮▮▮▮▮▮▮▮▮▮▮▮▮▮▮▮▮▮▮▮▮▮▮▮▮▮▮▮▮▮▮▮▮▮▮▮▮▮▮▮▮▮▮▮▮▮▮
  49.662 [24]   |▮▮▮▮▮▮▮▮▮▮▮▮▮▮▮
  55.665 [15]   |▮▮▮▮▮▮▮▮▮
  61.668 [3]    |▮▮
  67.670 [2]    |▮
  73.673 [3]    |▮▮
  79.676 [2]    |▮
  85.678 [2]    |▮

Latency distribution:
  10% in 29.12 ms
  25% in 32.33 ms
  50% in 39.42 ms
  75% in 44.11 ms
  90% in 53.81 ms
  95% in 64.59 ms
  99% in 85.68 ms

Status code distribution:
  [OK]                198 responses
  [PermissionDenied]  1 responses
  [Internal]          1 responses

Error distribution:
  [1]   rpc error: code = Internal desc = Internal error.
  [1]   rpc error: code = PermissionDenied desc = Permission denied.
```

Ferramenta de benchmarking open-source para grpc.
Fornece opções bastante úteis como -c (numero de concorrencia) e -n (numero de requests no total)

https://github.com/bojand/ghz

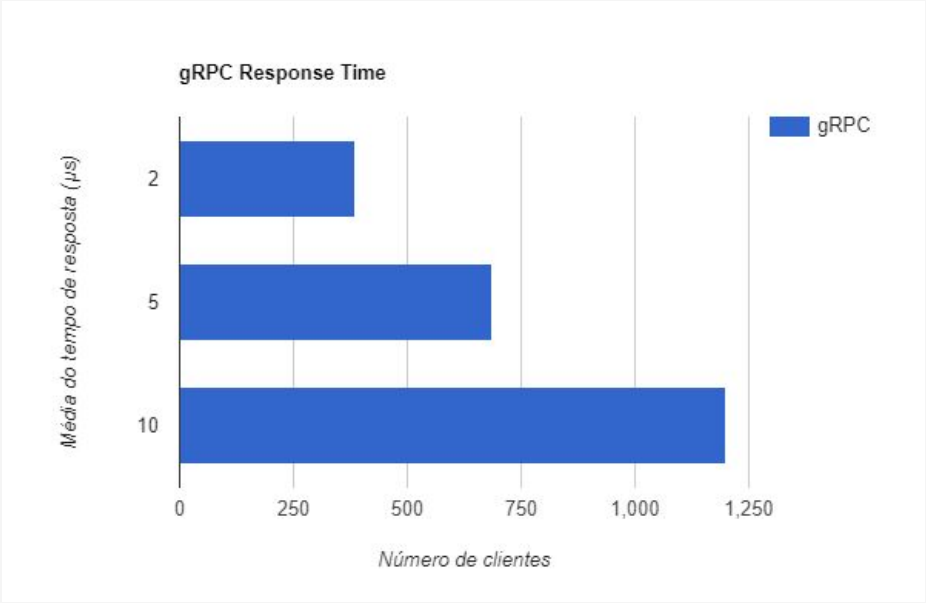
The total number of requests to run. Default is 200. The combination of -c and -n are critical in how the benchmarking is done. ghz takes the -c argument and spawns that many worker goroutines. In parallel these goroutines each do their share (n / c) requests. So for example with the default -c 50 -n 200 options we would spawn 50 goroutines which in parallel each do 4 requests.

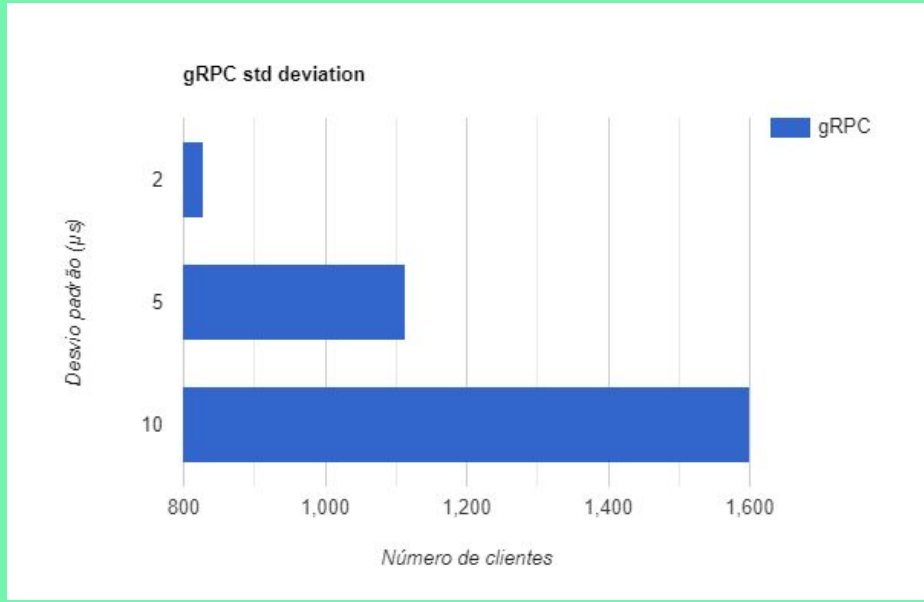
Resultado Final.

Cliente implementado

rodamos o cliente implementado por nós & professor

Tempo de resposta & desvio padrão





ghz benchmarking

```
vdcn@desktopVictor:/mnt/c/Go/src/calculadora/grpc/calculadora$ ghz --insecure --proto ./calculadora.proto --call calculadora.Calculator.Add -
d '{"Op": "add","P1":300, "P2":300}' -n 20000 -c 2 192.168.15.51:3300

Summary:
  Count:        20000
  Total:        3.32 s
  Slowest:      34.10 ms
  Fastest:      0.11 ms
  Average:      0.28 ms
  Requests/sec: 6030.01

Response time histogram:
  0.108 [1]     |
  3.508 [19993] |∎∎∎∎∎∎∎∎∎∎∎∎∎∎∎∎∎∎∎∎∎∎∎∎∎∎∎∎∎∎∎∎∎∎∎∎∎∎∎∎
  6.908 [4]     |
  10.307 [0]    |
  13.707 [0]    |
  17.107 [0]    |
  20.506 [0]    |
  23.906 [0]    |
  27.305 [0]    |
  30.705 [0]    |
  34.105 [2]    |

Latency distribution:
  10 % in 0.19 ms
  25 % in 0.22 ms
  50 % in 0.26 ms
  75 % in 0.31 ms
  90 % in 0.35 ms
  95 % in 0.38 ms
  99 % in 0.53 ms

Status code distribution:
  [OK]   20000 responses
```

```
vdcn@desktopVictor:/mnt/c/Go/src/calculadora/grpc/calculadora$ ghz --insecure --proto ./calculadora.proto --call calculadora.Calculator.Add -
d '{"Op": "add","P1":300, "P2":300}' -n 50000 -c 5 192.168.15.51:3300

Summary:
  Count:        50000
  Total:        3.98 s
  Slowest:      6.86 ms
  Fastest:      0.11 ms
  Average:      0.34 ms
  Requests/sec: 12548.99

Response time histogram:
  0.106 [1]     |
  0.781 [49751] |∎∎∎∎∎∎∎∎∎∎∎∎∎∎∎∎∎∎∎∎∎∎∎∎∎∎∎∎∎∎∎∎∎∎∎∎∎∎∎∎
  1.456 [222]   |
  2.131 [9]     |
  2.806 [2]     |
  3.480 [4]     |
  4.155 [0]     |
  4.830 [0]     |
  5.505 [5]     |
  6.180 [2]     |
  6.855 [4]     |

Latency distribution:
  10 % in 0.25 ms
  25 % in 0.28 ms
  50 % in 0.33 ms
  75 % in 0.38 ms
  90 % in 0.44 ms
  95 % in 0.49 ms
  99 % in 0.69 ms

Status code distribution:
  [OK]   50000 responses


vdcn@desktopVictor:/mnt/c/Go/src/calculadora/grpc/calculadora$
```

```
vdcn@desktopVictor:/mnt/c/Go/src/calculadora/grpc/calculadora$ ghz --insecure --proto ./calculadora.proto --call calculadora.Calculator.Add -
d '{"Op": "add","P1":300, "P2":300}' -n 100000 -c 10 192.168.15.51:3300

Summary:
  Count:        100000
  Total:        4.98 s
  Slowest:      7.89 ms
  Fastest:      0.13 ms
  Average:      0.45 ms
  Requests/sec: 20082.25

Response time histogram:
  0.132 [1]     |
  0.908 [98818] |∎∎∎∎∎∎∎∎∎∎∎∎∎∎∎∎∎∎∎∎∎∎∎∎∎∎∎∎∎∎∎∎∎∎∎∎∎∎∎∎
  1.683 [1104]  |
  2.459 [44]    |
  3.234 [5]     |
  4.010 [2]     |
  4.785 [2]     |
  5.561 [17]    |
  6.336 [5]     |
  7.112 [1]     |
  7.888 [1]     |

Latency distribution:
  10 % in 0.31 ms
  25 % in 0.37 ms
  50 % in 0.43 ms
  75 % in 0.50 ms
  90 % in 0.58 ms
  95 % in 0.66 ms
  99 % in 0.94 ms

Status code distribution:
  [OK]   100000 responses


vdcn@desktopVictor:/mnt/c/Go/src/calculadora/grpc/calculadora$
```

Obrigado!