Metodologias Experimentais em Informática

Teste de Hipóteses

Artur Coutinho - 2014230432 — Diogo Amores - 2015231975
Maria Inês Roseiro - 2015233281

6 de Dezembro de 2019

1 Introdução

Este relatório foi desenvolvido no âmbito da cadeira de *Metodologias Experiementais em Informática* tendo como objectivo o levantamento de hipóteses e o teste das mesmas, relativamente aos resultados obtidos na análise exploratória de dados. Todo este processo tem como objetivo a obtenção de uma resposta para a pergunta proposta:

De que forma algoritmos de ordenamento são afetados por memory faults?

Os scripts para a análise de dados, cálculo de intervalos de confiança e hipóteses testadas, foram realizados com recurso à linguagem R.

2 Variáveis

2.1 Variáveis independentes

Este tipo de variáveis, na maior parte dos casos, têm influência nos resultados obtidos nas experiências. Tal como referido no relatório anterior, foram consideradas 4 variáveis independentes:

1. Número de elementos da sequência *(n)* ;
2. Limite de variação dos elementos da sequência *(max_r)*;
3. Probabilidade de ocorrer um *memory fault* (*eps*);
4. O algoritmo de ordenamento *(QuickSort, MergeSort, InsertionSort e BubbleSort).*

2.2 Variáveis dependentes

Variáveis dependentes são determinadas através da realização das experiências. Para esta análise de dados foram consideradas três variáveis dependentes:

1. Tamanho da máxima subsequência ordenada *(max_size)*;
2. Tempo de execução de cada algoritmo *(execution_time)*;
3. Número de comparações que cada algoritmo efectua.

3 Intervalos de confiança

Estes intervalos são calculados de forma a que um valor se encontre no intervalo dos dados recolhidos, dada uma determinada probabilidade, sendo que, quanto maior o tamanho da amostra recolhida, mais minucioso se torna o valor obtido este intervalo.

A partir dos vários *datasets* provenientes da análise exploratória de dados, calculámos as médias *(m)* e desvios padrões *(sd)* para cada caso em análise, bem como os respectivos intervalos de confiança, com recurso à seguinte fórmula:

$$x \pm z_1 - \alpha \times \frac{sd}{\sqrt{n}}, \alpha = 0.05 \quad (1)$$

De forma a calcular um intervalo que considerássemos preciso, as amostras utilizadas contêm 100 elementos.

3.1 Variação do tamanho da sequência inicial *(n)*

Através dos resultados de *max_size* obtidos para cada algoritmo, para diferentes tamanhos de sequência *(n)* foram recolhidos os seguintes intervalos de confiança:

Algoritmo	n	Média	Desvio Padrão	Int. Confiança	Mínimo	Máximo
Bubble	100	79.62	20.18	4.01	46	100
Quick	100	40.97	13.23	2.62	22	78
Merge	100	40.93	10.84	2.15	19	71
Insertion	100	19.9	4.17	0.82	12	33
Bubble	1000	713.19	181.52	36.01	254	1000
Quick	1000	352.57	111.49	22.12	183	830
Merge	1000	395.32	194.59	38.61	158	577
Insertion	1000	78.47	19.48	3.86	44	133
Bubble	10000	7057.66	2101.56	416.99	445	1000
Quick	10000	2090.47	602.77	119.60	170	772
Merge	10000	2775.63	809.23	160.56	211	1000
Insertion	10000	256.95	49.05	9.73	45	136

3.2 Variação do limite de valores da sequência inicial *(max_r)*

Através dos resultados de *max_size* obtidos para cada algoritmo, para diferentes limites de valores da sequência *(max_r)* foram recolhidos os seguintes intervalos de confiança:

Algoritmo	max_r	Média	Desvio Padrão	Int. Confiança	Mínimo	Máximo
Bubble	500	653.14	192.69	38.23	342	993
Quick	500	244.79	75.01	14.88	130	462
Merge	500	288.14	93.32	18.51	131	839
Insertion	500	69.11	12.41	2.46	47	106
Bubble	1000	567.95	207.25	41.12	254	1000
Quick	1000	327.81	119.87	23.78	183	830
Merge	1000	285.73	93.02	18.456	158	577
Insertion	1000	68.47	14.97	2.97	44	133
Bubble	2000	713.19	181.52	36.01	445	1000
Quick	2000	352.57	111.49	22.12	170	772
Merge	2000	395.32	194.58	38.61	211	1000
Insertion	2000	78.47	19.48	3.86	45	136

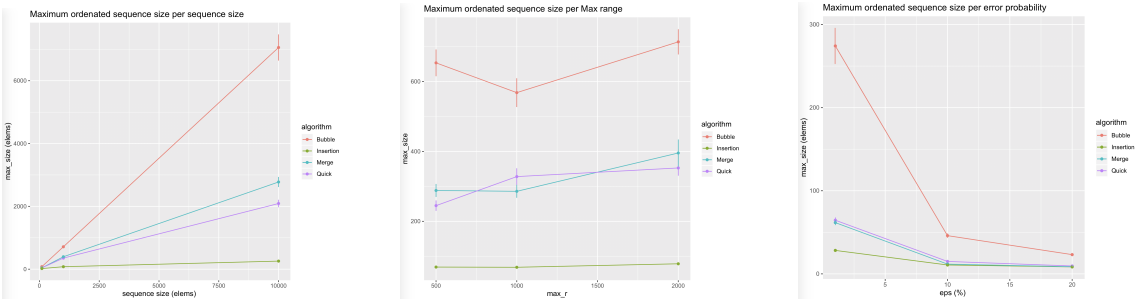
3.3 Variação da probabilidade de ocorrer um erro *(eps)*

Através dos resultados de *max_size* obtidos para cada algoritmo, para diferentes probabilidades de erro *eps* foram recolhidos os seguintes intervalos de confiança:

Algoritmo	eps	Média	Desvio Padrão	Int. Confiança	Mínimo	Máximo
Bubble	1	274.22	107.37	21.30	141	764

Table 3 continued from previous page

Quick	1	64.55	14.86	2.94	38	107
Merge	1	61.67	13.08	2.59	40	102
Insertion	1	28.23	5.17	1.03	20	45
Bubble	10	46	11.85	2.35	26	87
Quick	10	14.97	2.45	0.48	11	22
Merge	10	11.75	2.02	0.40	9	1 8
Insertion	10	10.7	1.33	0.26	9	15
Bubble	20	23.16	6.33	1.26	14	49
Quick	20	9.48	1.40	0.27	7	14
Merge	20	8.54	1.33	0.26	6	12
Insertion	20	8.37	1.01	0.20	7	11



(a) Intervalos de confiança para variações de n

(b) Intervalos de confiança para variações de *max_r*

(c) Intervalos de confiança para variações de *eps*

Figura 1: Gráficos com barras de erro

4 Testes de Hipóteses

4.1 Formulação de hipóteses

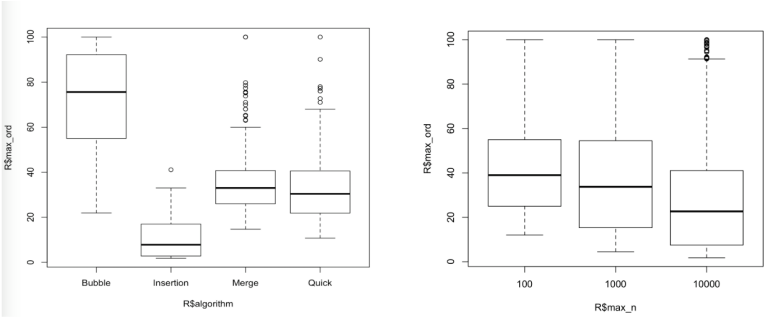
De maneira a verificar a interação das diversas variáveis independentes consideradas, foram formuladas 3 hipóteses:

H0: Será que a diferença observada na meta 1 entre algoritmos para valores 100, 1000 e 10000 de n é significativa?

H^A **0: Variável algoritmo não é significativa nas diferenças observadas.**

H^N **0: Variável n não é significativa nas diferenças observadas.**

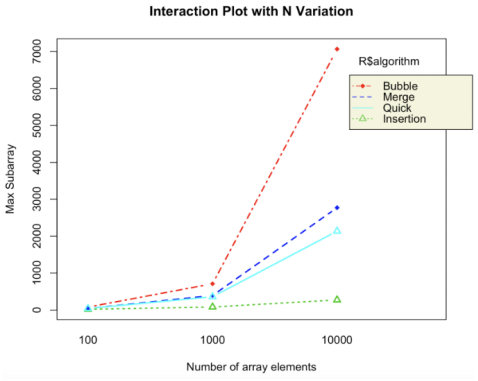
H^R **0: Não há interação entre as 2 variáveis.**



(a) Relação das variáveis dependentes com os diferentes algoritmos (variável independente)

Os dois gráficos apresentados mostram a percentagem de max_size máximo para cada variável independente. No caso específico do gráfico da direita, como esperado, o BubbleSort é o algoritmo que apresenta uma maior percentagem de valores de *max_size*, apresentando também uma média bastante elevada em comparação com os restantes. O InsertionSort é o algoritmo que tem a pior performance, estando bastante aquém dos restantes três. O MergeSort e o QuickSort mantém uma performance relativamente semelhante.

Por fim, é possível verificar que poderá existir uma relação entre esta variável independente e os valores da nossa variável dependente *(max_size)*. Relativamente ao segundo boxplot, a diferença é muito menos significativa. A tendência é, para valores mais altos de *max_n*, os valores de *max_size* diminuem, o que provavelmente indica uma relação entre estas duas variáveis.



(a) Interaction Plot com n e *max_size*

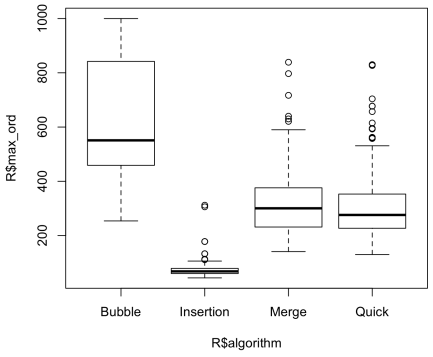
Através da visualização do interaction plot é notório a ausência de linhas paralelas entre as interações destas variáveis. Desta forma, é possível sugerir uma interação entre as duas variáveis independentes e o max_size. Os valores de max_size são diferentes entre algoritmo e max_n, existindo uma tendência ascendente dos valores quando os valores de max_n aumentam.

H0: Será que a diferença observada na meta 1 entre algoritmos para n/2, n e 2n de *max_r* é significativa?

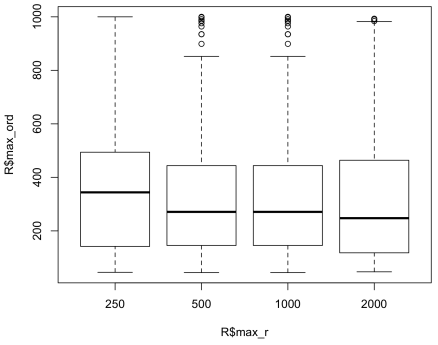
H^A **0: Variável algoritmo não é significativa nas diferenças observadas.**

H^M **0: Variável *max_r* não é significativa nas diferenças observadas.**

H^R **0: Não há interação entre as 2 variáveis.**



(a) Relação dos valores de *max_ord* (variável dependente) com os diferentes algoritmos (variável independente)



(b) Relação dos valores de *max_r* (variável dependente) com os diferentes algoritmos (variável independente)

Figura 4: Boxplots relativos à segunda hipótese formulada

Através da análise dos dois gráficos anteriores é possível verificar que existe uma diferença na performance de cada algoritmo. O BubbleSort como já concluído com a análise exploratória de dados, obtém os melhores resultados bem como InsertionSort os piores. O MergeSort e QuickSort obtém resultados relativamente semelhantes. Relativamente à mudança de max_r, conseguimos perceber pelo que existe alguma influência, visível com max_r de 250 (n/4), que gera resultados ligeiramente melhores (obtém a maior média também). No entanto, a diferença que apresentamos não é tão pronunciada como no boxplot *(a)*, onde se verifica facilmente a diferença de performances. Estes resultados podem indicar uma potencial relação entre a variáveis independente (algoritmo e max_r) e a variável dependente (subarray).

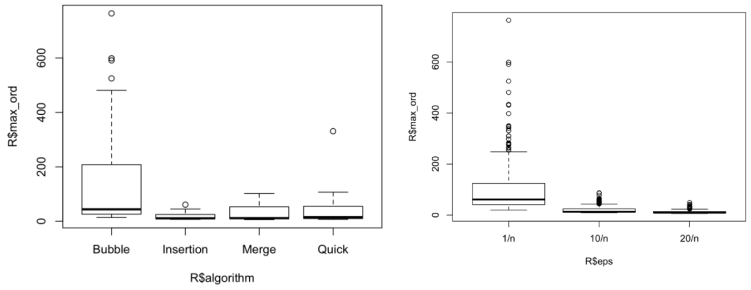
(a) Interaction Plot com *max_r* e *max_size*

Através da visualização do *interaction plot* entre *max_r* e *max_size*, é visível que não existe paralelismo entre nenhum par de linhas de interação. Assim, é possível sugerir que, de facto poderá existir interação entre estas duas variáveis.

O subarray máximo obtém valores distintos quando *max_r* é alterado, sendo notório que os valores de *max_size* tendem a decrescer com o aumento de *max_r*. No entanto, o BubbleSort contraria esta tendência com um decréscimo inicial, uma estabilização até 1000, seguido de um aumento da performance.

H0: Será que a diferença observada na meta 1 entre algoritmos para n/2, n e 2n de *eps* é significativa?

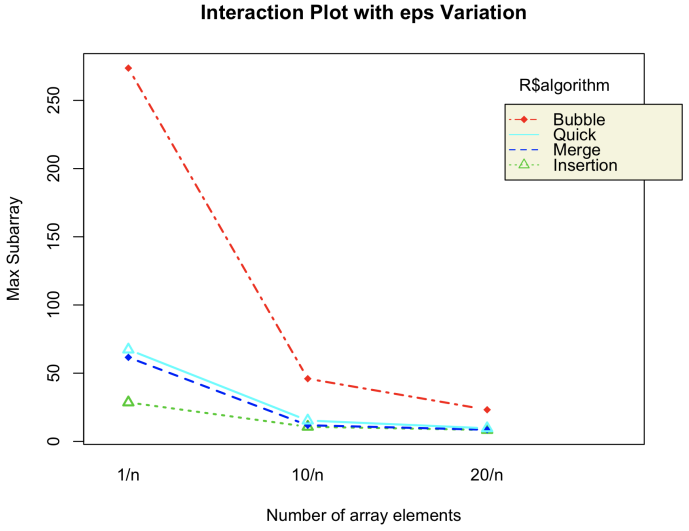
H^A **0: Variável algoritmo não é significativa nas diferenças observadas.**
H^E **0: Variável *eps* não é significativa nas diferenças observadas.**
H^R **0: Não há interação entre as 2 variáveis.**



(a) Relação das variáveis dependentes com os diferentes algoritmos (variável independente)

Através da análise dos boxplots é possível observar, no primeiro caso, uma diferença nas performances de cada algoritmo a nível do *max_size*. O BubbleSort contém os melhores resultados de uma maneira geral. No entanto, estas diferenças (relativamente a médias) são muito menos significativas que nos casos anteriormente estudados. Apesar disso, é sugestiva uma possível relação entre a variável dependente *max_size* e a variável independente algoritmo.

No segundo caso, existe uma diferença acrescida entre os valores de *eps* sendo que o primeiro valor apresenta os melhores resultados. Para além disso 10/n e 20/n mantêm valores bastante similares, o que poderá implicar uma certa relação entre esta variável e a variável independente (max_size).



(a) Interaction Plot com *eps* e *max_size*

Com este interaction plot é possível sugerir uma interação entre cada uma das variáveis independentes devido à inexistência de linhas paralelas no gráfico. A tendência é para um decréscimo de *max_size* com o aumento de eps. Mais uma vez, o **BubbleSor**t é o algoritmo que efetua a melhor performance, sendo que os restantes oscilam dentro de níveis próximos.

4.2 Cálculos

```
                                    Df    Sum Sq   Mean Sq F value Pr(>F)
as.factor(max_n)                     2 2.159e+09 1.079e+09  2271.3 <2e-16 ***
as.factor(algorithm)                 3 9.894e+08 3.298e+08   694.0 <2e-16 ***
as.factor(max_n):as.factor(algorithm)    6 1.499e+09 2.498e+08   525.7 <2e-16 ***
Residuals                         1188 5.646e+08 4.752e+05
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

(a) Two-Way Anova para n

```
                                    Df   Sum Sq  Mean Sq F value Pr(>F)
as.factor(max_r)                     3  1612101   537367   33.77 <2e-16 ***
as.factor(algorithm)                 3 63581404 21193801 1331.92 <2e-16 ***
as.factor(max_r):as.factor(algorithm)    9  2024911   224990   14.14 <2e-16 ***
Residuals                         1584 25204967    15912
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

(a) Two-Way Anova para *max_r*

```
                                    Df  Sum Sq Mean Sq F value Pr(>F)
as.factor(eps)                       2 2200639 1100320  1028.0 <2e-16 ***
as.factor(algorithm)                 3 1832621  610874   570.7 <2e-16 ***
as.factor(eps):as.factor(algorithm)    6 2001261  333544   311.6 <2e-16 ***
Residuals                         1188 1271632    1070
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

(a) Two-Way Anova para *eps*

Recorrendo aos resultados obtidos pelo teste de Two-way ANOVA, com *p-values* e níveis de significância de 0.05 é possível tirar as seguintes conclusões:

- O p-value das variáveis independentes *(n, max_r e eps)* é menor que 0.05 sendo então possível concluir que os valores de *max_r* são significantes e associados à nossa variável dependente *max_size*.
- O p-value do *algoritmo* é menor que 0.05 sendo então possível concluir que os valores de *max_r* são significantes e associados à nossa variável dependente *max_size*.
- O p-value da interação entre as variáveis independentes e o algoritmo é menor que 0.05 sendo então possível concluir que esta relação entre as respectivas variáveis é significante para os resultados obtidos para a variável dependente *(max_size)*.

No entanto, antes de, efectivamente se poder rejeitar ou aceitar as hipóteses propostas, é necessário confirmar os pressupostos assumidos, utilizando o teste de **Shapiro-Wilk** para normalidade do dataset (este segue uma distribuição normal) e o teste de **Bartlett** para homogeneidade das variâncias do mesmo.

```
    Bartlett test of homogeneity of variances

data:  R$max_ord by interaction(R$max_n, R$algorithm)
Bartlett's K-squared = 4734.3, df = 11, p-value < 2.2e-16
```

(a) Bartlett test

```
        Shapiro-Wilk normality test

data:  anova2$res
W = 0.6229, p-value < 2.2e-16
```

(b) Shapiro-Wilk test

Figura 11: Valores para n

```
Bartlett test of homogeneity of variances

data:  R$max_ord by interaction(R$max_r, R$algorithm)
Bartlett's K-squared = 1333.1, df = 15, p-value < 2.2e-16
```

(a) Bartlett test

```
Shapiro-Wilk normality test

data:  anova2$res
W = 0.92995, p-value < 2.2e-16
```

(b) Shapiro-Wilk test

Figura 12: Valores para *max_r*

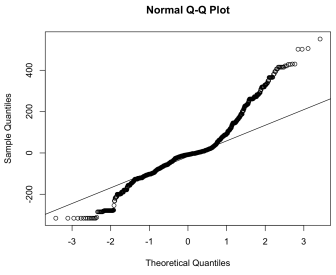
```
Bartlett test of homogeneity of variances

data:  R$max_ord by interaction(R$eps, R$algorithm)
Bartlett's K-squared = 4282.9, df = 11, p-value < 2.2e-16
```

(a) Bartlett test

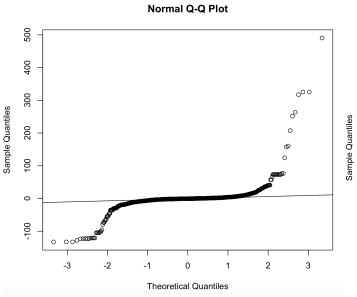
```
Shapiro-Wilk normality test

data:  anova2$res
W = 0.39349, p-value < 2.2e-16
```

(b) Shapiro-Wilk test

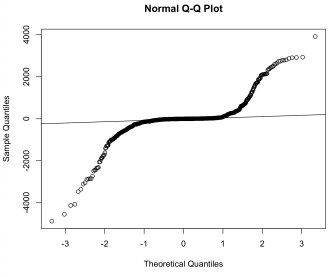
Figura 13: Valores para *eps*



(a) Normality plot para *max_r*



(b) Normality plot para *eps*



(c) Normality plot para *n*

Porém, após o resultado dos testes de Shapiro-Wilk e Bartlett é visível que o nossos dataset falham em ambos os testes. Podemos então concluir que os nossos dados não seguem uma curva normal e também não é possível assumir uma variância homogénea nos dados.

Assim, não podemos de imediato aceitar ou rejeitar a hipótese formulada, apesar de o teste de Two-way ANOVA ser conhecido por funcionar mesmo quando as suposições de normalidade e homogeneidade de variância serem rejeitadas. Optámos então por utilizar um teste de Randomization Non-Parametric TWO-WAY ANOVA para oficialmente aceitar ou rejeitar a hipótese formulada inicialmente.

"Randomization Non-Parametric TWO-WAY ANOVA" [VARIÁVEL INDEPENDENTE] 0" "algoritmo: 0" "[VARIÁVEL INDEPENDENTE]:algoritmo: 6e-04" [VARÁVEL INDEPENDENTE]- n, max_r, eps

Após a realização deste teste, podemos então aceitar as conclusões fornecidas pelo teste de **Two-Way ANOVA inicial**, visto os valores serem bastante semelhantes aos valores inicialmente encontrados.

Sendo assim, é possível rejeitar as hipóteses nula e consequentemente, as hipóteses dependentes, pelas razões especificadas no teste de Two-Way ANOVA.

4.3 Post-HOC Testes - Tuckey

Os testes de post-HOC encontram-se em anexo, juntamente com os seus resultados. Nestes testes, as interações relevantes dos resultados estão sublinhadas, sendo estas todas que cumpram a condição

p adj maior que 0.05.

5 Referências

1. https://www.r-bloggers.com/box-plot-with-r-tutorial/
2. https://towardsdatascience.com/exploratory-data-analysis-in-r-explore-one-variable-using-pseudo-facebook-dataset-29031767eb07
3. https://www.geeksforgeeks.org/analysis-of-different-sorting-techniques/
4. https://statistics.laerd.com/spss-tutorials/two-way-anova-using-spss-statistics-2.php
5. http://www.sthda.com/english/wiki/two-way-anova-test-in-r