

Os valores de testes estão presentes no código, na forma de dois arrays, representando a quantidade de valores que são manipulados, respectivamente:

```
int[] valoresDeTesteMapeamento = { 1, 2, 3, 10, 100, 1000, 10000, 100000, 1000000, 10000000 };  
int[] valoresDeTesteLista = { 1, 2, 3, 10, 100, 1000, 20000, 10000 };
```

E as coleções a serem comparadas:

```
Map<Integer, Integer> mapOne = new ConcurrentHashMap<Integer, Integer>();  
Map<Integer, Integer> mapTwo = Collections.synchronizedMap(new HashMap<Integer, Integer>());  
  
CopyOnWriteArrayList<Integer> listOne = new CopyOnWriteArrayList<>();  
List<Integer> listTwo = Collections.synchronizedList(new ArrayList<Integer>());
```

Comparando ConcurrentHashMap e Collections.SynchronizedMap, os resultados foram obtidos conforme segue abaixo:

```
Teste Concurrent HashMap  
Test Put, Qtd_valores: 1, Time: 0  
Test Put, Qtd_valores: 2, Time: 0  
Test Put, Qtd_valores: 3, Time: 0  
Test Put, Qtd_valores: 10, Time: 0  
Test Put, Qtd_valores: 100, Time: 0  
Test Put, Qtd_valores: 1000, Time: 0  
Test Put, Qtd_valores: 10000, Time: 18  
Test Put, Qtd_valores: 100000, Time: 121  
Test Put, Qtd_valores: 1000000, Time: 793  
Test Put, Qtd_valores: 10000000, Time: 19378
```

```
Teste Collections.synchronizedMap  
Test Put, Qtd_valores: 1, Time: 1  
Test Put, Qtd_valores: 2, Time: 1  
Test Put, Qtd_valores: 3, Time: 1  
Test Put, Qtd_valores: 10, Time: 32  
Test Put, Qtd_valores: 100, Time: 358  
Test Put, Qtd_valores: 1000, Time: 4  
Test Put, Qtd_valores: 10000, Time: 12  
Test Put, Qtd_valores: 100000, Time: 54  
Test Put, Qtd_valores: 1000000, Time: 477  
Test Put, Qtd_valores: 10000000, Time: 43312
```

```
Teste Concurrent HashMap  
Test Get, Qtd_valores: 1, Time: 0  
Test Get, Qtd_valores: 2, Time: 0  
Test Get, Qtd_valores: 3, Time: 0  
Test Get, Qtd_valores: 10, Time: 0  
Test Get, Qtd_valores: 100, Time: 0  
Test Get, Qtd_valores: 1000, Time: 0  
Test Get, Qtd_valores: 10000, Time: 10  
Test Get, Qtd_valores: 100000, Time: 20  
Test Get, Qtd_valores: 1000000, Time: 10  
Test Get, Qtd_valores: 10000000, Time: 120
```

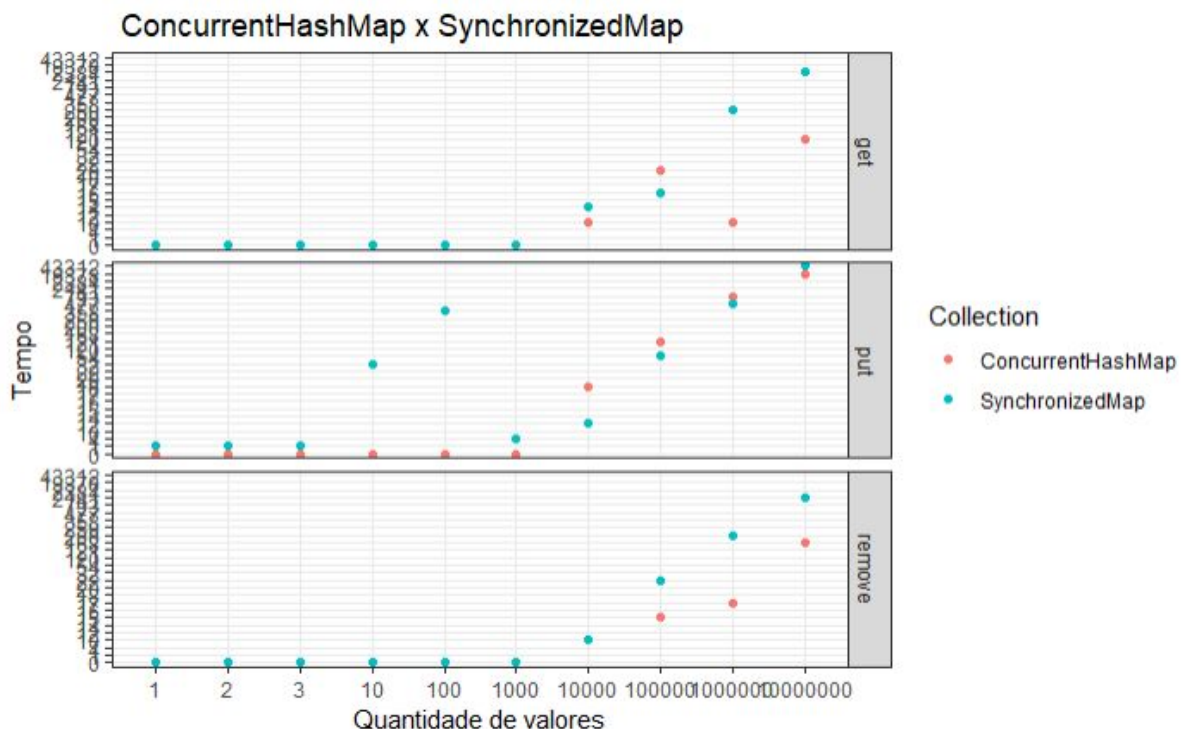
```
Teste Collections.synchronizedMap  
Test Get, Qtd_valores: 1, Time: 0  
Test Get, Qtd_valores: 2, Time: 0  
Test Get, Qtd_valores: 3, Time: 0  
Test Get, Qtd_valores: 10, Time: 0  
Test Get, Qtd_valores: 100, Time: 0  
Test Get, Qtd_valores: 1000, Time: 0  
Test Get, Qtd_valores: 10000, Time: 14  
Test Get, Qtd_valores: 100000, Time: 16  
Test Get, Qtd_valores: 1000000, Time: 250  
Test Get, Qtd_valores: 10000000, Time: 2384
```

Teste Concurrent HashMap  
 Test Remove, Qtd\_valores: 1, Time: 0  
 Test Remove, Qtd\_valores: 2, Time: 0  
 Test Remove, Qtd\_valores: 3, Time: 0  
 Test Remove, Qtd\_valores: 10, Time: 0  
 Test Remove, Qtd\_valores: 100, Time: 0  
 Test Remove, Qtd\_valores: 1000, Time: 0  
 Test Remove, Qtd\_valores: 10000, Time: 10  
 Test Remove, Qtd\_valores: 100000, Time: 15  
 Test Remove, Qtd\_valores: 1000000, Time: 17  
 Test Remove, Qtd\_valores: 10000000, Time: 168

Teste Collections.synchronizedMap  
 Test Remove, Qtd\_valores: 1, Time: 0  
 Test Remove, Qtd\_valores: 2, Time: 0  
 Test Remove, Qtd\_valores: 3, Time: 0  
 Test Remove, Qtd\_valores: 10, Time: 0  
 Test Remove, Qtd\_valores: 100, Time: 0  
 Test Remove, Qtd\_valores: 1000, Time: 0  
 Test Remove, Qtd\_valores: 10000, Time: 10  
 Test Remove, Qtd\_valores: 100000, Time: 22  
 Test Remove, Qtd\_valores: 1000000, Time: 200  
 Test Remove, Qtd\_valores: 10000000, Time: 2142

É possível visualizar através da quantidade de operações representada por “Qtd\_valores”, que Concurrent HashMap mantêm melhor desempenho, conseguindo realizar as operações em menor tempo do que Collections.synchronizedMap.

Alguns fatores contribuem para tal, por exemplo: Collections.synchronizedMap bloqueia todo o mapa em torno de cada operação, enquanto Concurrent Hashap bloqueia apenas algumas partes para algumas operações, ou em alguns casos, não uso algoritmo de bloqueio para algumas operações.



Comparações entre CopyOnWriteArrayList e Collections.synchronizedList, através do código, obtemos os seguintes dados:

Teste CopyOnWriteArrayList  
Test Add, Qtd\_valores: 1, Time: 0  
Test Add, Qtd\_valores: 2, Time: 0  
Test Add, Qtd\_valores: 3, Time: 0  
Test Add, Qtd\_valores: 10, Time: 9  
Test Add, Qtd\_valores: 100, Time: 0  
Test Add, Qtd\_valores: 1000, Time: 21  
Test Add, Qtd\_valores: 20000, Time: 3718  
Test Add, Qtd\_valores: 10000, Time: 4838

Teste Collections.synchronizedList  
Test Add, Qtd\_valores: 1, Time: 1  
Test Add, Qtd\_valores: 2, Time: 1  
Test Add, Qtd\_valores: 3, Time: 1  
Test Add, Qtd\_valores: 10, Time: 1  
Test Add, Qtd\_valores: 100, Time: 1  
Test Add, Qtd\_valores: 1000, Time: 2  
Test Add, Qtd\_valores: 20000, Time: 13  
Test Add, Qtd\_valores: 10000, Time: 5

Teste CopyOnWriteArrayList  
Test Get, Qtd\_valores: 1, Time: 0  
Test Get, Qtd\_valores: 2, Time: 0  
Test Get, Qtd\_valores: 3, Time: 8  
Test Get, Qtd\_valores: 10, Time: 0  
Test Get, Qtd\_valores: 100, Time: 0  
Test Get, Qtd\_valores: 1000, Time: 0  
Test Get, Qtd\_valores: 20000, Time: 9  
Test Get, Qtd\_valores: 10000, Time: 11

Teste Collections.synchronizedList  
Test Get, Qtd\_valores: 1, Time: 0  
Test Get, Qtd\_valores: 2, Time: 0  
Test Get, Qtd\_valores: 3, Time: 3  
Test Get, Qtd\_valores: 10, Time: 0  
Test Get, Qtd\_valores: 100, Time: 0  
Test Get, Qtd\_valores: 1000, Time: 0  
Test Get, Qtd\_valores: 20000, Time: 0  
Test Get, Qtd\_valores: 10000, Time: 0

Teste CopyOnWriteArrayList  
Test Remove, Qtd\_valores: 1, Time: 16  
Test Remove, Qtd\_valores: 2, Time: 0  
Test Remove, Qtd\_valores: 3, Time: 0  
Test Remove, Qtd\_valores: 10, Time: 31  
Test Remove, Qtd\_valores: 100, Time: 5  
Test Remove, Qtd\_valores: 1000, Time:  
141  
Test Remove, Qtd\_valores: 20000, Time:  
48972  
Test Remove, Qtd\_valores: 10000, Time:  
28897

Teste Collections.synchronizedList  
Test Remove, Qtd\_valores: 1, Time: 16  
Test Remove, Qtd\_valores: 2, Time: 2  
Test Remove, Qtd\_valores: 3, Time: 2  
Test Remove, Qtd\_valores: 10, Time: 4  
Test Remove, Qtd\_valores: 100, Time: 37  
Test Remove, Qtd\_valores: 1000, Time:  
33  
Test Remove, Qtd\_valores: 20000, Time:  
7959  
Test Remove, Qtd\_valores: 10000, Time:  
4628

É perceptível, que Collections.synchronizedList possui desempenho superior ao de CopyOnWriteArrayList, devido as sincronizações de cópias, gerando cópias das matrizes de

dados, alterando a cópia, e atualizando a referência volátil da matriz para apontar para essa nova matriz. É uma grande quantidade de operações que baixa o desempenho da coleção.

