**Descrição das Soluções aos Problemas**

No problema 1, optámos, por uma solução dinâmica, que fosse armazenando valores para evitar o uso de recursões desnecessárias. Assim, a nossa implementação itera pelos elementos da sequência do início para o fim e, para cada um deles, itera pelos elementos anteriores, também do início para o fim, sem exceções ou saídas de ciclos. São criados 2 vetores da mesma dimensão da sequência: o **lss** (*longest subsequence size*) e o **quant** (*quantity*). O primeiro, como o nome indica, mantém o tamanho da maior subsequência estritamente crescente que contém o elemento do índice considerado, e o segundo regista a quantidade de sequências desse mesmo tamanho que contêm o elemento.

Por cada iteração pelos elementos da sequência, é atualizada uma variável que contém o maior elemento do **lss**, isto é, o tamanho da maior subsequência que queremos, e é também atualizado o valor da soma dos elementos de `quant` cujos índices correspondem aos índices que contém esse tal tamanho em **lss**, isto é, a soma do número total de sequências de tamanho máximo.

No que toca ao problema 2, por cada elemento da sequência 1, iterámos também por todos os elementos da sequência 2 do início para o fim, sem exceções ou saídas dos ciclos, o que nos garante que estamos na presença de um limite assimptótico apertado de grandeza n2. Foi utilizada uma tabela da mesma dimensão da sequência 2 que, para cada índice, guardasse o tamanho da maior subsequência estritamente crescente comum entre as 2 sequências que acabasse nesse elemento da sequência.

O maior valor da tabela, resultado que queremos, vai sendo atualizado a cada iteração, numa variável. Será previsível, à partida, que haverá índices da tabela que irão conter 0 (os índices dos elementos da 2ª sequência que não estão presentes na 1ª). Esta solução foi, então, posteriormente otimizada com recurso a um pré-processamento que, na leitura da 2ª linha para o vetor, apenas escrevesse os elementos que também estivessem presentes na sequência anterior. Com recurso a um *set* desses elementos, fizemos com que a procura fosse O(1), na maioria dos casos.

Pseudocódigo de partes das lógicas por detrás das soluções apresentadas:

Problema 1: Problema 2:

**for** i=0 to seq.size()-1 **do** **for** i=0 to seq1.size()-1 **do**

lis[i] = quant[i] = 1 curr = 0

**for** j=0 to i **do** **for** j=0 to seq2.size()-1 **do**

**if** seq[j] < seq[i] **then**  **if** seq2[j] < seq1[i] **then**

**if** lis[j] + 1 > lis[i] **then**  curr = max(curr, lcis[j])

lis[i] = lis[j] + 1 **if** seq2[j] == seq1[i] **then**

quant[i] = quant[j] lcis[j] = max(curr+1, lcis[j])

**if** lis[j] + 1 == lis[i] **then** res = max(lcis[j], res)

quant[i] += quant[j] **end if**

**end if end for**

**end if end for**

**end for**

**end for**

**Análise Teórica**

Problema 1:

* Leitura da sequência para vetor; respetivamente: O(n)
* Algoritmo que percorre os elementos anteriores da seq. por cada elemento: O(n2)
* Apresentação dos dados, o output retirado das tabelas criadas: O(1)

Complexidade de tempo: O(n2), complexidade de espaço: O(n)

Problema 2:

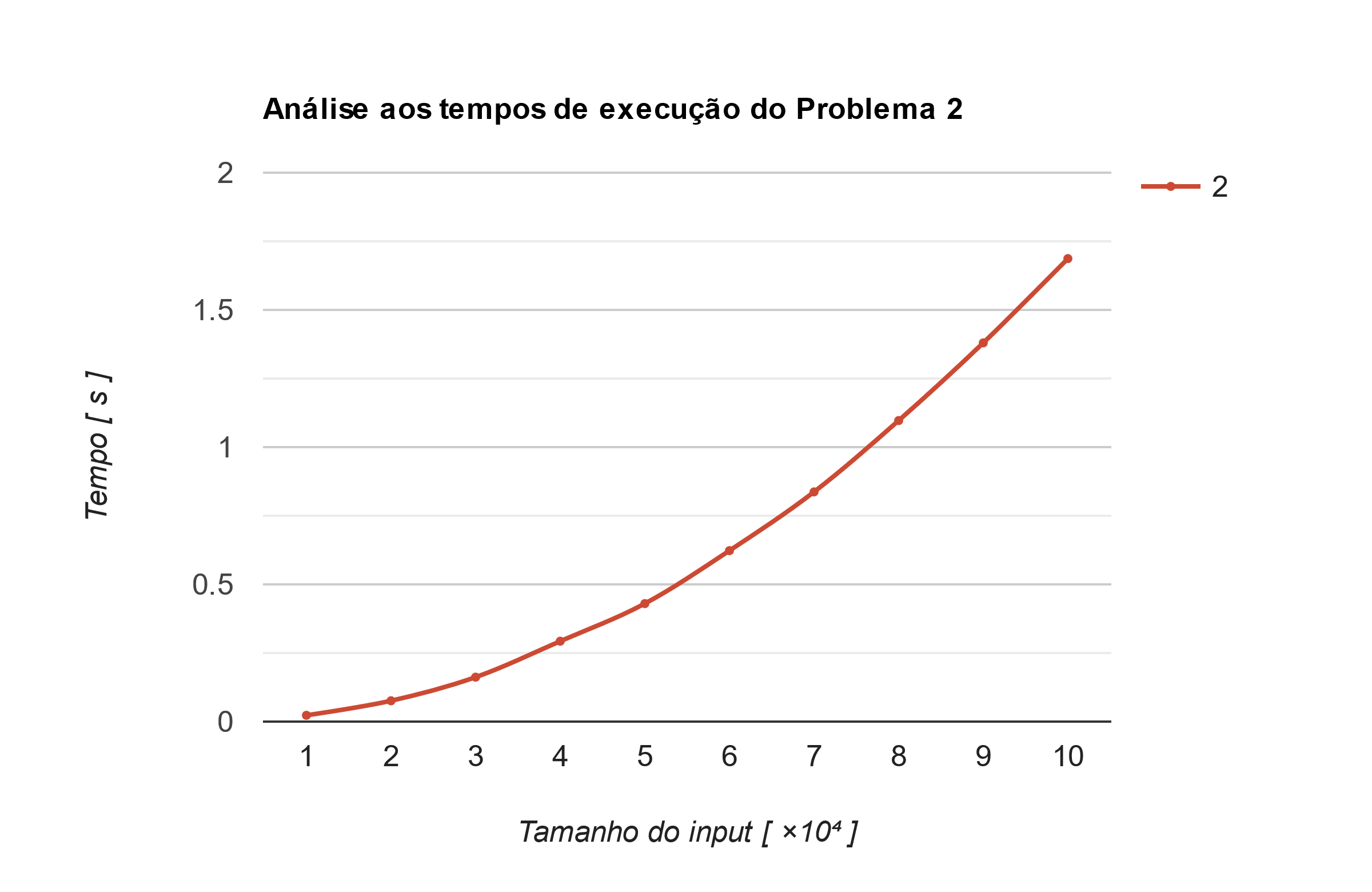
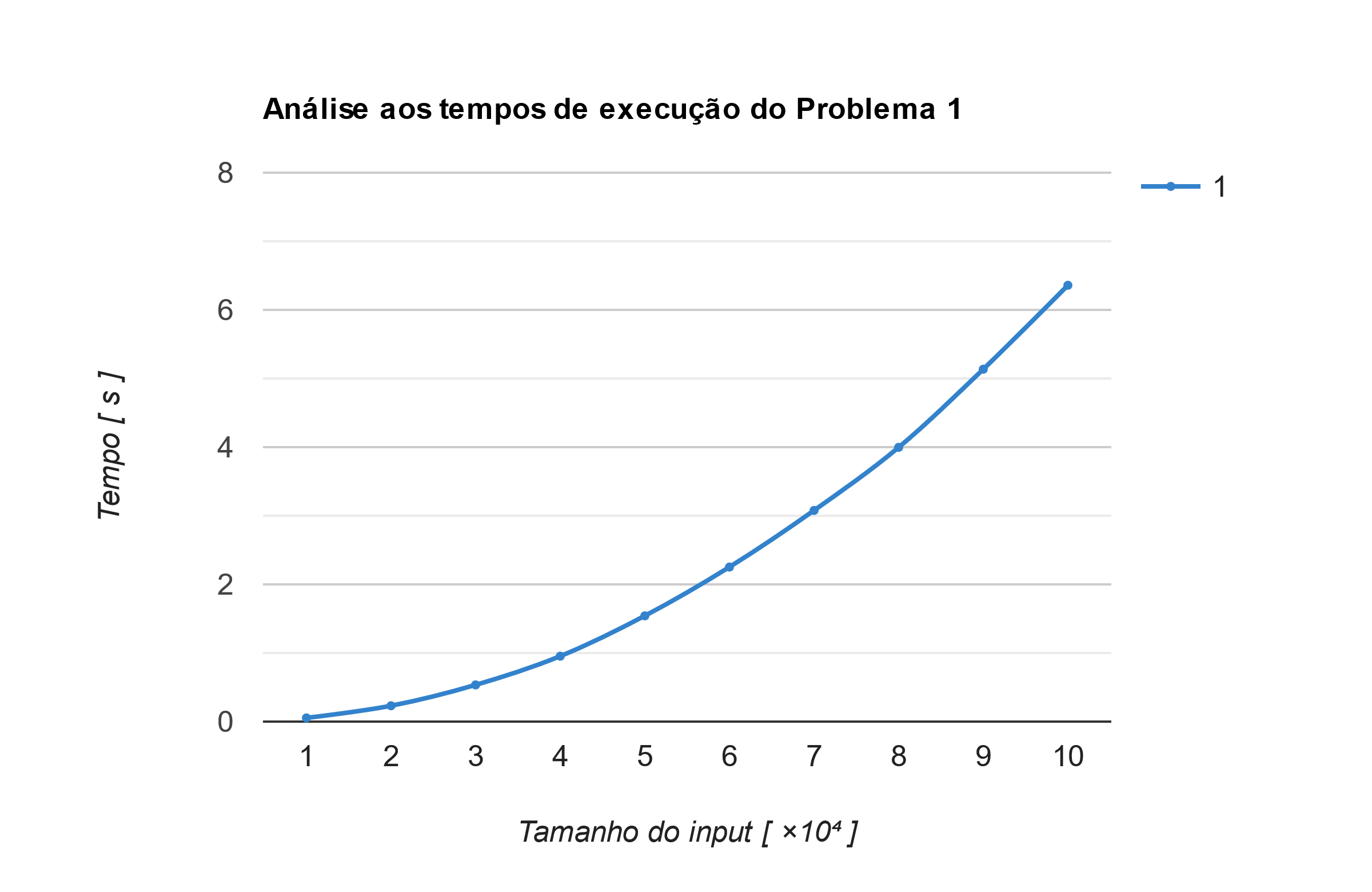
* Leitura da 1ª e 2ª sequência para vetores, respetivamente: O(m), O(n)
* Algoritmo que percorre os elementos da 2ª seq. por cada elemento da 1ª: Θ(m×n)
* Apresentação dos dados, o output retirado da tabela criada: O(1)

Complexidade de tempo: Θ(m×n), complexidade de espaço: O(n).

**Avaliação Experimental dos Resultados**

A análise aos tempos de execução foi realizada com base em gráficos com tempos associados. Para cada problema, foram registados 10 resultados, um para cada tamanho da sequência de *input* e de forma incremental, desde 1×104 até 1×105, de 10000 em 10000, e calculados os tempos para cada instância. No 2º problema, ambas as sequências dadas têm o mesmo tamanho.

Para cada tamanho, foram realizados 50 testes com inputs sempre diferentes com o **random\_k** e calculada a média, com recurso à ferramenta **hyperfine**. O tempo encontra-se em segundos (eixo dos YYs) e o tamanho na potência 104 (eixo dos XXs).



Recordando a complexidade das soluções dos 2 problemas, as curvas dos gráficos gerados estão, ambas, de acordo com aquilo que seria de esperar de um algoritmo com limite assimptótico apertado de grandeza n2.