

# Relatório 1: Benchmark EP Em Golang

Kevin S. Pereira, Thomazio Giacobbe

<https://github.com/thomaziogiacobbe/NPB-Golang>

# Benchmark EP em Golang

- Baseado na implementação em C++ de Dalvan Griebler et al.
  - NPB-OMP; implementação em OpenMP
  - <https://github.com/GMAP/NPB-CPP>
- Adaptações utilizando features de multithreading do Golang
  - goroutines (criação de tarefas)
  - channels (passagem de mensagem)
- Outras adaptações da linguagem
  - time (biblioteca para lidar com tempo)

# Bibliotecas

- math
  - Funções matemáticas
- runtime
  - Informações da CPU
- strconv
  - Conversões entre strings e outros tipos
- go-pretty (<https://github.com/jedib0t/go-pretty>)
  - Biblioteca third party para formatação dos resultados em interface

# Bloco paralelo

- Função separada da main, parallelEP
- Variáveis compartilhadas declaradas no escopo da função parallelEP
- Variáveis privadas declaradas no escopo da função anônima

```
func parallelEP(
    np int,
    an float64,
    sx *float64,
    sy *float64,
    q []float64,
    tt *time.Duration,
) {
    const k_offset = -1

    qTemp := [NQ]float64{}

    var result ResultData

    resultChn := make(chan ResultData, np)

    start := time.Now()
    defer getExecTime(tt, &start)
    for k := 1; k <= np; k++ {
        go func(k int) {
            var (
                sxThis, syThis      = 0.0, 0.0
                t1, t2, t3, t4, x1, x2 float64
                kk, ik, l            int
                qq, x                = [NQ]float64{}, [NK_PLUS]float64{}
                resultThis          ResultData
            )
            // ...
        }
    }
}
```

# Bloco paralelo: Go

- Comando go instanciando np tarefas da função anônima
- Diferente da implementação original que usa a cláusula reduction no laço, as goroutines processam pelas variáveis privadas e comunicam seus resultados pelo channel

```
for k := 1; k <= np; k++ {
    go func(k int) {
        var (
            sxThis, syThis      = 0.0, 0.0
            t1, t2, t3, t4, x1, x2 float64
            kk, ik, l           int
            qq, x               = [NQ]float64{}, [NK_PLUS]float64{}
            resultThis          ResultData
        )

        kk = k_offset + k
        t1 = S
        t2 = an

        for i := 0; i <= 100; i++ {
            ik = kk / 2
            if (2 * ik) != kk {
                t3 = npb.Randlc(&t1, t2)
            }
            if ik == 0 {
                break
            }
            t3 = npb.Randlc(&t2, t2)
            kk = ik
        }

        npb.Vranlc(2*NK, &t1, A, x[:])

        // ...
    }
}
```

## Bloco paralelo: Channels

- Declarado como variável compartilhada o channel para comunicação com buffer de tamanho np
- As np threads calculam seus resultados, salvam na struct, enviam para o channel e terminam

```
// ...  
  
for i := 0; i < NK; i++ {  
    x1 = 2.0*x[2*i] - 1.0  
    x2 = 2.0*x[2*i+1] - 1.0  
    t1 = math.Pow(x1, 2) + math.Pow(x2, 2)  
    if t1 <= 1.0 {  
        t2 = math.Sqrt(-2.0 * math.Log(t1) / t1)  
        t3 = x1 * t2  
        t4 = x2 * t2  
        l = int(math.Max(math.Abs(t3), math.Abs(t4)))  
        qq[l] += 1.0  
        sxThis += t3  
        syThis += t4  
    }  
}  
resultThis.sxResult = sxThis  
resultThis.syResult = syThis  
resultThis.qResult = qq  
resultChn <- resultThis  
}(k)  
}
```

## Bloco paralelo: Channels

- A thread principal na função parallelEP bloqueia aguardando resultado do channel. Quando recebe, ele lê da struct e acumula o resultado em suas variáveis

```
for k := 1; k <= np; k++ {  
    result = <-resultChn  
    *sx += result.sxResult  
    *sy += result.syResult  
    qTemp = result.qResult  
    for j := range q {  
        q[j] += qTemp[j]  
    }  
}
```

## Bloco paralelo: Observações

- Optamos por channel e não mutex para evitar bloqueios e sincronismos entre as threads
- O buffer do channel é importante para evitar bloqueios parecidos com mutex
- Explorar comunicação assíncrona

```
// ...  
  
for i := 0; i < NK; i++ {  
    x1 = 2.0*x[2*i] - 1.0  
    x2 = 2.0*x[2*i+1] - 1.0  
    t1 = math.Pow(x1, 2) + math.Pow(x2, 2)  
    if t1 <= 1.0 {  
        t2 = math.Sqrt(-2.0 * math.Log(t1) / t1)  
        t3 = x1 * t2  
        t4 = x2 * t2  
        l = int(math.Max(math.Abs(t3), math.Abs(t4)))  
        qq[l] += 1.0  
        sxThis += t3  
        syThis += t4  
    }  
}  
reductionMutex.Lock()  
*sx += sxThis  
*sy += syThis  
reductionMutex.Unlock()  
arrayResultMutex.Lock()  
for i := 0; i < NQ; i++ {  
    q[i] += qq[i]  
}  
arrayResultMutex.Unlock()  
}(k)  
}
```



# Resultados

- Mesmos resultados numéricos da implementação em C++ de Dalvan Griebler et al.
- Resultados diferentes da implementação original em Fortran (RNR Technical Report RNR-94-007)
- Desempenho foi aproximadamente 2 vezes pior do que a implementação em C++ e 6 vezes pior do que a implementação em Fortran
- Os benchmarks foram executados em um Intel i7 7700, que possui 4 núcleos físicos e 8 threads

$$Mop/s = \frac{2^{M+1} \cdot 1 \times 10^6}{t}$$

# Resultados: S

```
Run: go build run x
NAS Parallel Benchmarks 4.1 Parallel Golang version - EP Benchmark

Number of random numbers generated: 3.3554432e+07
```

EP BENCHMARK RESULTS	
CPU Time	417.528852ms
N	2^24
No. Gaussian Pairs	1.3176389e+07
Sums	-3247.8346520346136 -6958.407078382822
Counts	
0	6.140517e+06
1	5.8653e+06
2	1.100361e+06
3	68546
4	1648
5	17
6	0
7	0
8	0

  

EP BENCHMARK COMPLETED	
Class	S
Size	33554432
Number of available threads	8
Number of iterations	0
Time in seconds	417.528852ms
Mop/s total	80.36434330051999
Operation type	Random numbers generated
Verification	SUCCESSFUL
NPB Version	4.1

  

```
Process finished with the exit code 0
```

```
thouta@thouta: ~/Downloads/npbcpp/NPB-CPP/NPB-OMP/bin
File Edit View Search Terminal Tabs Help

thouta@thouta: ~/Downloads/npbcpp... x thouta@thouta: ~/Downloads/npbcpp... x
```

EP Benchmark Results:

```
CPU Time = 0.2448
N = 2^ 24
No. Gaussian Pairs = 13176389
Sums = -3.247834652034389e+03 -6.958407078382825e+03
Counts:
0      6140517
1      5865300
2      1100361
3       68546
4        1648
5          17
6           0
7           0
8           0
```

  

```
EP Benchmark Completed
class_npb      = S
Size           = 33554432
Total threads  = 8
Iterations     = 0
Time in seconds = 0.24
Mop/s total    = 137.07
Operation type = Random numbers generated
Verification   = SUCCESSFUL
Version        = 4.1
Compile date   = 15 Feb 2023
Compiler ver   = 11.3.0
OpenMP version = 201511
```

  

```
Compile options:
CC      = g++ -std=c++14
CLINK   = $(CC)
C_LIB   = -lm
C_INC   = -I../common
CFLAGS  = -O3 -fopenmp -mcmodel=medium
CLINKFLAGS = -O3 -fopenmp -mcmodel=medium
RAND    = randdp
```

# Resultados: A

```
Run: go build run x
Number of random numbers generated: 5.36870912e+08
```

EP BENCHMARK RESULTS	
CPU Time	6.634792301s
N	2^28
No. Gaussian Pairs	2.10832767e+08
Sums	-4295.875165632618 -15807.32573678597
Counts	
0	9.8257395e+07
1	9.3827014e+07
2	1.7611549e+07
3	1.110028e+06
4	26536
5	245
6	0
7	0
8	0

  

EP BENCHMARK COMPLETED	
Class	A
Size	536870912
Number of available threads	8
Number of iterations	0
Time in seconds	6.634792301s
Mop/s total	80.9175159739488
Operation type	Random numbers generated
Verification	SUCCESSFUL
NPB Version	4.1

  

```
Process finished with the exit code 0
```

```
thouta@thouta: ~/Downloads/npbcpp/NPB-CPP/NPB-OMP/bin
File Edit View Search Terminal Tabs Help

thouta@thouta: ~/Downloads/npbcpp... x thouta@thouta: ~/Downloads/npbcpp... x

EP Benchmark Results:

CPU Time = 3.2421
N = 2^ 28
No. Gaussian Pairs = 210832767
Sums = -4.295875165636600e+03 -1.580732573678463e+04
Counts:
0 98257395
1 93827014
2 17611549
3 1110028
4 26536
5 245
6 0
7 0
8 0

EP Benchmark Completed
class_npb = A
Size = 536870912
Total threads = 8
Iterations = 0
Time in seconds = 3.24
Mop/s total = 165.59
Operation type = Random numbers generated
Verification = SUCCESSFUL
Version = 4.1
Compile date = 15 Feb 2023
Compiler ver = 11.3.0
OpenMP version = 201511

Compile options:
CC = g++ -std=c++14
CLINK = $(CC)
C_LIB = -lm
C_INC = -I../common
CFLAGS = -O3 -fopenmp -mcmmodel=medium
CLINKFLAGS = -O3 -fopenmp -mcmmodel=medium
RAND = randdp
```

# Resultados: B

Run: go build run x

Number of random numbers generated: 2.147483648e+09

EP BENCHMARK RESULTS	
CPU Time	25.892174469s
N	2^30
No. Gaussian Pairs	8.43345606e+08
Sums	40338.155424419456 -26606.691928105378
Counts	
0	3.9305847e+08
1	3.75280898e+08
2	7.0460742e+07
3	4.438852e+06
4	105691
5	948
6	5
7	0
8	0

EP BENCHMARK COMPLETED	
Class	B
Size	2147483648
Number of available threads	8
Number of iterations	0
Time in seconds	25.892174469s
Mop/s total	82.93948623631917
Operation type	Random numbers generated
Verification	SUCCESSFUL
NPB Version	4.1

Process finished with the exit code 0

thouta@thouta: ~/Downloads/npbcpp/NPB-CPP/NPB-OMP/bin

File Edit View Search Terminal Tabs Help

thouta@thouta: ~/Downloads/npbcpp... x

thouta@thouta: ~/Downloads/npbcpp... x

EP Benchmark Results:

```
CPU Time = 12.2288
N = 2^ 30
No. Gaussian Pairs = 843345606
Sums = 4.033815542441809e+04 -2.660669192812149e+04
Counts:
0 393058470
1 375280898
2 70460742
3 4438852
4 105691
5 948
6 5
7 0
8 0
```

EP Benchmark Completed

```
class_npb = B
Size = 2147483648
Total threads = 8
Iterations = 0
Time in seconds = 12.23
Mop/s total = 175.61
Operation type = Random numbers generated
Verification = SUCCESSFUL
Version = 4.1
Compile date = 15 Feb 2023
Compiler ver = 11.3.0
OpenMP version = 201511
```

Compile options:

```
CC = g++ -std=c++14
CLINK = $(CC)
C_LIB = -lm
C_INC = -I../common
CFLAGS = -O3 -fopenmp -mcmodel=medium
CLINKFLAGS = -O3 -fopenmp -mcmodel=medium
RAND = randdp
```

# Resultados: C

```
Run: go build run x
Number of random numbers generated: 8.589934592e+09
```

EP BENCHMARK RESULTS	
CPU Time	1m49.350633847s
N	2^32
No. Gaussian Pairs	3.373275903e+09
Sums	47643.679279943346 -80840.72988043903
Counts	
0	1.572172634e+09
1	1.501108549e+09
2	2.81805648e+08
3	1.7761221e+07
4	424017
5	3821
6	13
7	0
8	0

  

EP BENCHMARK COMPLETED	
Class	C
Size	8589934592
Number of available threads	8
Number of iterations	0
Time in seconds	1m49.350633847s
Mop/s total	78.55404481714088
Operation type	Random numbers generated
Verification	SUCCESSFUL
NPB Version	4.1

  

Process finished with the exit code 0

```
thouta@thouta: ~/Downloads/npcpp/NPB-CPP/NPB-OMP/bin
File Edit View Search Terminal Tabs Help

thouta@thouta: ~/Downloads/npcpp... x thouta@thouta: ~/Downloads/npcpp... x

EP Benchmark Results:

CPU Time = 49.2901
N = 2^ 32
No. Gaussian Pairs = 3373275903
Sums = 4.764367927995972e+04 -8.084072988049158e+04
Counts:
0 1572172634
1 1501108549
2 281805648
3 17761221
4 424017
5 3821
6 13
7 0
8 0

EP Benchmark Completed
class_npb = C
Size = 8589934592
Total threads = 8
Iterations = 0
Time in seconds = 49.29
Mop/s total = 174.27
Operation type = Random numbers generated
Verification = SUCCESSFUL
Version = 4.1
Compile date = 15 Feb 2023
Compiler ver = 11.3.0
OpenMP version = 201511

Compile options:
CC = g++ -std=c++14
CLINK = $(CC)
C_LIB = -lm
C_INC = -I../common
CFLAGS = -O3 -fopenmp -mcmodel=medium
CLINKFLAGS = -O3 -fopenmp -mcmodel=medium
RAND = randdp
```

# Relatório 1: Benchmark EP Em Golang

Kevin S. Pereira, Thomazio Giacobbe

<https://github.com/thomaziogiacobbe/NPB-Golang>