AULA 24

Complex. computacional: P versus NP

CLR 36 ou CLRS 34

Complexidade computacional

Classifica os problemas em relação à dificuldade de resolvê-los algoritmicamente.

Disciplinas:

MAC0430 Algoritmos e Complexidade de Computação

MAC5722 Complexidade Computacional

Palavras

Para resolver um problema usando um computador é necessário descrever os dados do problema através de uma seqüência de símbolos retirados de algum alfabeto.

Este alfabeto pode ser, por exemplo, o conjunto de símbolos ASCII ou o conjunto $\{0, 1\}$.

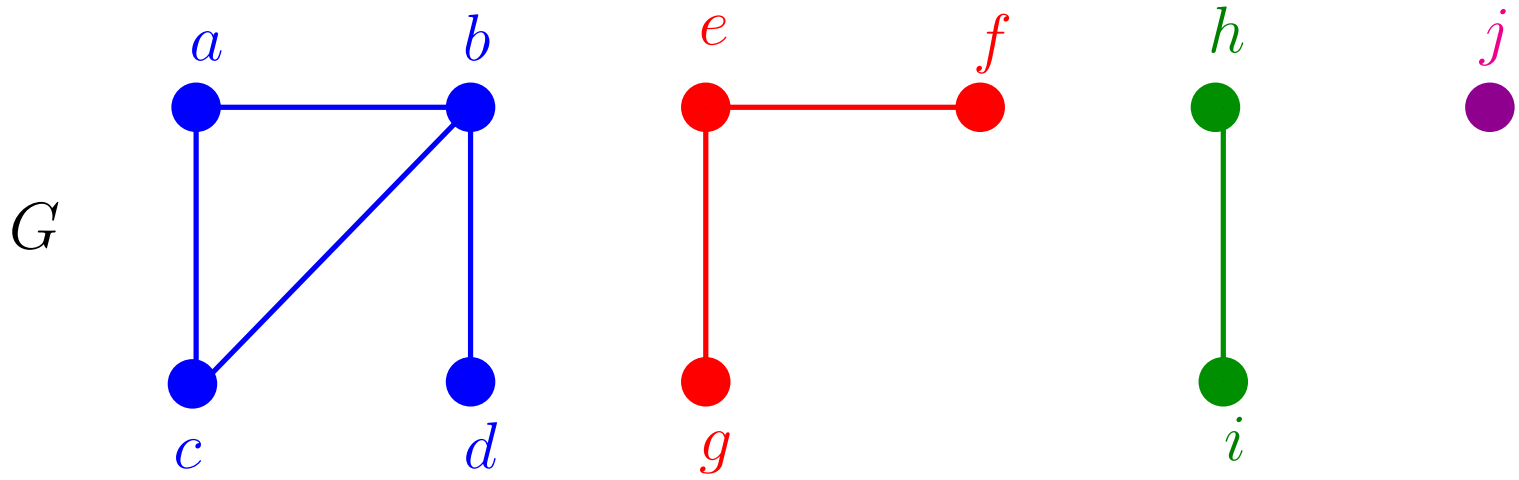
Qualquer seqüência dos elementos de um alfabeto é chamada de uma palavra.

Não é difícil codificar objetos tais como racionais, vetores, matrizes, grafos e funções como palavras.

O tamanho de uma palavra w, denotado por $\langle w \rangle$ é o número de símbolos usados em w, contando multiplicidades. O tamanho do racional '$123/567$' é 7.

Exemplo 1

Grafo



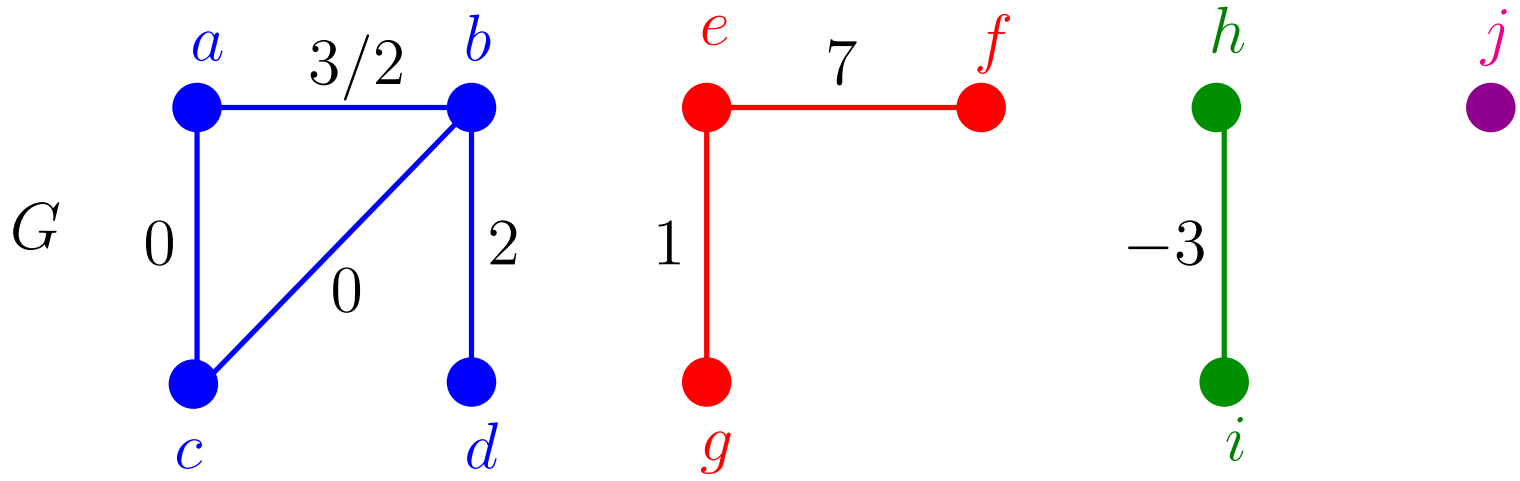
Palavra:

$$(\{a, b, c, d, e, f, g, h, i, j\}, \{\{bd\}, \{eg\}, \{ac\}, \{hi\}, \{ab\}, \{ef\}, \{bc\}\}$$

Tamanho da palavra: 59

Exemplo 2

Função



Palavra:

$$((\{bd\}, 2), (\{eg\}, 1), (\{ac\}, 0), (\{hi\}, -3), (\{ab\}, 3/2), (\{ef\}, 7), (\{bc, 0)\})$$

Tamanho da palavra: 67

Tamanho de uma palavra

Para os nossos propósitos, não há mal em subestimar o tamanho de um objeto.

Não é necessário contar rigorosamente os caracteres '{', '}', '(', ')' e ',' dos exemplos anteriores.

Tamanho de um inteiro p é essencialmente $\lg |p|$.

Tamanho do racional p/q é, essencialmente, $\lg |p| + \lg |q|$.

Tamanho de um vetor $A[1\,..\,n]$ é a soma dos tamanhos de seus componentes

$$\langle A \rangle = \langle A[1] \rangle + \langle A[2] \rangle + \cdots + \langle A[n] \rangle .$$

Problemas e instâncias

Cada conjunto específico de dados de um problema define uma instância.

Tamanho de uma instância é o tamanho de uma palavra que representa a instância.

Problema que pede uma resposta do tipo SIM ou NÃO é chamado de problema de decisão.

Problema que procura um elemento em um conjunto é um problema de busca.

Problema que procura um elemento de um conjunto de soluções viáveis que seja melhor possível em relação a algum critério é um problema de otimização

Máximo divisor comum

Problema: Dados dois números inteiros não-negativos a e b, determinar mdc(a, b).

Exemplo:
máximo divisor comum de 30 e 24 é 6
máximo divisor comum de 514229 e 317811 é 1
máximo divisor comum de 3267 e 2893 é 11

Problema de busca

Instância: a e b

Tamanho da instância: $\langle a \rangle + \langle b \rangle$, essencialmente

$$\lg a + \lg b$$

Consumo de tempo do algoritmo Café-Com-Leite é $O(b)$.
Consumo de tempo do algoritmo EUCLIDES é $O(\lg b)$.

Máximo divisor comum (decisão)

Problema: Dados dois números inteiros não-negativos a, b e k, mdc$(a, b) = k$?

Exemplo:
máximo divisor comum de 30 e 24 é 6
máximo divisor comum de 514229 e 317811 é 1
máximo divisor comum de 3267 e 2893 é 11

Problema de decisão: resposta SIM ou NÃO

Instância: a, b, k

Tamanho da instância: $\langle a \rangle + \langle b \rangle + \langle k \rangle$, essencialmente

$$\lg a + \lg b + \lg k$$

Subseqüência comum máxima

Problema: Encontrar uma ssco máxima de $X[1\,..\,m]$ e $Y[1\,..\,n]$.

Exemplos: X = A B C B D A B

Y = B D C A B A

ssco máxima = B C A B

Problema de otimização

Instância: $X[1\,..\,m]$ e $Y[1\,..\,n]$

Tamanho da instância: $\langle X\rangle + \langle Y\rangle$, essencialmente

$$n + m$$

Consumo de tempo REC-LCS-LENGTH é $\Omega(2^{\min\{m,n\}})$.
Consumo de tempo LCS-LENGTH é $\Theta(mn)$.

Subseqüência comum máxima (decisão)

Problema: $X[1\,..\,m]$ e $Y[1\,..\,n]$ possuem uma ssco máxima $\geq k$?

Exemplo: X = A B C B D A B

Y = B D C A B A

ssco máxima = B C A B

Problema de decisão: resposta SIM ou NÃO

Instância: $X[1\,..\,m]$, $Y[1\,..\,n]$, k

Tamanho da instância: $\langle X\rangle + \langle Y\rangle + \langle k\rangle$, essencialmente

$$n + m + \lg k$$

Problema booleano da mochila

Problema (Knapsack Problem): Dados n, $w[1\,..\,n]$ $v[1\,..\,n]$ e W, encontrar uma mochila boolena ótima.

Exemplo: $W = 50$, $n = 4$

	1	2	3	4
w	40	30	20	10
v	840	600	400	100
x	0	1	1	0

valor $= 1000$

Problema de otimização

Instância: n, $w[1\,..\,n]$ $v[1\,..\,n]$ e W

Tamanho da instância: $\langle n \rangle + \langle w \rangle + \langle v \rangle + \langle W \rangle$, essencialmente $\lg n + n \lg W + n \lg V + \lg W$
Consumo de tempo MOCHILA-BOOLEANA é $\Theta(nW)$.

Problema booleano da mochila (decisão)

Problema (Knapsack Problem): Dados n, $w[1\,..\,n]$ $v[1\,..\,n]$ e W e k, existeuma mochila boolena de valor $\geq k$.

Exemplo: $W = 50$, $n = 4$, $k = 1010$

	1	2	3	4
w	40	30	20	10
v	840	600	400	100
x	0	1	1	0

valor $= 1000$

Problema de decisão: resposta SIM ou NÃO

Instância: n, $w[1\,..\,n]$ $v[1\,..\,n]$, W e k

Tamanho da instância: $\langle n\rangle + \langle w\rangle + \langle v\rangle + \langle W\rangle + \lg k$, essencialmente $\lg n + n \lg W + n \lg V + \lg W + \lg k$

Problema fracionário da mochila

Problema: Dados n, $w[1\,..\,n]$ $v[\,..\,n]$ e W, encontrar uma mochila ótima.

Exemplo: $W = 50$, $n = 4$

	1	2	3	4
w	40	30	20	10
v	840	600	400	100
x	1	1/3	0	0

valor $= 1040$

Problema de otimização

Instância: n, $w[1\,..\,n]$ $v[1\,..\,n]$ e W

Tamanho da instância: $\langle n\rangle + \langle w\rangle + \langle v\rangle + \langle W\rangle$, essencialmente $\lg n + n \lg W + n \lg V + \lg W$

Consumo de tempo MOCHILA-FRACIONÁRIA é $\Theta(n \lg n)$.

Problema fracionário da mochila (decisão)

Problema: Dados n, $w[1\,..\,n]$ $v[\,..\,n]$, W e k, existe uma mochila de valor $\geq k$?

Exemplo: $W = 50$, $n = 4$, $k = 1010$

	1	2	3	4
w	40	30	20	10
v	840	600	400	100
x	1	1/3	0	0

valor $= 1040$

Problema de decisão: resposta SIM ou NÃO

Instância: n, $w[1\,..\,n]$ $v[1\,..\,n]$ e W

Tamanho da instância: $\langle n\rangle + \langle w\rangle + \langle v\rangle + \langle W\rangle$, essencialmente $\lg n + n \lg W + n \lg V + \lg W$

Modelo de computação

É uma descrição abstrata e conceitual de um computador que será usado para executar um algoritmo.

Um modelo de computação especifica as operações elementares um algoritmo pode executar e o critério empregado para medir a quantidade de tempo que cada operação consome.

Operações elementares típicas são operações aritméticas entre números e comparações.

No critério uniforme supõe-se que cada operação elementar consome uma quantidade de tempo constante.

Problemas polinomiais

Análise de um algoritmo em um determinado modelo de computação estima o seu consumo de tempo e quantidade de espaço como uma função do tamanho da instância do problema.

Exemplo: o consumo de tempo do algoritmo EUCLIDES (a, b) é expresso como uma função de $\langle a \rangle + \langle b \rangle$.

Um problema é solúvel em tempo polinomial se existe um algoritmo que consome tempo $\mathrm{O}(\langle I \rangle^c)$ para resolver o problema, onde c é uma constante e I é instância do problema.

Exemplos

- Máximo divisor comum

  Tamanho da intância: $\lg a + \lg b$

  Consumo de tempo Café-Com-Leite é $\mathrm{O}(b)$ (não-polinomial)

  Consumo de tempo EUCLIDES é $\mathrm{O}(\lg b)$ (polinomial)

- Subseqüência comum máxima

  Tamanho da instância: $n + m$

  Consumo de tempo REC-LEC-LENGTH é $\Omega(2^{\min\{m,n\}})$ (não-polinomial)

  Consumo de tempo LEC-LENGTH é $\Theta(mn)$ (polinomial).

Mais exemplos

- Problema booleano da mochila

  Tamanho da instância: $\lg n + n \lg W + n \lg V + \lg W$
  Consumo de tempo MOCHILA-BOOLEANA é $\Theta(nW)$
  (não-polinomial)

- Problema fracionário da mochila
  Tamanho da instância: $\lg n + n \lg W + n \lg V + \lg W$
  Consumo de tempo MOCHILA-FRACIONÁRIA é
  $\Theta(n \lg n)$
  (polinomial).

- Ordenação de inteiros $A[1\,..\,n]$

  Tamanho da instância: $n \lg M$,
  $M := \max\{|A[1]|, |A[2]|, \ldots, |A[n]|\} + 1$

  Consumo de tempo MERGE-SORT é $\Theta(n \lg n)$
  (polinomial).

Classe P

Por um algoritmo eficiente entendemos um algoritmo polinomial.

A classe de todos os problemas de decisão que podem ser resolvidos por algoritmos polinomiais é denotada por P (classe de complexidade).

Exemplo: As versões de decisão dos problemas:

> máximo divisor comum, subseqüência comum máxima e mochila fracionária

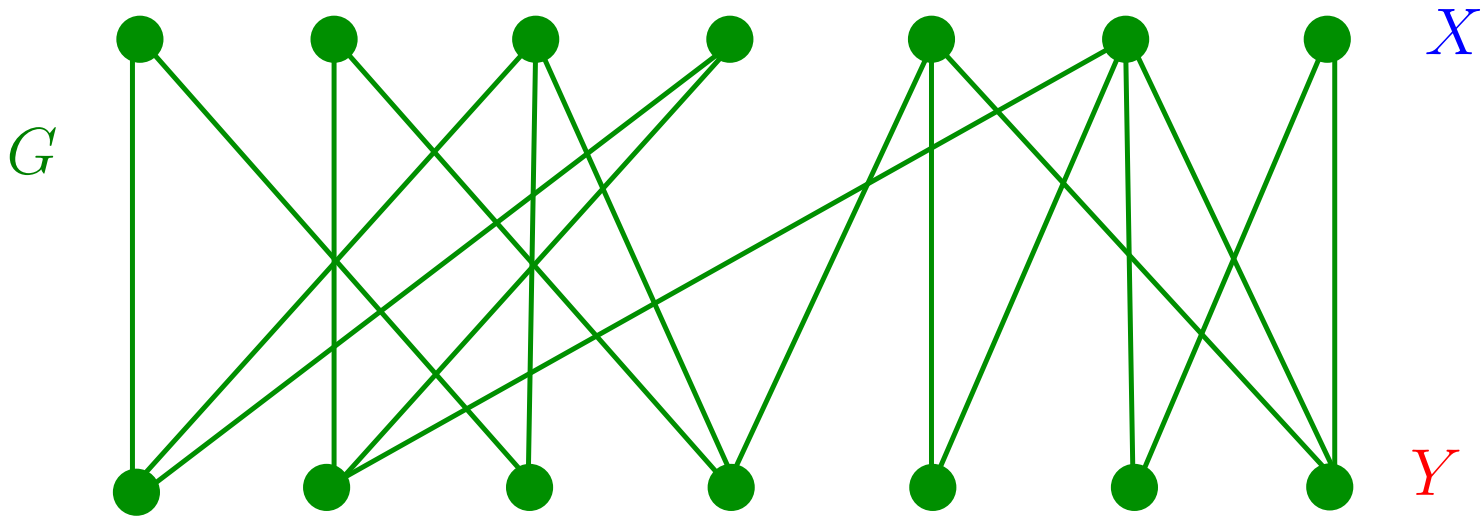
estão em P.

Para muitos problemas, não se conhece algoritmo essencialmente melhor que "testar todas as possibilidades". Em geral, isso não está em P.

Arthur e Merlin

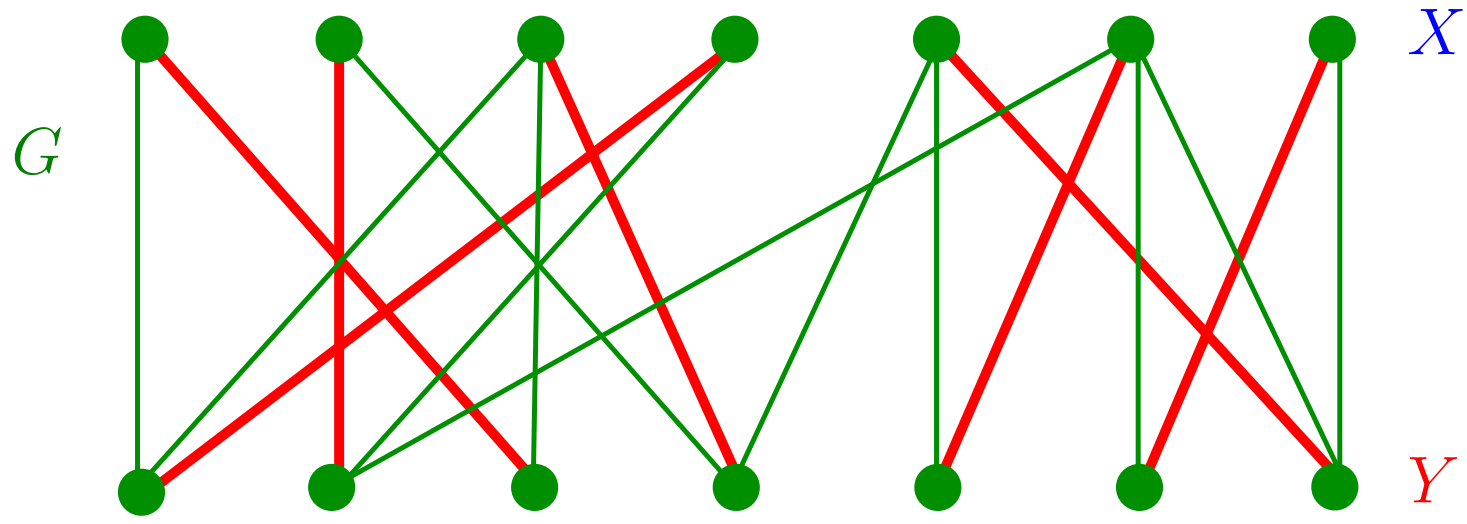
Emparelhamentos

Problema: Dado um grafo bipartido encontrar um emparelhamento perfeito.



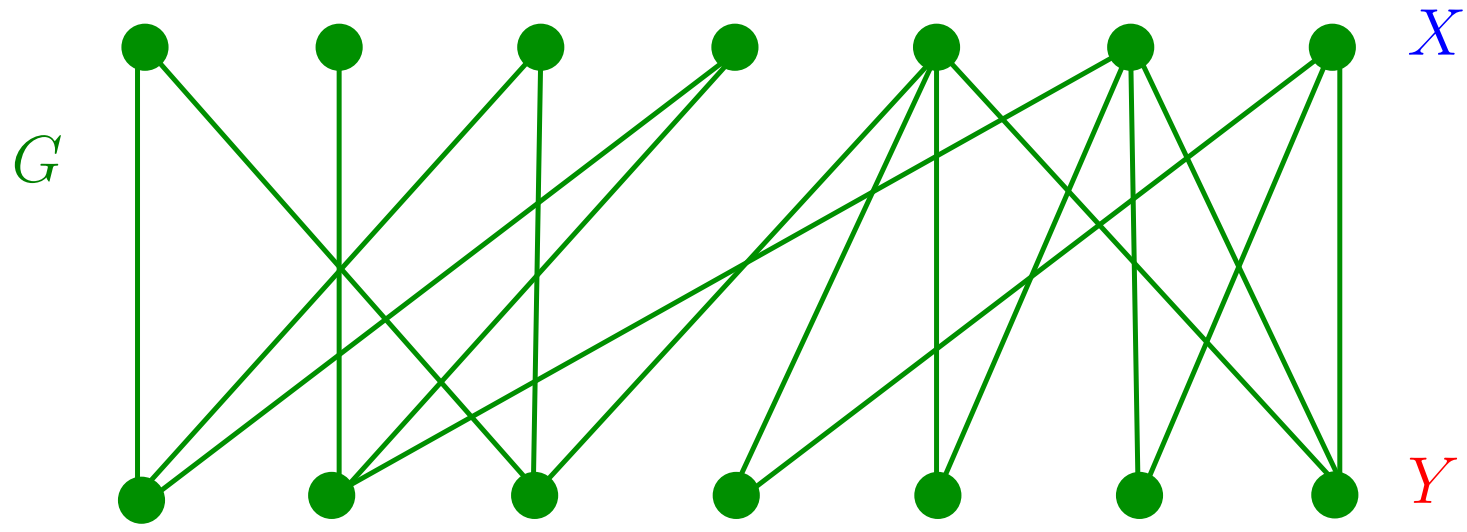
Emparelhamentos

Problema: Dado um grafo bipartido encontrar um emparelhamento perfeito.



Emparelhamentos

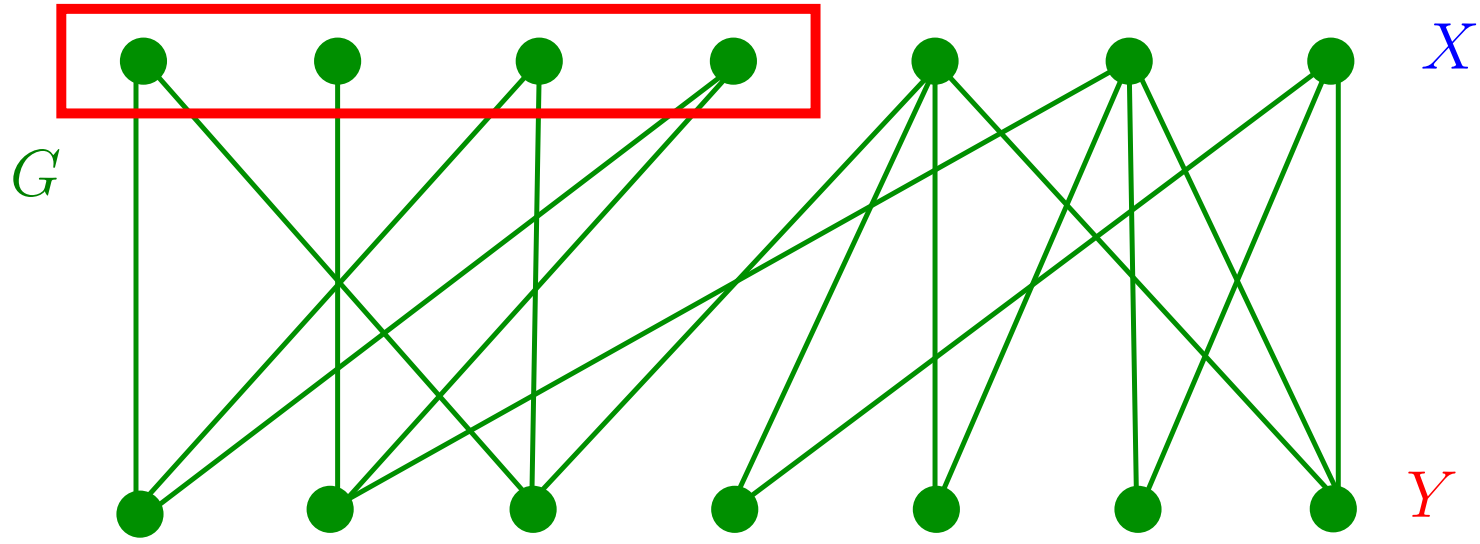
Problema: Dado um grafo bipartido encontrar um emparelhamento perfeito.



NÃO existe! Certificado?

Emparelhamentos

Problema: Dado um grafo bipartido encontrar um emparelhamento bipartido.



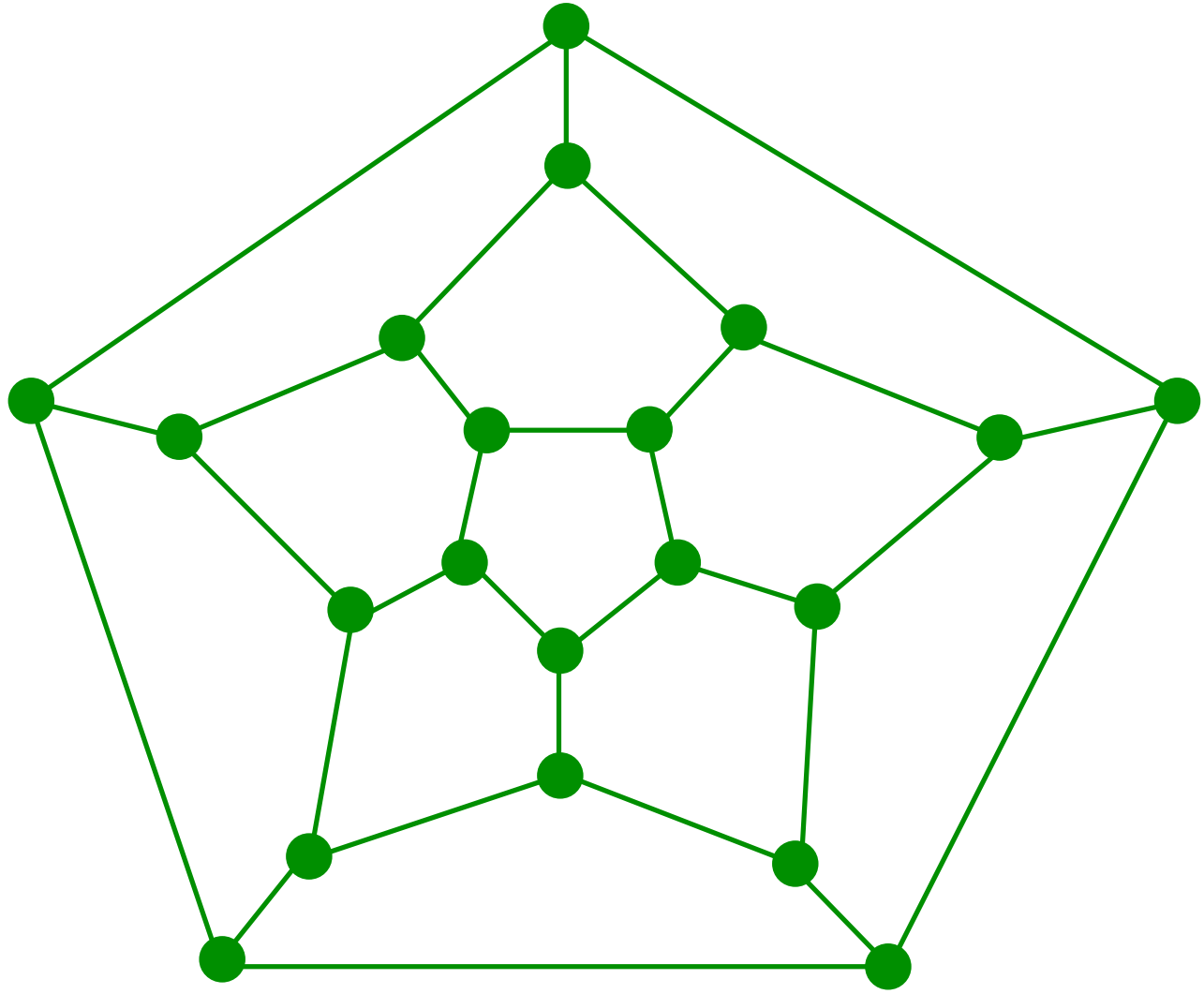
NÃO existe! Certificado: $S \subseteq X$ tal que $|S| > |\text{vizinhos}(S)|$.
Teorema de Hall: G tem um emparelhamento perfeito se e somente se

$$|S| \leq |\text{vizinhos}(S)|, \quad \text{para todo } S \subseteq X.$$

Grafos hamiltonianos

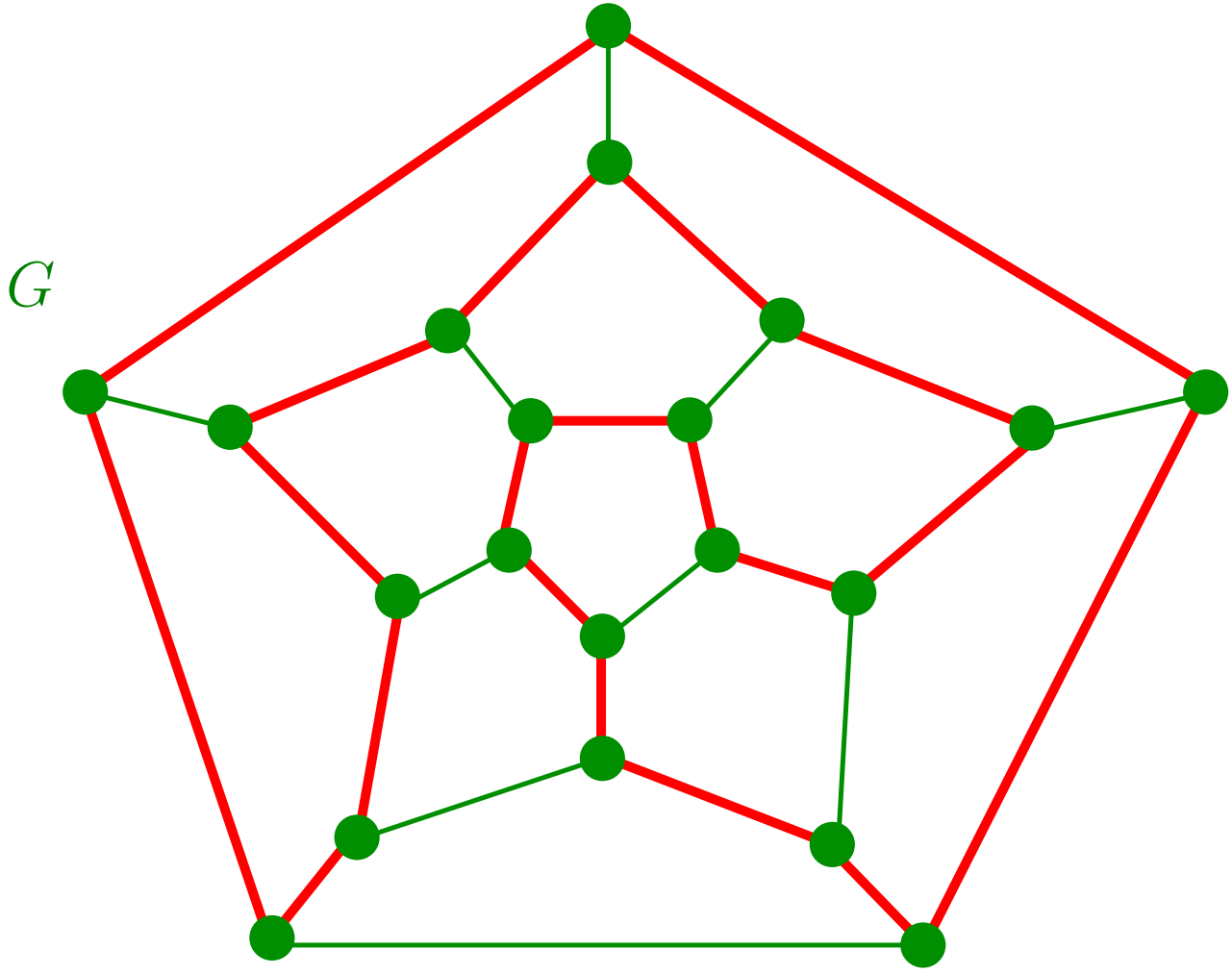
Problema: Dado um grafo encontrar um ciclo hamiltoniano.

G



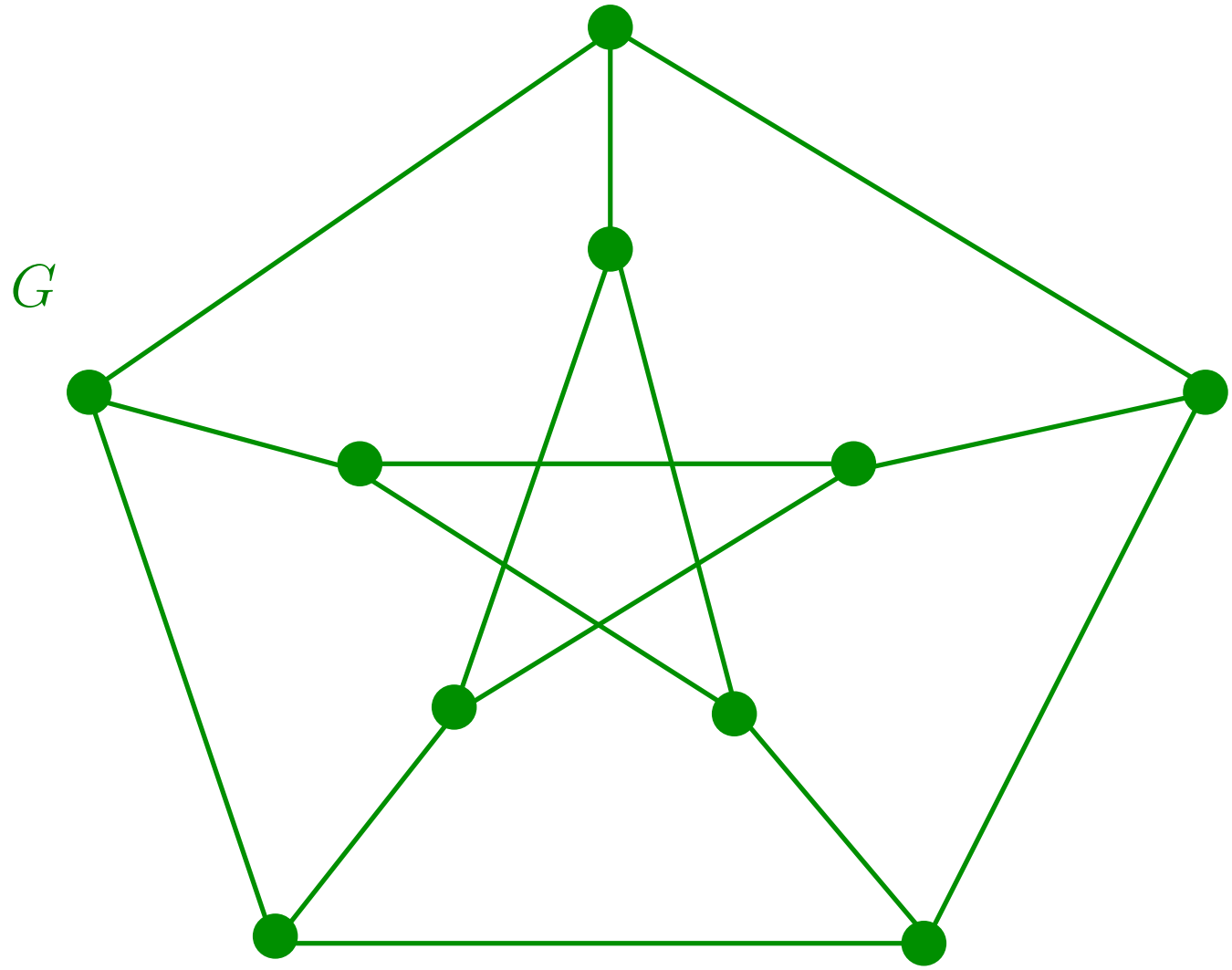
Grafos hamiltonianos

Problema: Dado um grafo encontrar um ciclo hamiltoniano.



Grafos hamiltonianos

Problema: Dado um grafo encontrar um ciclo hamiltoniano.



NÃO existe! Certificado? Hmmm . . .

Verificador polinomial para SIM

Um verificador polinomial para a resposta SIM de um problema Π é uma algoritmo polinomial ALG(I, C) que

recebe uma instância I de Π e um objeto C, $\langle C \rangle = \mathrm{O}(p(\langle I \rangle))$,

e

devolve SIM se e somente se $\Pi(I) =$ SIM,

onde $p(n)$ é um polinômio.

O objeto C é dito um certificado polinomial ou certificado curto da resposta SIM a $\Pi(I)$.

Exemplos

- Se G é hamiltoniano, então um ciclo hamiltoniano de G é um certificado polinomial:

  dados um grafo G e C pode-se verificar em tempo $\mathrm{O}(\langle G \rangle)$ se C é um ciclo hamiltoniano.

- se $X[1\,..\,m]$ e $Y[1\,..\,n]$ possuem uma ssco $\geq k$, então uma subseqüência comum $Z[1\,..\,k]$ é um certificado polinomial resposta:

  dados $X[1\,..\,m], Y[1\,..\,n]$ e $Z[1\,..\,k]$ pode-se verificar em tempo $\mathrm{O}(m+n)$ se Z é ssco de X e Y.

- se n é um número composto, então um divisor $d > 1$ de n é um certificado polinomial.

Classe NP

Formada pelos problemas de decisão que possuem um verificador polinomial para a resposta SIM.

Em outras palavras, um problemas de decisão Π está em NP se existe um problema Π' em P e uma função polinomial $p(n)$ tais que, para cada instância I de Π, existe um objeto C, $\langle C \rangle = \mathrm{O}(p(\langle I \rangle))$, e

$\Pi(I) =$ SIM se e somente se $\Pi'(I, C) =$ SIM.

O objeto C é dito um certificado polinomial ou certificado curto da resposta SIM de $\Pi(I)$.

Exemplos

Problemas de decisão com certificado polinomial para SIM:

- existe subseqüência crescente $\geq k$?
- existe subcoleção disjunta $\geq k$ de intervalos?
- existe mochila booleana de valor $\geq k$?
- existe mochila de valor $\geq k$?
- existe subseqüência comum $\geq k$?
- grafo tem ciclo de comprimento $\geq k$?
- grafo tem ciclo hamiltoniano?
- grafo tem emparelhamento (casamento) perfeito?

Todos esses problemas estão em NP.

$\mathbf{P} \subseteq \mathbf{NP}$

Prova:

se Π é um problema em P, então pode-se tomar a seqüência de instruções realizadas por um algoritmo polinomial para resolver $\Pi(I)$ como certificado polinomial da resposta SIM a $\Pi(I)$.

Outra prova:

Pode-se construir um verificador polinomial para a resposta SIM a Π utilizando-se um algoritmo polinomial para Π como subrotina e ignorando-se o certificado C.

Exemplo de certificado

Seqüência de operações do algoritmo EUCLIDES:

```
mdc(317811,514229)
   mdc(514229,317811)
      mdc(317811,196418)
         mdc(196418,121393)
            mdc(121393,75025)
               mdc(75025,46368)
                  mdc(46368,28657)
                     mdc(28657,17711)
                        mdc(17711,10946)
                           mdc(10946,6765)
                              mdc(6765,4181)
                                 mdc(4181,2584)
                                    mdc(2584,1597)
                                       mdc(1597,987)
                                          mdc(987,610)
                                             mdc(610,377)
```

Exemplo de certificado (cont.)

```
mdc(377,233)
  mdc(233,144)
    mdc(144,89)
      mdc(89,55)
        mdc(55,34)
          mdc(34,21)
            mdc(21,13)
              mdc(13,8)
                mdc(8,5)
                  mdc(5,3)
                    mdc(3,2)
                      mdc(2,1)
                        mdc(1,0)
```

certificado polinomial para mdc$(317811, 514229) = 1$

Outro certificado

EXTENDED- EUCLIDES (a, b) devolve d junto com x, y tais que $ax + by = d$

- podemos verificar se $d \mid a$ e $d \mid b$
- podemos verificar se $ax + by = d$

Se $d' \mid a$ e $d' \mid b$ então

$$d' \mid (ax + by) = d$$

e portanto $d' \leq d$

Conclusão: $d = \text{mdc}(a, b)$ e x, y são um certificado curto deste fato.

$\mathbf{P} \neq \mathbf{NP}?$

É crença de muitos que a classe NP é maior que a classe P, ainda que isso

não tenha sido provado até agora.

Este é o intrigante problema matemático conhecido pelo rótulo "P $\neq$ NP?"

Não confunda NP com "não-polinomial".

Verificador polinomial para NÃO

Um verificador polinomial para a resposta NÃO de um problema Π é uma algoritmo polinomial ALG(I, C) que

recebe uma instância I de Π e um objeto C, $\langle C \rangle = \mathrm{O}(p(\langle I \rangle))$,

e

devolve SIM se e somente se $\Pi(I) =$ NÃO,

onde $p(n)$ é um polinômio.

O objeto C é dito um certificado polinomial ou certificado curto da resposta NÃO a $\Pi(I)$.

Exemplos

- Se um grafo G com bipartição X e Y não possui um emparelhamento perfeito então $S \subseteq X$ tal que $|S| > |\text{vizinhos}(S)|$ é um certificado polinomial desse fato:

  dados um grafo G com bipartição X e Y e $S \subseteq X$ pode-se verificar em tempo $\mathrm{O}(\langle G \rangle)$ se $|S| > |\text{vizinhos}(S)|$.

- se n não é um número primo, então um divisor $d > 1$ de n é um certificado polinomial desse fato.

Classe co-NP

A classe co-NP é definida trocando-se SIM por NÃO na definição de NP.

Um problema de decisão Π está em co-NP se admite um certificado polinomial para a resposta NÃO.

Os problemas em NP $\cap$ co-NP admitem certificados polinomiais para as respostas SIM e NÃO.

Em particular, P $\subseteq$ NP $\cap$ co-NP.

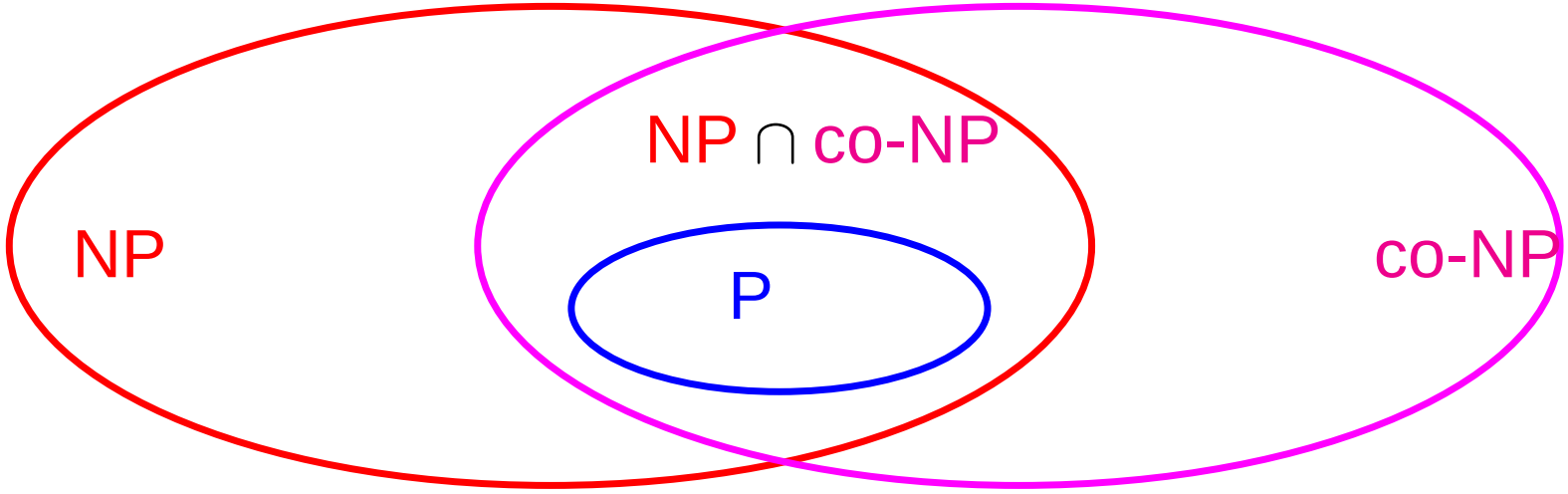
Exemplos

Problemas de decisão com certificado polinomial para NÃO:

- não existe subseqüência crescente $\geq k$?
- um dado número é primo?
- não existe subseqüência comum $\geq k$?
- grafo não tem emparelhamento (casamento) perfeito?

Todos esses problemas estão em co-NP.

P, NP e co-NP



P $\neq$ NP?

NP $\cap$ co-NP $\neq$ P?

NP $\neq$ co-NP?