Emparelhamento Exercícios

Laboratório de Programação Competitiva - 2020

Pedro Henrique Paiola

Array and Operations (CodeForces 498C)

- **Dados:**
  - Vetor de inteiros positivos: a[1], a[2], ..., a[n]
  - “Bons” pares: (i_1,j_1), ..., (i_m,j_m) em que i_k+j_k é ímpar e $i_k < j_k$
- Para cada par (i, j), podemos escolher algum inteiro v (v > 1) que divide ambos os números a[i] e a[j], e então realizar esta divisão: a[i] = a[i]/v e a[j] = a[j]/v
- **Objetivo:** determinar o maior número de operações que podem ser executadas sequencialmente. Cada par pode ser utilizado mais de uma vez

Array and Operations (CodeForces 498C)

- A condição de que i + j é ímpar nos garante que um destes números é par e o outro é ímpar. Isso nos permite criar um grafo bipartido, onde os números pares ficam à esquerda e os ímpares à direita (por exemplo), e as arestas vão ligar apenas vértices de partições diferentes.
- Mas isso por si só não é suficiente para modelar o exercício como um problema de emparelhamento máximo.

Array and Operations (CodeForces 498C)

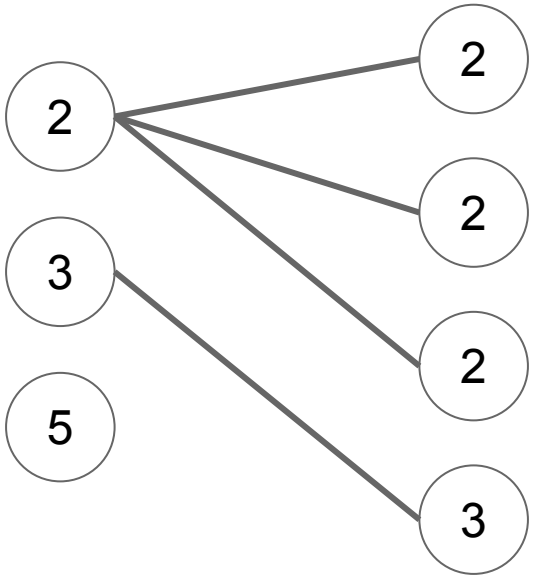
- Vamos considerar, a princípio, que temos apenas dois números e um par que permite realizar a operação descrita entre eles.
- Se queremos MAXIMIZAR o número de operações, podemos verificar quantos fatores primos em comum eles têm.
- Exemplo: 30 e 24

  30 => 2*3*5
  24 => 2*2*2*3

Neste caso, podemos dividir os números 30 e 24 duas vezes (uma vez por 2 e uma vez por 3)

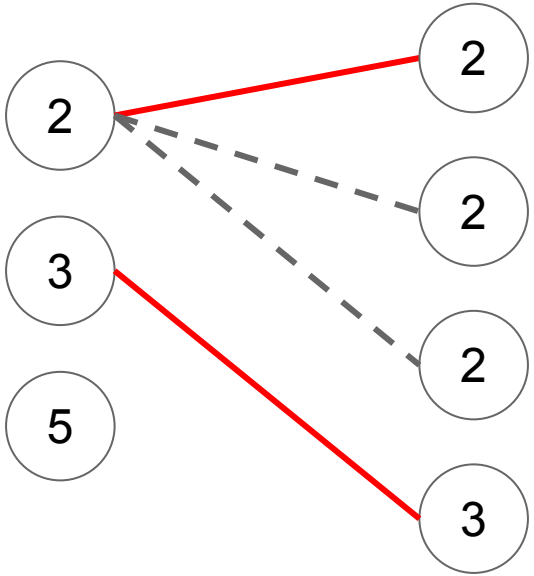
Array and Operations (CodeForces 498C)

- Isto pode ser representado por um grafo. Para cada número a[k] teremos um grupo de vértices, um vértice para cada um seus fatores (incluindo repetições). Se temos um par (i, j), então ligaremos por arestas todos os fatores iguais de a[i] e a[j].



Array and Operations (CodeForces 498C)

- Aplicando o emparelhamento máximo, vamos obter a melhor combinação possível de fatores a serem usados efetivamente das divisões, de forma a fazer o maior número de operações possíveis.



Weird Fence (UVA 11262)

- **Entrada**: posições de postes azuis e vermelhos.
- **Objetivo**: determinar a distância mínima de correntes de forma que pelo menos k postes sejam conectados. Cada corrente deve conectar postes de cores diferentes.

Weird Fence (UVA 11262)

- A restrição de que uma corrente só pode conectar postes de cores diferentes permite representar o problema por um grafo bipartido.
- Para um tamanho de corrente **x** podemos determinar, aplicando algum algoritmo de emparelhamento máximo, quantos postes conseguimos conectar. Uma forma de fazer isso é colocar no grafo apenas as arestas que conectam dois postes com distância menor ou igual a **x**.
- Para descobrir o menor **x** que permite fazer pelo menos **k** conexões de forma eficiente podemos aplicar uma busca binária.