# **Executar os experimentos e coletar os dados (medidas) - Run the experiments and collect the data/measurements**

Nesta seção, apresentamos os resultados obtidos após a execução dos algoritmos de ordenação tendo como entrada os arquivos contendo os arrays gerados para os diferentes tamanhos (100, 1000 e 10000) e probabilidades de erro (0.01, 0.02 e 0.05).

Importante ressaltar que, foram analisadas somente as variáveis dependentes *Percentage of largest subarray size* (***%LSS***) e *Percentage of unordered elements quantity* (***%UEQ***),uma vez que estas variáveis, por serem um percentual, já se encontram normalizadas (i.e. numa mesma ordem de grandeza) em relação à variável independente *Array size*.

## Exploratory Data Analysis

Primeiramente foi realizada uma análise sobre a distribuição das variáveis dependentes %LSS e %UEQ. Para auxiliar nesta tarefa, foram gerados histogramas, gráficos boxplot e tabelas contendo os dados sobre a média, a mediana, o desvio padrão e os valores mínimo e.

As figuras a seguir ilustram alguns exemplos dos histogramas e os gráficos boxplot que foram gerados para cada combinação de *Algoritmo* X *Prob. Falha* X *Tam. Array*. Em cada figura, estão exibidos os gráficos sobre as variáveis dependentes %LSS e %UEQ.

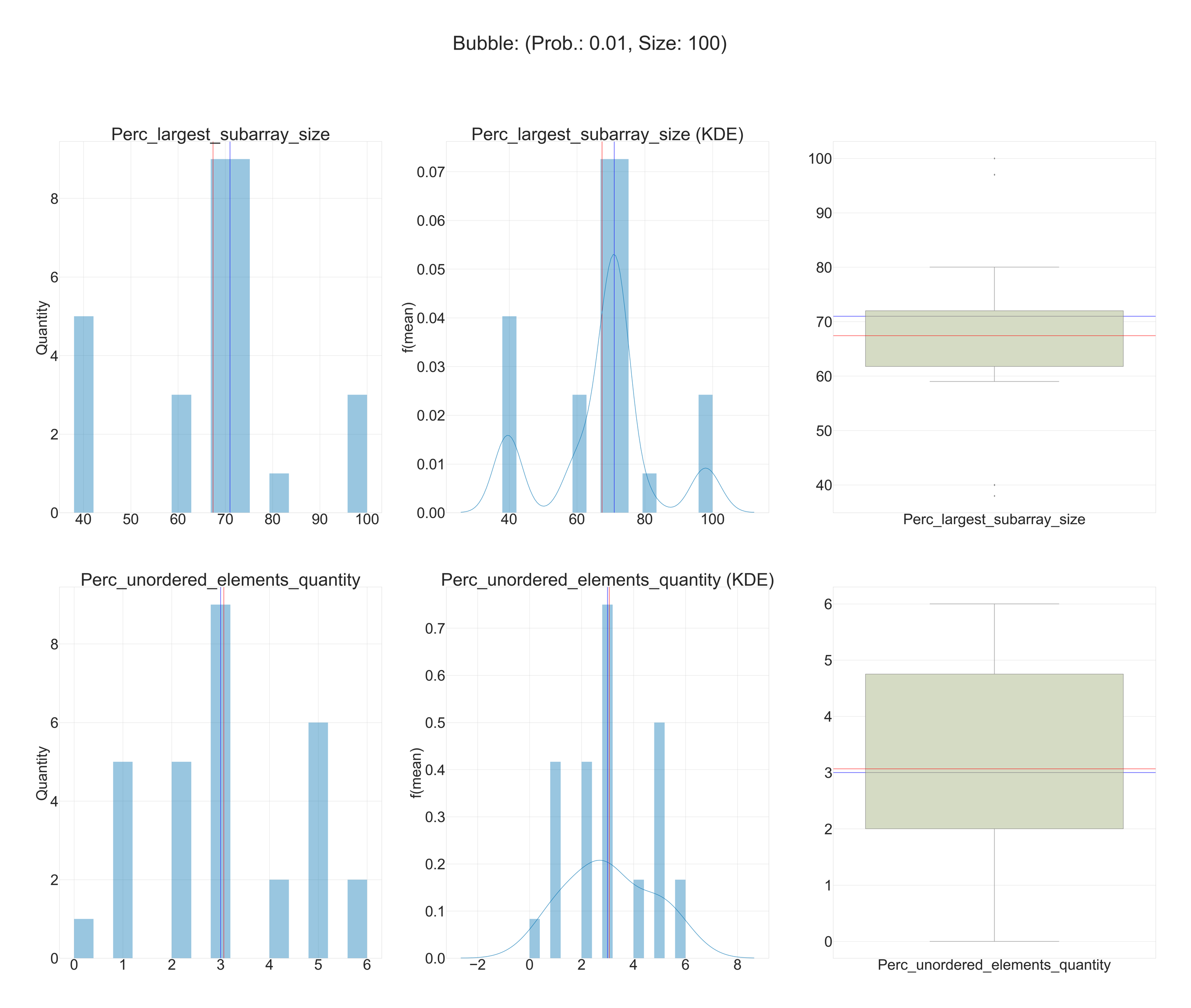


Figura : Histogramas e boxplot gerados para o algoritmo BubbleSort, com prob. erro 0.01 e tamanho de array 100

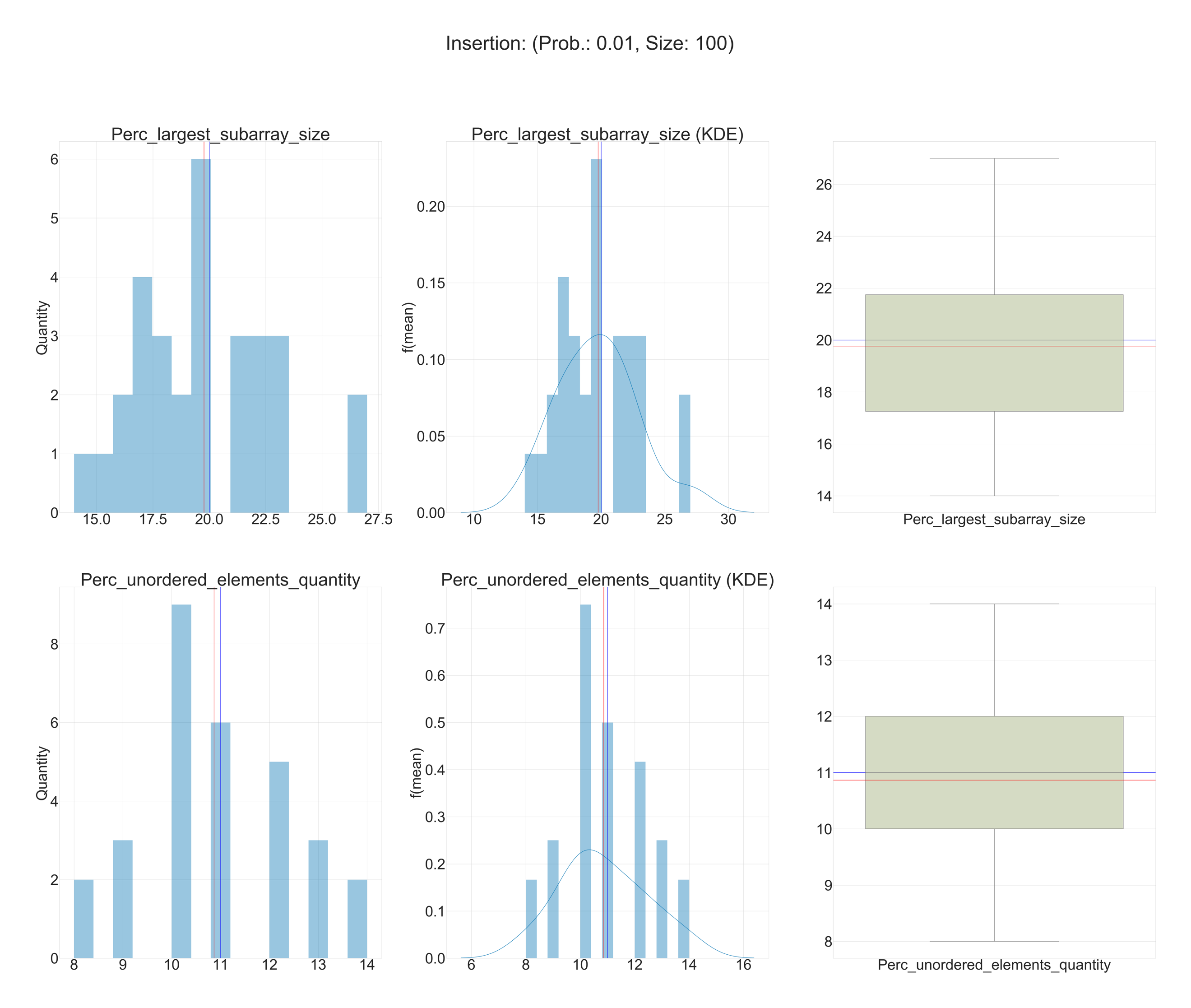


Figura : Histogramas e boxplot gerados para o algoritmo InsertionSort, com prob. erro 0.01 e tamanho de array 100

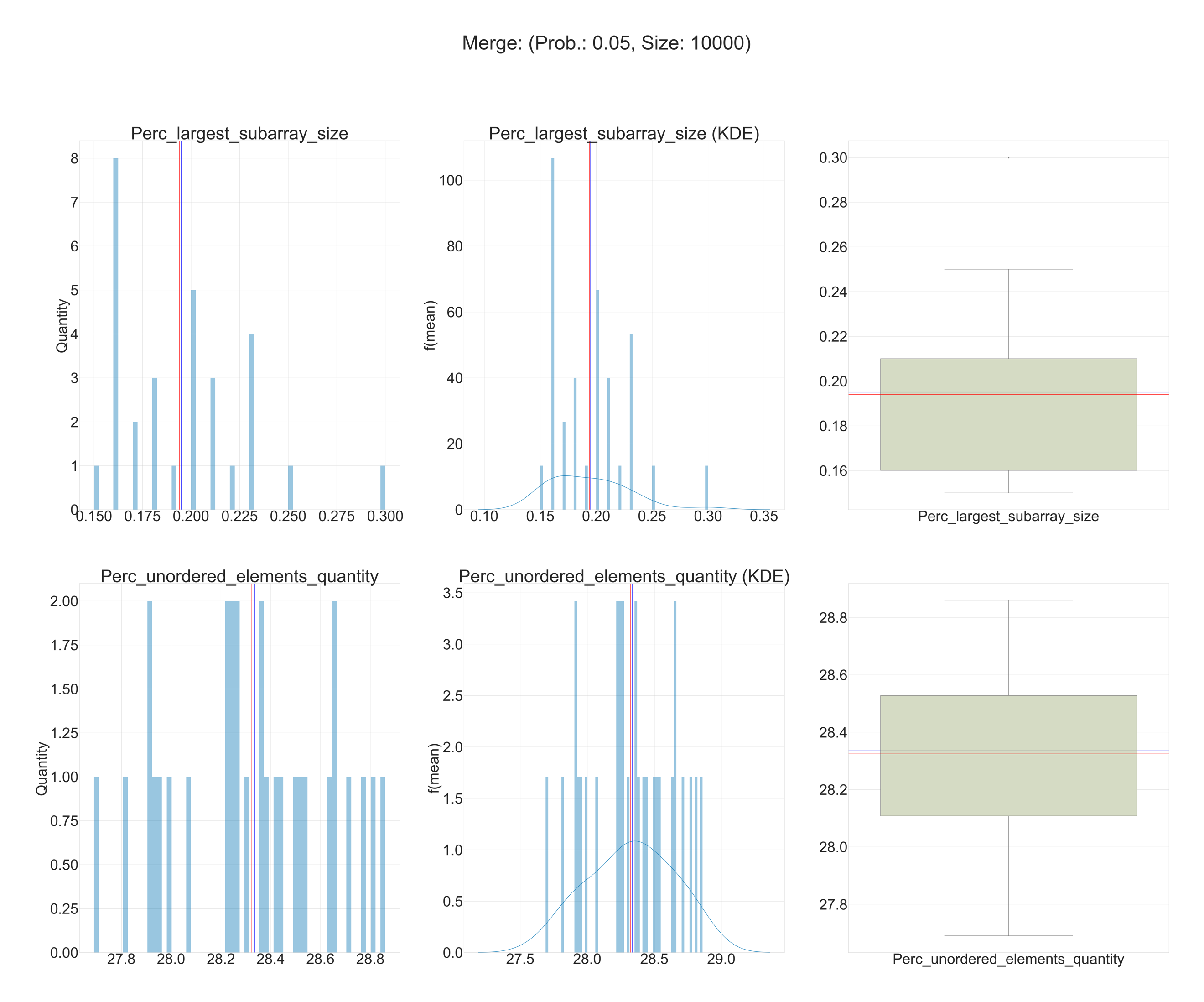


Figura : Histogramas e boxplot gerados para o algoritmo MergeSort, com prob. erro 0.05 e tamanho de array 10000

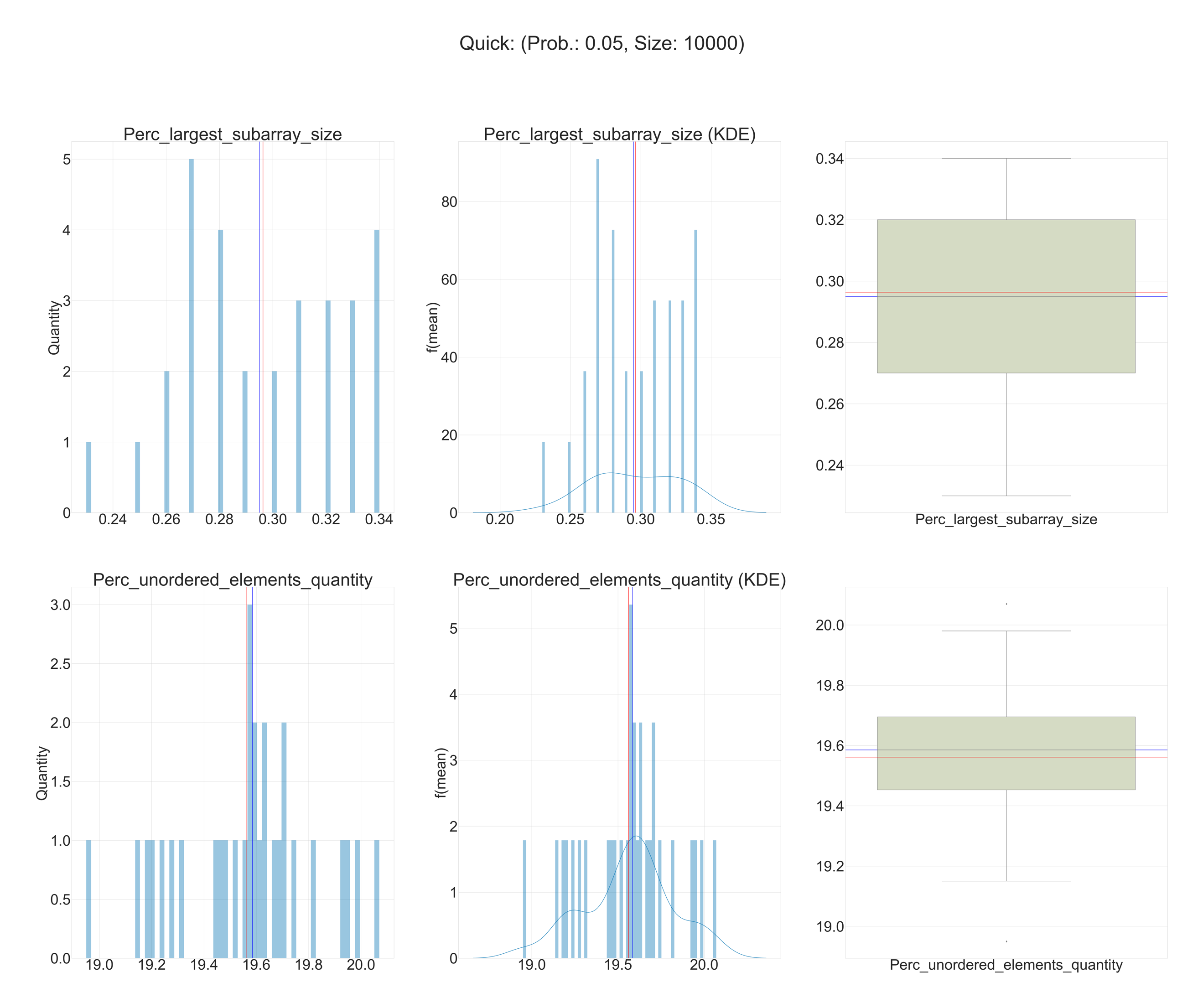


Figura : Histogramas e boxplot gerados para o algoritmo QuickSort, com prob. erro 0.05 e tamanho de array 10000

Nas tabelas a seguir, de acordo com as figuras exibidas anteriormente, estão ilustradas as informações sobre a distribuição das variáveis dependentes %UEQ e %LSS.

Tabela : Dados sobre a distribuição da variável dependente %UEQ

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Prob. failure | Array size | Algorithm | Percentage of unordered elements quantity (%UEQ) | | | | |
| Mean | Median | Std. Deviation | Minimum | Maximum |
| 0.01 | 100 | bubble | 3.07 | 3.0 | 1.64 | 0.0 | 6.0 |
| 0.01 | 100 | insertion | 10.87 | 11.0 | 1.59 | 8.0 | 14.0 |
| 0.05 | 10000 | merge | 28.32 | 28.34 | 0.31 | 27.69 | 28.86 |
| 0.05 | 10000 | quick | 19.56 | 19.58 | 0.26 | 18.95 | 20.07 |

Tabela : Dados sobre a distribuição da variável dependente %LSS

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Prob. failure | Array size | Algorithm | Percentage of largest subarray size (%LSS) | | | | |
| Mean | Median | Std. Deviation | Minimum | Maximum |
| 0.01 | 100 | bubble | 67.4 | 71.0 | 15.84 | 38.0 | 100.0 |
| 0.01 | 100 | insertion | 19.77 | 20.0 | 3.11 | 14.0 | 27.0 |
| 0.05 | 10000 | merge | 0.19 | 0.2 | 0.03 | 0.15 | 0.3 |
| 0.05 | 10000 | quick | 0.3 | 0.3 | 0.03 | 0.23 | 0.34 |

Os dados sobre a distribuição das variáveis dependentes %UEQ e %LSS, representados nas tabelas acima, também foram disponibilizados em forma de gráficos. Um exemplo pode ser na figura seguir.

Uma imagem contendo instrumento para escrita, material de escritório, lápis

Descrição gerada automaticamente

Figura : Dados de distribuição para prob. erro 0.01 e todos os algoritmos e tamanhos de array

## Verificação quanto à normalidade da distribuição das variáveis dependentes

Para dar prosseguimento à análise dos dados, um passo importante é verificar se as variáveis dependentes possuem uma distribuição do tipo normal. Neste trabalho, geramos gráficos do tipo Q-Q para uma análise visual da normalidade das variáveis e escolhemos o método de Shapiro-Wilk para confirmarmos ou não a normalidade.

Após a aplicação deste método, concluímos que somente a variável dependente %UEQ (*percentage of unordered elements quantity*) possuía uma distribuição normal em relação à média para todos os algoritmos, sendo que isto ocorreu para toda combinação entre as variáveis independentes *Sorting algorithm*, *Probability of failure* e *Array size*. Com base nestes resultados, escolhemos testar somente as hipóteses referentes variável %UEQ, já que utilizamos o método ANOVA para o teste de hipótese e tal método tem como premissa a distribuição normal dos valores das variáveis. A seguir alguns exemplos dos resultados do teste de Shapiro-Wilk e dos gráficos Q-Q de onde tiramos tais conclusões.

Exemplo-01: probabilidade de erro **0.01** e tamanho do array **100**:

* Variável: %LSS(*percentage of largest subarray size*)
  + Shapiro-Wilk
    - BUBBLE : W = 0.854840 / p\_value = 0.0007
    - INSERTION: W = 0.961790 / p\_value = **0.3439**
    - MERGE : W = 0.937173 / p\_value = **0.0763**
    - QUICK : W = 0.869708 / p\_value = 0.0016
  + Gráficos Q-Q

Uma imagem contendo texto, mapa

Descrição gerada automaticamente

* Variável: %UEQ (*percentage of unordered elements quantity*)
  + Shapiro-Wilk
    - BUBBLE : W = 0.934980 / p\_value = **0.0666**
    - INSERTION: W = 0.949881 / p\_value = **0.1678**
    - MERGE : W = 0.936667 / p\_value = **0.0739**
    - QUICK : W = 0.936049 / p\_value = **0.0712**
  + Gráficos Q-Q

Uma imagem contendo texto, mapa, água

Descrição gerada automaticamente

Exemplo-02: probabilidade de erro **0.05** e tamanho do array **1000**:

* Variável: %LSS(*percentage of largest subarray size*)
  + Shapiro-Wilk
    - BUBBLE : W = 0.891642 / p\_value = 0.0052
    - INSERTION: W = 0.921918 / p\_value = 0.0300
    - MERGE : W = 0.878308 / p\_value = 0.0025
    - QUICK : W = 0.917583 / p\_value = 0.0232
  + Gráficos Q-Q

Uma imagem contendo texto, mapa, água, céu

Descrição gerada automaticamente

* Variável: %UEQ (*percentage of unordered elements quantity*)
  + Shapiro-Wilk
    - BUBBLE : W = 0.936459 / p\_value = **0.0730**
    - INSERTION: W = 0.931681 / p\_value = **0.0544**
    - MERGE : W = 0.961920 / p\_value = **0.3465**
    - QUICK : W = 0.963954 / p\_value = **0.3892**
  + Gráficos Q-Q

Uma imagem contendo texto, mapa

Descrição gerada automaticamente

## Teste das Hipóteses

Para os testes das hipóteses, foram definidos um grau de confiança de 95%. Conforme mencionado no tópico anterior, somente foram testadas as hipóteses referentes à variável dependente %UEQ (*percentage of unordered elements quantity*), sendo elas:

* **Hypothesis 1**: For a given probability of failure and array size, tested algorithms will produce a different percentage of unordered elements quantity.
* **Hypothesis 3**: For each algorithm, the array size and probability of failure have a significative impact on the percentage of unordered elements quantity.

Como nosso conjunto de dados contem 03 variáveis independentes, utilizamos o método ANOVA para simplificar os testes de hipótese, pois este método permite que vários grupos de variáveis sejam comparadas ao mesmo tempo.

### **Testando a Hipótese-1**

Para cada combinação das variáveis *Probability of failure* e *Array size*, foram formulados os seguintes testes:

* **H0** = Não houve diferença significativa entre os diferentes algoritmos em relação à variável %UEQ.
* **H1** = Houve diferença significativa entre os diferentes algoritmos em relação à variável %UEQ.

Resultados dos testes:

* Probabilidade = 0.01 e Tamanho = 100

sum\_sq df F PR(>F)

algoritmo 1174.466667 3.0 171.368721 **1.845762e-42**

Residual 265.000000 116.0

* Probabilidade = 0.01 e Tamanho = 1000

sum\_sq df F PR(>F)

algoritmo 1635.180250 3.0 1817.615579 **2.610034e-97**

Residual 34.785667 116.0

* Probabilidade = 0.01 e Tamanho = 10000

sum\_sq df F PR(>F)

algoritmo 1948.204989 3.0 15831.708335 **2.332995e-151**

Residual 4.758210 116.0

* Probabilidade = 0.02 e Tamanho = 100

sum\_sq df F PR(>F)

algoritmo 1106.5 3.0 88.617785 **7.054900e-30**

Residual 482.8 116.0

* Probabilidade = 0.02 e Tamanho = 1000

sum\_sq df F PR(>F)

algoritmo 2286.83025 3.0 1990.144336 **1.507899e-99**

Residual 44.43100 116.0

* Probabilidade = 0.02 e Tamanho = 10000

sum\_sq df F PR(>F)

algoritmo 3081.306943 3.0 18442.105239 **3.406264e-155**

Residual 6.460427 116.0

* Probabilidade = 0.05 e Tamanho = 100

sum\_sq df F PR(>F)

algoritmo 2274.291667 3.0 225.58173 **3.104916e-48**

Residual 389.833333 116.0

* Probabilidade = 0.05 e Tamanho = 1000

sum\_sq df F PR(>F)

algoritmo 3704.037583 3.0 1202.21591 **3.625400e-87**

Residual 119.132333 116.0

* Probabilidade = 0.05 e Tamanho = 10000

sum\_sq df F PR(>F)

algoritmo 4948.821397 3.0 19487.120298 **1.402080e-156**

Residual 9.819533 116.0

A partir dos resultados, obtivemos confiança suficiente para rejeitar a hipótese nula (H0) para todas as combinações dos testes. Portanto, podemos afirmar com 95% de confiança de que houve uma diferença significativa entre os diferentes algoritmos em relação aos valores médios registrados para a variável %UEQ.

### **Testando a Hipótese-3**

Para cada algoritmo foram formulados os seguintes testes:

* **H0** = Os valores das variáveis *Array size* e *Probability of failure* não impactaram significativamente os valores da variável %UEQ.
* **H1** = Os valores das variáveis *Array size* e *Probability of failure* impactaram significativamente os valores da variável %UEQ..

Resultados dos testes:

* Algoritmo BubbleSort

sum\_sq df F PR(>F)

size\_of\_array 0.019548 1.0 0.01350 **0.90759**

probabilidade\_erro 3590.644437 1.0 2479.71637 **7.570027e-137**

Residual 385.169623 266.0

* Algoritmo MergeSort

sum\_sq df F PR(>F)

size\_of\_array 1784.873256 1.0 383.585800 **1.702008e-53**

probabilidade\_erro 13009.963175 1.0 2795.961630 **3.801570e-143**

Residual 1237.731651 266.0

* Algoritmo QuickSort

sum\_sq df F PR(>F)

size\_of\_array 121.551362 1.0 32.964196 **2.556033e-08**

probabilidade\_erro 5626.955128 1.0 1526.005558 **3.457996e-112**

Residual 980.841817 266.0

* Algoritmo InsertionSort

sum\_sq df F PR(>F)

size\_of\_array 309.843991 1.0 75.905868 **3.236586e-16**

probabilidade\_erro 6806.075083 1.0 1667.358570 **1.414758e-116**

Residual 1085.798822 266.0

A partir dos resultados, obtivemos confiança suficiente para rejeitar a hipótese nula (H0) para os algoritmos MergeSort, QuickSort e InsertionSort. Portanto, podemos afirmar com grau de 95% de confiança de que os valores das variáveis independentes *Array size* e *Probability of failure* impactaram significativamente os valores da variável %UEQ.

Já para o algoritmo BubbleSort, não obtivemos confiança suficiente para rejeitar totalmente a hipótese nula (H0). Analisando o resultado, podemos afirmar com um grau de 95% de confiança de que o valor da variável independente *Probability of failure* impactou significativamente a variável dependente %UEQ. Mas não podemos afirmar o mesmo em relação à variável independente *Array size*. Isso demonstra que para o algoritmo BubbleSort os tamanhos dos arrays utilizados neste experimento não foram determinantes para definirem os valores referentes ao percentual de elementos desornados obtidos.

## Análise de desempenho dos algoritmos em relação à variável %UEQ

Nesta seção descrevemos uma análise do desempenho geral de cada algoritmo em relação à variável %UEQ (*percentage of unordered elements quantity*). Para auxiliar nesta análise, foram gerados três gráficos de barras, sendo um para cada valor de *Probability of failure*, os quais trazem informações sobre os valores da variável %UEQ agrupados pelas variáveis *Array size* e *Sorting algorithm*. Estes gráficos estão exibidos nas três figuras a seguir. Cada barra apresenta o valor médio amostral da variável %UEQ dentro de cada grupo. Acima de cada barra, existe uma linha vertical a qual indica os limites inferior e superior do intervalo de confiança para um nível de significância de 5% (α=0.05).

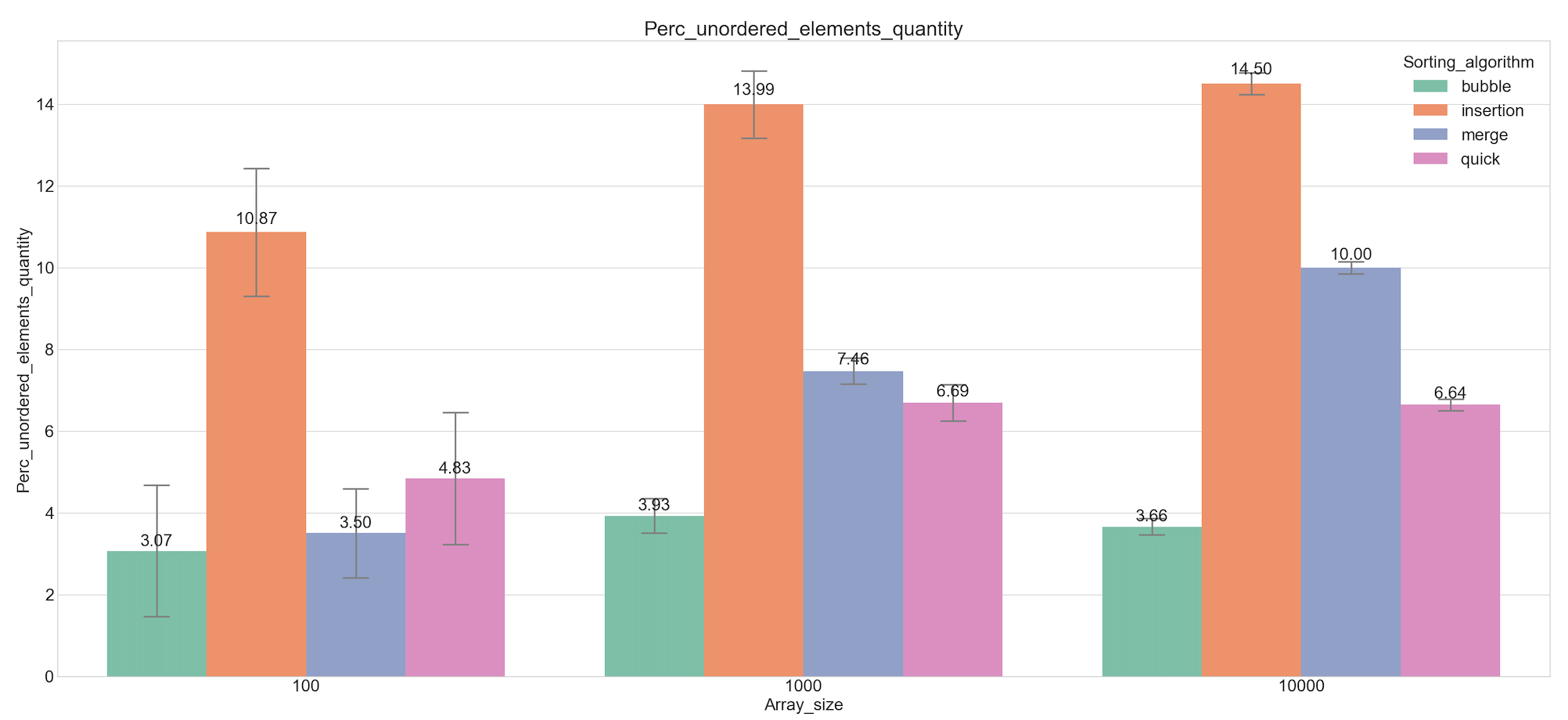


Figura : Dados referentes à Probability of failure = 0.01

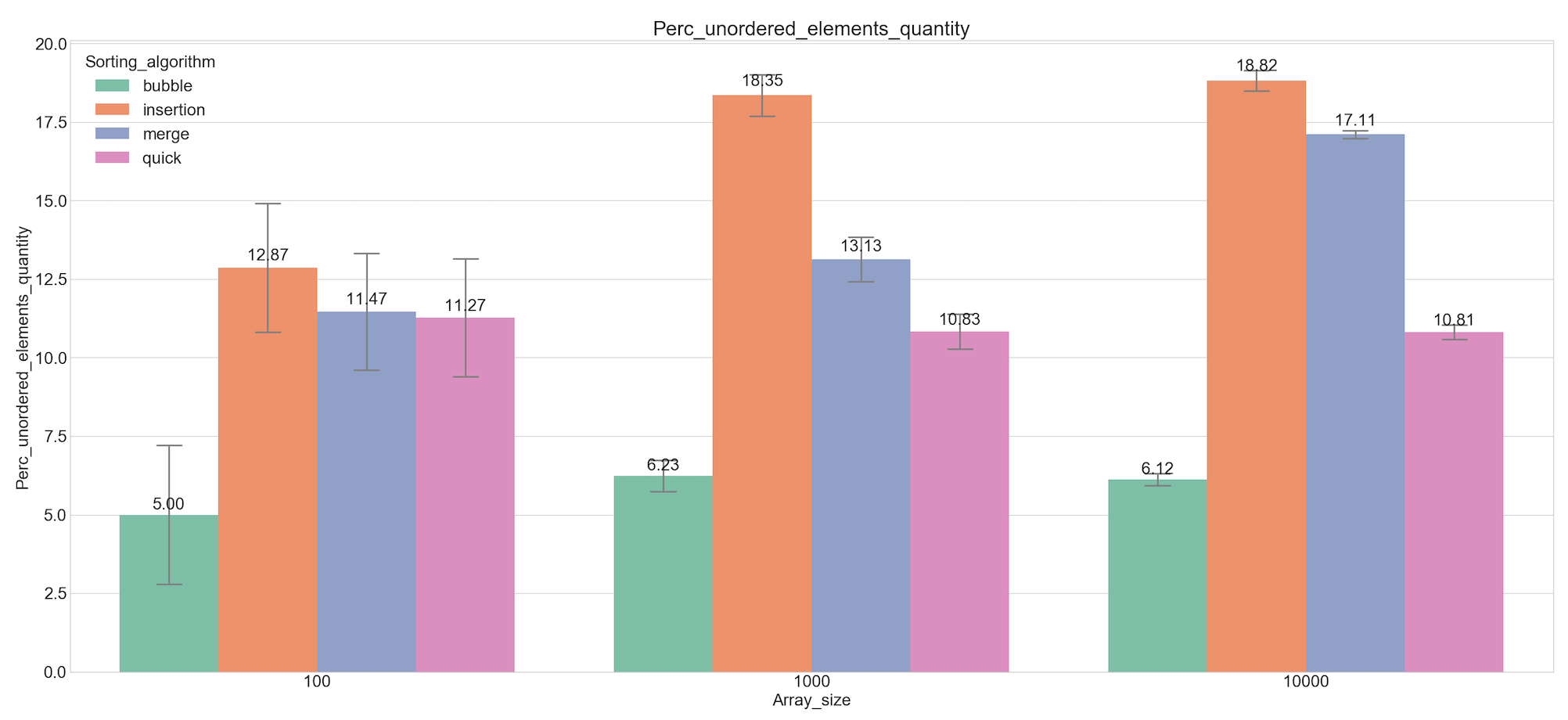


Figura : Dados referentes à Probability of failure = 0.02

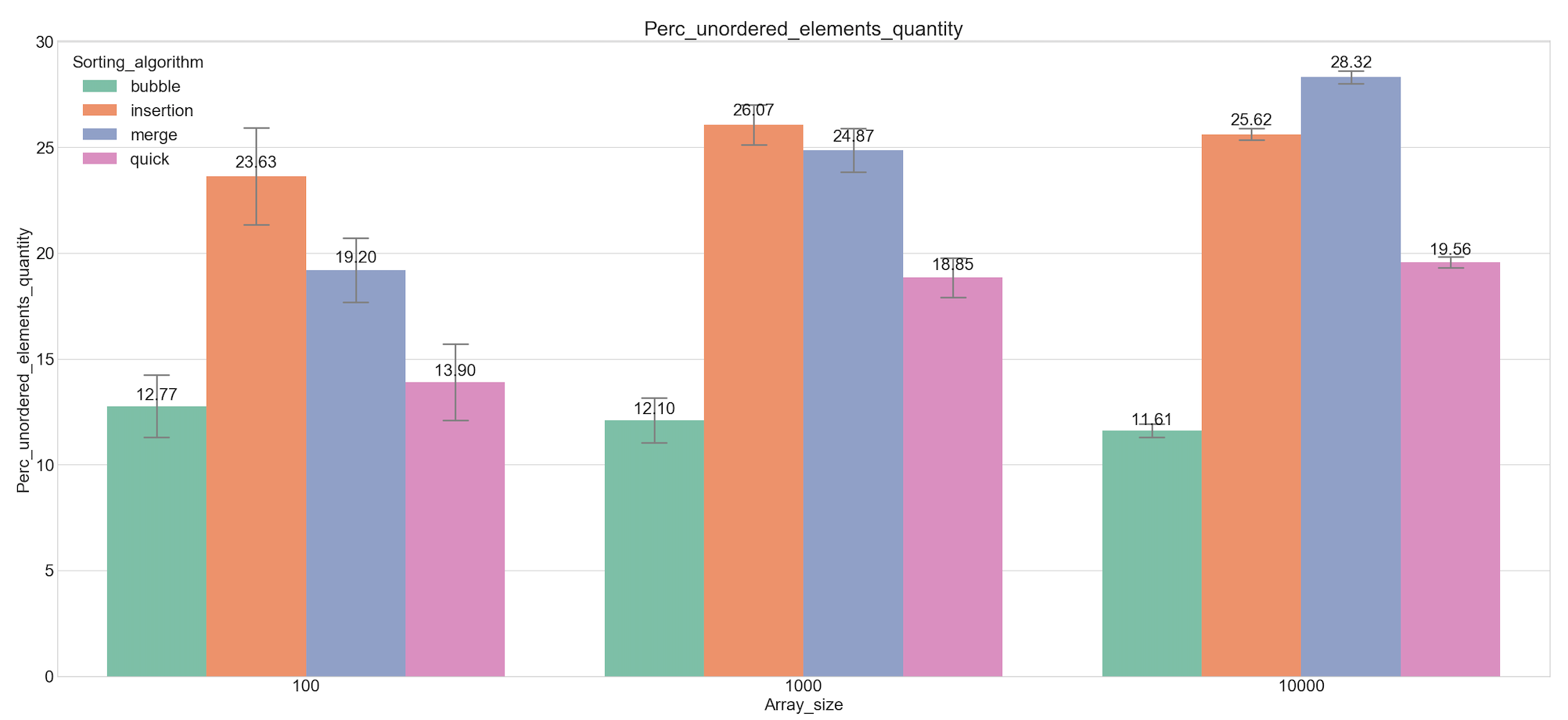


Figura : Dados referentes à Probability of failure = 0.05

Com base nas informações dos gráficos, podemos observar claramente que o algoritmo BubbleSort gerou um menor valor médio para a variável %UEQ para todas as combinações definidas (*Probability of failure* X *Array size* X *Sorting algorithm*). Isso demonstra que o BubbleSort foi o algoritmo menos afetado pelas falhas simuladas neste experimento. Acreditamos que tal fato se deve ao próprio funcionamento deste algoritmo, onde a ordem de todos elementos são verificados a cada interação, fazendo com que um elemento posicionado incorretamente em uma interação (por causa de uma falha de memória) possa ter seu local corrigido numa próxima interação. Ainda em relação ao BubbleSort, podemos identificar que apesar dos diferentes tamanhos do array de entrada (100, 1000 e 10000), não houve um impacto significativo no valor médio obtido para a variável %UEQ, comprovando assim o que foi identificado nos testes da *Hipótese-3* para este algoritmo, onde utilizamos o método ANOVA.

Concluímos ainda que, em nosso experimento o algoritmo que apresentou o pior desempenho em relação à variável %UEQ foi o InsertionSort. Este algoritmo obteve as maiores médias em quase todas as configurações, sendo que esta média foi muito superior aos dos outros algoritmos nos testes realizados com *Probability of failure* = 0.01. Por fim, os testes mostraram que no geral o algoritmo QuickSort obteve um desempenho superior ao do MergeSort.

Podemos então definir um ranking de melhor desempenho destes algoritmos durante este experimento como sendo o seguinte:

1. BubbleSort
2. QuickSort
3. MergeSort
4. InsertionSort