

## \* NP Completo

Eficiente  $\approx$  Polinomial

↳ Em quê?

## \* Tamanho da Entrada

Nome:

Entrada:

Pergunta:

$$N = \Omega(\log n + \log w + n)$$

Esse tempo  $\Theta(n \cdot w)$  essa função que aparece na notação e descreve o tempo assintótico ele está limitado por um polinômio de  $N$  ou seja  $\exists c \text{ t.q. } n \cdot w \leq N^c$ ? Não  $\log w$  cresce mais devagar que  $w$  por isso não seria polinomial.

Trocando  $n \cdot 2^x \leq \log n + n^c$ ?

## \* Classe P

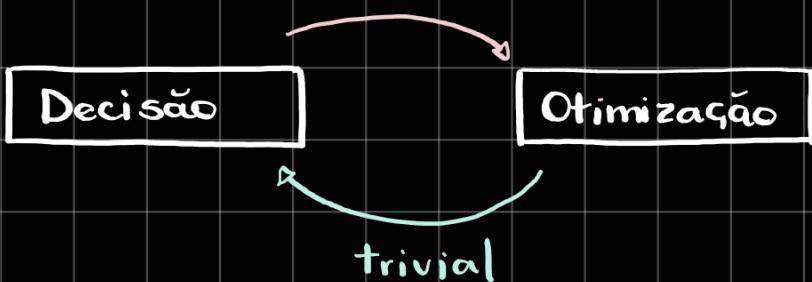
A classe P de problemas é o conjunto dos problemas de decisão que podem ser resolvidos em tempo polinomial no tamanho da entrada. Em uma máquina de Turing Determinística.

SSM (Versão de Decisão)

Entrada:  $n, v, K$

Pergunta:  $\exists$  subvetor de  $v$  com soma  $\geq K$ ?

A questão é se sei resolver a versão de decisão eu consigo resolver as outras versões também.



MAIOR ELEMENTO

Entrada:  $n, v[], K$

Pergunta: O maior elemento de  $v$  é  $\geq K$ ?

maior = -∞

para  $i$  até  $n-1$

se  $v[i] > \text{maior}$   $\leftarrow O(\lg v[i])$

    maior =  $v[i]$

se maior  $\geq K$

    return SIM

senão

    return NÃO

$O(n + \sum \lg v[i])$

é linear no tamanho da entrada, polinomial no tamanho da entrada que é o espaço para representar eles.

Pertence a P, basta escrever o algoritmo

PAR

Entrada: Inteiro  $n$

Pergunta:  $n$  é par?

$\begin{cases} O(1) \text{ se pegar último bit} \\ O(\lg n) \text{ se for dividir por 2} \end{cases}$

Pertence a P

## DIVISIVEL POR K

Entrada:  $n, K$

Pergunta:  $n$  é divisível por  $K$ ? {

$$\begin{cases} O(\lg(n) + \lg K) \\ O(\lg n \cdot \lg K) \end{cases}$$

Pertence a P

## PRIMO

Entrada:  $n$

Pergunta:  $n$  é primo? {

$$\begin{cases} O(\lg n) = N \\ O(\sqrt{n}) = O(n^{1/2}) \end{cases}$$

$$\begin{aligned} N &= \lg n && \downarrow \\ 2^N &= n && O((2^N)^{1/2}) \\ &&& O(2^{N/2}) \\ &&& O(\sqrt{2^N}) \end{aligned}$$

Primo ( $n$ )

para  $i=2$  até  $\lfloor \sqrt{n} \rfloor$

se  $n$  é divisível por  $i$

return NÃO

return SIM

Não é um algoritmo de tempo polinomial no tamanho da entrada.

Existe um algoritmo Primo que é polinomial no tamanho da entrada porém não é esse algoritmo acima.

Podemos dizer que Primo pertence a P.

## \* Classe NP

A classe NP de problemas é o conjunto dos problemas de decisão que podem ser verificados em tempo polinomial no tamanho da entrada. Em uma máquina de Turing Não-Determinística.

	P	NP
Teorema	Escrever uma Demonstração	Conferir uma Demonstração

Problema

Encontrar  
Resposta

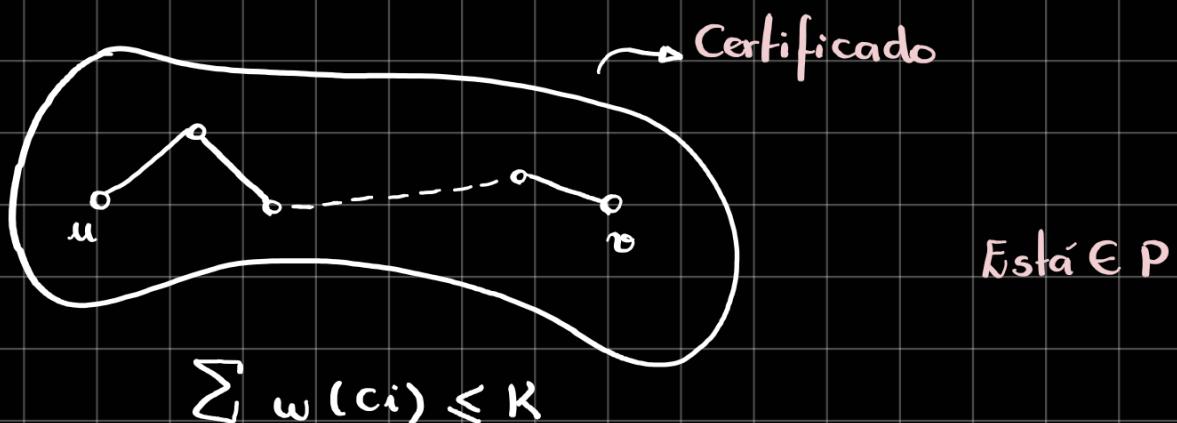
Verificar uma  
Resposta

$P \subseteq NP$

## CAMINHO MÍNIMO

Entrada: Grafo ponderado  $G$ , inteiro  $K$ ,  $u, v \in V(G)$

Pergunta:  $\exists$  caminho entre  $u$  e  $v$  de custo  $\leq K$ ?



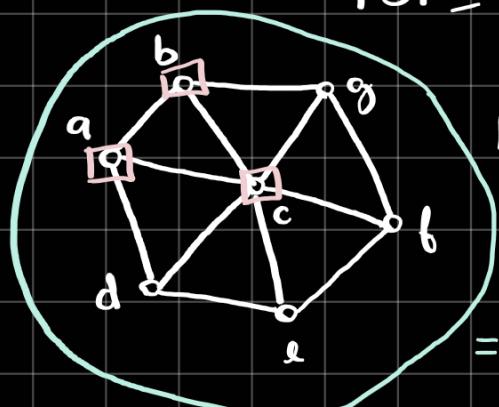
Se me der o caminho fica fácil conferir a resposta essa informação adicional é o certificado para o problema.

## CLIQUE

Entrada: Um grafo  $G$ , inteiro  $K$

Pergunta:  $\exists$  clique de tam  $\geq K$ ?  $\exists S \subseteq V(G)$

$|S| \geq K$  tal que  $\forall u, v \in S, u, v \in E(G)$ ?



$K=3$

Qualquer conjunto de vértice que pegar existe aresta entre eles

$\Rightarrow$  Uma instância SIM

Não sei resolver de forma eficiente para um grafo qualquer se me der um grafo gigantesco não sei resolver em tempo polinomial esse problema.

Agora se me der o conjunto  $S$  consigo conferir facilmente uma resposta.

Certificado :  $S$

Algoritmo não determinístico  
para encontrar uma CLIQUE  
de tamanho  $K$

$\text{ALGO}(G, K)$

$$S = \emptyset$$

para  $i = 1$  até  $K$

$x = \text{escolhe}(V(G)) \Rightarrow$  fazendo um papel

$S = S \cup \{x\}$  oráculo

Se  $S$  é clique

return SIM

return NÃO

Algoritmo

Se  $|S| < K$  ou  $S \notin V(f)$   
return NÃO

$\forall u \in S$   
 $\forall v \in S$   
se  $u \neq v \in E$  então  
return SIM

Pertence a NP

MOCHILA

Entrada:  $n, C, p_i, v_i, K$

Pergunta:  $\text{Lucro} \geq K ?$

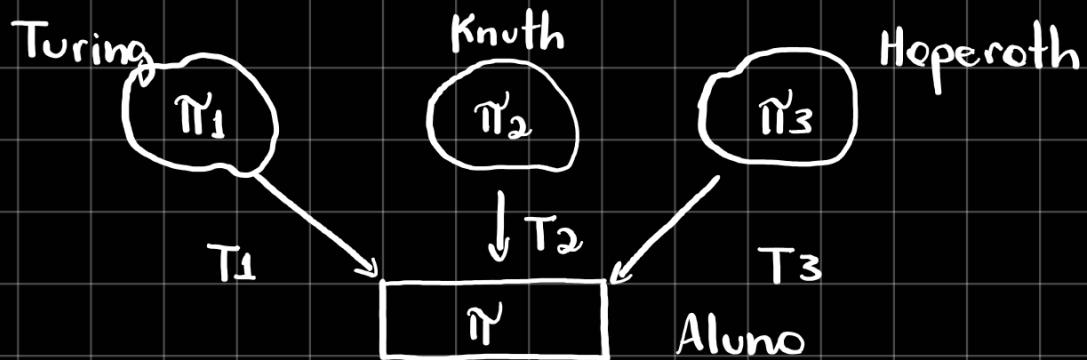
Pertence a NP

$$S = \boxed{1, 5, 6, 9} \Rightarrow \sum_{i \in S} p_i \leq C$$

$$\sum_{i \in S} v_i \geq K$$

Não sabemos se estão em P

- Mochila
- Clique



$\Pi'$  é um problema de decisão e  $I'$  uma entrada de  $\Pi'$



Se  $T(I')$  roda em tempo  $|I'|^c$   
 $|I| \leq |I'|^c$

Segunda caixa roda em tempo  $|I|^d$ .

Mas  $|I| \leq |I'|^c$ , então

$$|I|^d \leq (|I'|^c)^d = |I'|^{cd} = |I'|^c$$

Conclusão: Se  $\exists$  transformação polinomial de  $I' \in \Pi'$  para  $I \in \Pi$  e  $\exists$  algoritmo polinomial p/  $\Pi$ , então  $\exists$  algoritmo polinomial para  $\Pi'$ .

Uma transformação polinomial de um problema  $\Pi'$  para um problema  $\Pi$  é um algoritmo que recebe uma instânc-

Cia  $I'$  de  $\tilde{\Pi}'$  e retorna uma instância  $I$  de  $\tilde{\Pi}$  tal que.

- ① o algoritmo roda em tempo polinomial em  $|I'|$
- ②  $I'$  tem resposta SIM para  $\tilde{\Pi}'$  se somente se  $I$  tem resposta SIM para  $\tilde{\Pi}$ .

Se  $\exists$  uma transformação polinomial de  $\tilde{\Pi}'$  para  $\tilde{\Pi}$ , escrevemos

$$\tilde{\Pi}' \leq_p \tilde{\Pi} \text{ ou } \tilde{\Pi}' \in \mathbb{P}$$

Proposição: Se  $\tilde{\Pi}_1 \leq_p \tilde{\Pi}_2$  e  $\tilde{\Pi}_2 \leq_p \tilde{\Pi}_3$

$$|\tilde{\Pi}_1^c| \quad |\tilde{\Pi}_2^d|$$

$$\Rightarrow \tilde{\Pi}_1 \leq_p \tilde{\Pi}_3$$

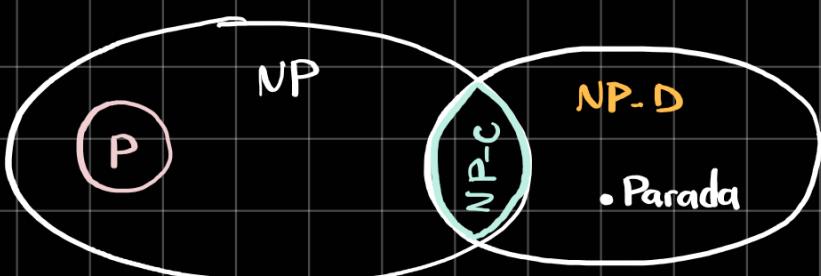
$$|\tilde{\Pi}_1^c|$$

### \* Classe NP-Difícil

Um problema  $\tilde{\Pi}$  pertence à classe NP-Difícil se  $\forall \tilde{\Pi}' \in \mathbb{NP}$ ,  
 $\tilde{\Pi}' \leq_p \tilde{\Pi}$

Obs.: Se  $\tilde{\Pi}$  é NP-Difícil e  $\exists$  algoritmo polinomial  $p_1(\tilde{\Pi})$   
então  $\exists$  algoritmo para todo problema em NP, isto é  
 $P = NP$ . Só que isso não foi provado ainda.

NP-Completo =  $\mathbb{NP} \cap \mathbb{NP\text{-Difícil}}$



## PARADA

Entrada: Máquina Turing M, Entrada algoritmo E para M

Pergunta: M para?

Teorema de Cook-Levin: SAT é NP-Difícil

## SAT

Entrada: Uma expressão lógica  $\varphi$  (fórmula) na CNF

Pergunta: Existe atribuição às variáveis de  $\varphi$  tal que a fórmula é verdadeira?

Forma Normal Conjuntiva (CNF)

Pertence à NP-D

Um literal é uma variável ou a negação de uma variável:

Ex:  $x_1, \bar{x}_1$

Uma cláusula é a disjunção (OU | OR) de literais.

Ex:  $x_1 \vee x_2, \bar{x}_1 \vee x_2, \dots$

Uma fórmula está na CNF se ela é a conjunção (E | AND) de cláusulas.

$$\varphi = C_1 \wedge \dots \wedge C_m$$

$$C_i = (l_{i1} \vee l_{i2} \vee \dots \vee l_{in})$$

$$l_{ij} = \begin{cases} x_k \\ \bar{x}_k \end{cases}$$

SAT ∈ NP  $\Rightarrow$  SAT é NP-Completo

$$\varphi = (x_1 \vee x_2) \wedge (\bar{x}_2) \wedge (\bar{x}_1 \vee x_2 \vee x_3) \wedge (x_1 \vee \bar{x}_3)$$

$x_1 = T$

$x_2 = F$

$x_3 = T$

## CLIQUE

Entrada:  $G, K$

Pergunta:  $G$  possui clique de tamanho  $\geq K$ ?

CLIQUE  $\in$  NP?

Certificado:  $S \subseteq V(G)$ , uma clique de tamanho  $\geq K$

**Certificado**

Se $ S  < K$ , return NÃO
Se $S \subseteq V(G)$ , return NÃO
Para $i, j \in S$
se $i \neq j$ e $ij \in E(G)$
return NÃO
return SIM

CLIQUE  $\in$  NP-Difícil?

Mostraremos que  $SAT \leq_p \text{CLIQUE}$

① Escolher de qual problema quero reduzir

② Qual é o algoritmo que realmente pega uma instância e gera outra instância

③ Mostrar que a resposta do problema original para

a instância original é a mesma resposta da instância que construi para o problema do qual eu reduzi.

Fazer a redução a partir de SAT

$\Phi = C_1 \wedge \dots \wedge C_m$ , construiremos  $G, K$   
n variáveis

$l_{ij}$  é um literal da  $i$ -ésima  
 $\rightarrow$  cláusula  $\in C_i$

$\forall$  literal  $l_{ij}$  de  $\Phi$  crie um vértice  $v_{ij}$  em  $V(G)$ .

$\forall$  par de vértices  $v_{ij}, v_{kp}$ , adicione arestas entre eles se:

①  $i \neq k$  (estão vindo de cláusulas diferentes, vértices da mesma cláusula não terão arestas entre si, vértices correspondentes a literais da mesma cláusula não contém vértices entre si).

②  $l_{ij} = l_{kp}$  Ou os literais correspondem a variáveis diferentes correspondem à mesma variável e ambos são positivos ou ambos são negativos

Faça  $K=m$

$l_{11} \quad l_{12} \quad l_{21} \quad l_{31} \quad l_{32} \quad l_{33} \quad l_{41} \quad l_{42}$

$\Phi = (x_1 \vee x_2) \wedge (\neg x_2) \wedge (\bar{x}_1 \vee x_2 \vee x_3) \wedge ((\bar{x}_1 \vee \bar{x}_3)$

$C_1$

$C_2$

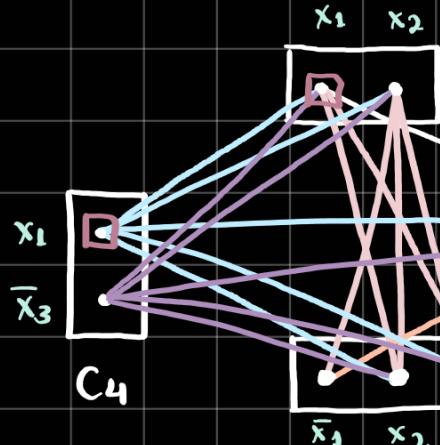
$C_3$

$C_4$

$x_1 = T$

$x_2 = F$

$x_3 = T$



$x_2$  e  $x_2$  pode

$x_2$  e  $\bar{x}_2$  são a mesma varia-

vel, ambos não são

positivos ou negativos

se é positivo e o outro

é negativo.

$K=4$  número de cláusulas

Para que seja verdadeiro  $x_1=T, x_2=F, x_3=T$ , toda cláusula tem pelo menos um literal verdadeiro.

Se existe uma formula de tornar verdadeira, existe uma clique no grafo acima.

Tamanho certo é porque  $K$  é igual ao número de cláusulas, se na formula selecionei um literal de cada cláusula de fato no desenho peguei um conjunto de vértice do tamanho certo.

Mostramos que se  $\varphi$  é SIM então  $(G, K)$  é SIM ✓

Já fizemos antes

Se  $\varphi$  é não então  $(G, K)$  é não, se não é possível ter a cláusula de cima então não é possível ter uma clique de tamanho  $K$ .

=

Mas dessa forma não é o mais fácil de fazer, mais fácil fazer algo equivalente que é usar contra positivo, que é o que?

Se a instância que eu construi tem uma resposta SIM então a instância original é SIM.

Se  $(G, K)$  é SIM  $\Rightarrow \varphi$  é SIM.

$$A \rightarrow B = \neg A \rightarrow \neg B$$

Com isso CLIQUE é NP-Difícil e já sabia que CLIQUE pertencia a NP, logo ele é NP-Completo

Suponha  $\varphi$  é satisfatível. Se  $l_{1,1}, l_{2,2} \dots l_{m,m}$  literais verdadeiros. Mostraremos que os vértices correspondentes  $\{v_{1,1}, \dots, v_{m,m}\}$  formam uma clique de tamanho  $K$  em  $G$ .

Note que pela construção como esses vértices satisfazem 1, ou seja, fazem parte de cláusulas diferentes e satisfazem 2 (explicado acima) ou seja, correspondem a variáveis diferentes ou tem que ser da mesma variável concordando ambas positivas ou negativas então as arestas existem logo isso é uma clique.

Com isso  $SIM \rightarrow SIM$

Agora vamos ver  $SIM \leftarrow SIM$

Por outro lado, seja  $S$  uma clique de tamanho  $K$  em  $G$ . Primeiro não pode ter dois vértices que vieram da mesma cláusula, segundo note que  $K = m$  então tenho que ter um vértice e cada vértice dessa clique correspondente a um literal agora constrói uma atribuição da seguinte maneira para cada vértice da clique faça o literal correspondente ser verdadeiro note que isso satisfaz a cláusula mas note também que nenhum conflito está sendo gerado porque nenhum literal pode aparecer na clique negado e não negado por isso consigo satisfazer a fórmula original e portanto a instância é  $SIM$  também. Concluímos a prova.

$SAT \leq_p 3\text{-SAT}$

↳ cada cláusula tem  $\leq 3$  literais

Pertence a NP-C

Toda instância desse problema gera uma instância de SAT também, então o mesmo argumento para falar que SAT é NP posso usar o mesmo que 3-SAT pertence a NP o mesmo argumento de conferir uma resposta funciona aqui.

Claramente 3-SAT ∈ NP

Se mostrarmos SAT ≤<sub>p</sub> 3-SAT então 3-SAT é NP-Difícil

$$\varphi = C_1 \wedge C_2 \wedge \dots \wedge C_m$$

Diagram illustrating the formula  $\varphi = C_1 \wedge C_2 \wedge \dots \wedge C_m$ . The clauses  $C_1$  and  $C_2$  are shown as clouds containing specific sets of literals.  $C_1$  contains  $l_1 \wedge l_2 \wedge l_3 \wedge l_4$ .  $C_2$  contains  $l_1 \wedge l_2 \wedge l_3 \wedge l_4 \wedge l_5$ .

$$(l_1 \vee l_2 \vee l_3 \vee l_4)$$

$$(l_1 \vee l_2 \vee y_1) \wedge (\bar{y}_1 \vee l_3 \vee l_4)$$

$$(l_1 \vee l_2 \vee l_3 \vee l_4 \vee l_5) \leftarrow 5$$

$$(l_1 \vee l_2 \vee y_1) \wedge (\bar{y}_1 \vee l_3 \vee l_4 \vee l_5)$$

$$\underbrace{(y_1 \vee l_3 \vee y_2)}_{3} \wedge (\bar{y}_1 \vee l_4 \vee l_5)$$

Comprimento da formula resultante, a cada passo tira um literal então o comprimento da formula resultante correspondente a uma cláusula o número de cláusulas que constrói é cada uma de tamanho constante no maximo 3 e o nº delas é linear no comprimento da cláusula original e é polinomial mantendo a correspondencia entre as instâncias.

CLIQUE ∈ NP

Se a resposta SIM, dado um certificado C, consigo verificar que a resposta é SIM em tempo polinomial na entrada  $(G, K)$ .

↳ certificado vai ser polinomial no tamanho da entrada

$$C \subseteq V(G), |C| \leq |V(G)|$$

PARADA

Entrada: Algoritmo A, Entrada E

Pergunta: A(E)?

CLIQUE  $\leq_p$  PARADA  
 $(G, K)$                      $(A, E)$

$$E = (G, K)$$

$$|(A, E)| = |(G, K)| + c$$

$\uparrow$        $\underbrace{\quad}_{\text{  }}$        $\uparrow$

O da direita para se somente se tenho uma clique em tempo polinomial.

Em tempo polinomial dado a clique construir o da parada.

$\forall S \subseteq V(G), |S| = K$        $= A$   
Se S é clique  
pare  
while (1);

Preciso de uma quantidade constante de bits e não depende da entrada.

↳ Bytes de entrada para o algoritmo

Se ele parar é porque tem uma clique de tamanho K  
sendo assim mostrei que a parada é NP-Difícil.

Se eu conseguir reduzir qualquer problema NP-Difícil para um problema X, então X é NP difícil também.  
Meu problema vai ser pelo menos tão difícil quanto todo NP-Difícil.

## CONJUNTO INDEPENDENTE

Entrada:  $G, K$

Pergunta:  $\exists S \subseteq V(G), |S| \geq K$

t.q.  $\forall u, v \in S, u, v \notin E(G) ?$

CLIQUE  $\leq_p$  C.I (Redução de um problema que sei que é dif.)

$(G, K) \rightarrow (G', K')$  círculo para o que eu quero mostrar que é dif.)

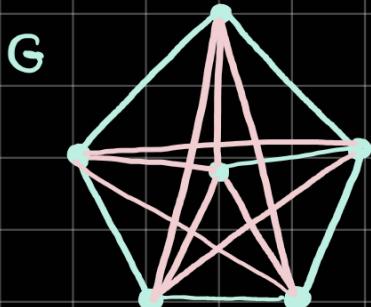
$$K' = K$$

$$G' = \bar{G} \quad \bar{G} = (V(G), \bar{E})$$

Pega a entrada do problema da esquerda e constrói a entrada do problema da direita essa é a transformação.

$\bar{G}$  é o complemento de um grafo, que vai ter o mesmo conjunto de vértices do grafo original e tem um conjunto de aresta  $\bar{E}$ , que são pares  $u, v$  (são vértices no grafo original) mas que as arestas entre eles não existem no grafo original.

$$\bar{E} = \{u, v \mid u, v \in V(G), u, v \notin E(G)\}$$

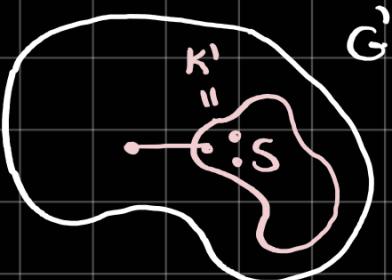
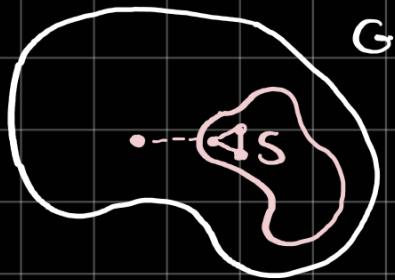


$\bar{G}, \bar{E}$  (arestas que estão faltando)

Se  $(G, K)$  é SIM p/ CLIQUE

$\Rightarrow$

$(G', K')$  é SIM p/ C.I.



Imagina que a instância tem de fato uma clique, então tenho um grafo  $G$  e achei uma clique de tamanho  $S$ , sei que o  $G'$  ele é parecido olha para o mesmo conjunto  $S$  em  $G'$  (sei que tem que existir porque os vértices são os mesmos).

Se no  $G$  tinha uma aresta entre cada par de vértice se era uma clique a definição do complemento vai apagar todas as arestas e adicionar em quem não era adjacente.

Se tinha uma clique em  $S$  de  $G'$ , temos um conj. independente em  $S$  de tamanho  $K'$  em  $G'$ .

Com isso concluímos que é uma resposta SIM para conj. independente.

Na outra direção é o mesmo argumento apenas trocar clique por conj. ind.

Se  $\tilde{\Pi} \in NP$  e  $\tilde{\Pi}' \in NP-D \Rightarrow \tilde{\Pi} \leq_p \tilde{\Pi}'$

$\tilde{\Pi} \leq_p \tilde{\Pi}'$

- |   |         |         |                                   |
|---|---------|---------|-----------------------------------|
| ① | fácil   | fácil   | OK                                |
| ② | fácil   | difícil | OK      ex: par $\rightarrow$ sat |
| ③ | difícil | fácil   | impossível, só se $P = NP$        |
| ④ | difícil | difícil | OK                                |

fácil - polinomial  
difícil - NP-D

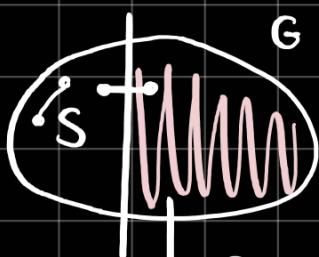
## COBERTURA POR VÉRTICE (Vertex Cover)

Entrada:  $G, k$

Pergunta:  $\exists S \subseteq V(G), |S| \leq k,$

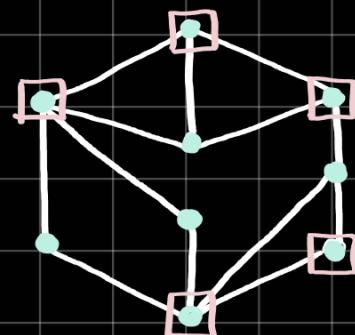
t.q.  $\forall u, v \in E(G), u \in S \text{ ou } v \in S?$

Quero escolher um conjunto de vértice (pequeno) de forma que toda aresta tenha pelo menos uma das suas extremidades selecionadas



Quero um conjunto  $S$  que cubra todas as arestas do grafo.

Oscara que estão fora de  $S$  são Conjunto Independente. (não tem nenhuma aresta entre eles)

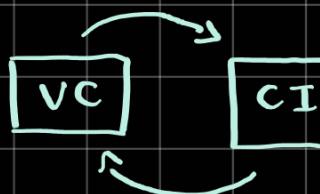


Se uma aresta tem uma extremidade em  $S$ , significa que nenhuma aresta tem as duas extremidades fora de  $S$ .

$u   v$	$v \in S$
$u \in S   v \in S$	não pertence
$u \in S   v \notin S$	$v \in S$
$u \notin S   v \in S$	$v \in S$
$u \notin S   v \notin S$	não pode acontecer

$\forall u, v \in V(G) - S, u, v \notin E(G)$

logo  $V(G) - S$  é CI.



Se tenho vertex cover o que sobra é conj. indep. ao contrário é vdd.

Obs.:  $\forall G, \forall S \subseteq V(G), S$  é Vertex Cover  $\Leftrightarrow V(G) - S$  é C.I.

$\uparrow$  V.C.      CI  $\downarrow$

$$|S| + |V(G) - S| = |V(G)|$$

$$CI \leqslant_p VC$$

$$(G, K) \quad (G', K')$$

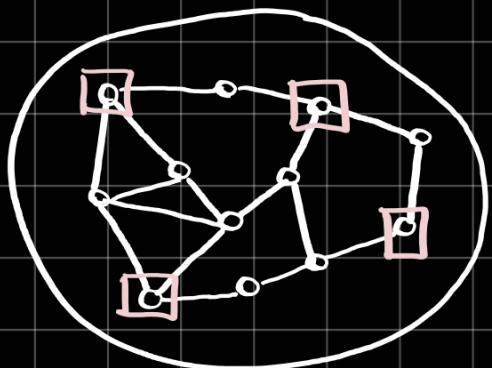
$$\begin{array}{c} \uparrow V.C. \quad CI \downarrow \\ |S| + |V(G) - S| = |V(G)| \\ K' + K \\ \Downarrow \\ K' = V(G) - K \end{array}$$

**CONJUNTO DOMINANTE (Dominating Set)**

Entrada:  $G, K$   $\mapsto$  conjunto

Pergunta:  $\exists D \subseteq V(G), |D| \leq K,$   
 t.q.  $\forall v \in V(G), N[v] \cap D = \emptyset$

$$N(v) \cup \{v\}$$



$$VC \leqslant_p CD$$

$$(G, K) \quad (G', K')$$



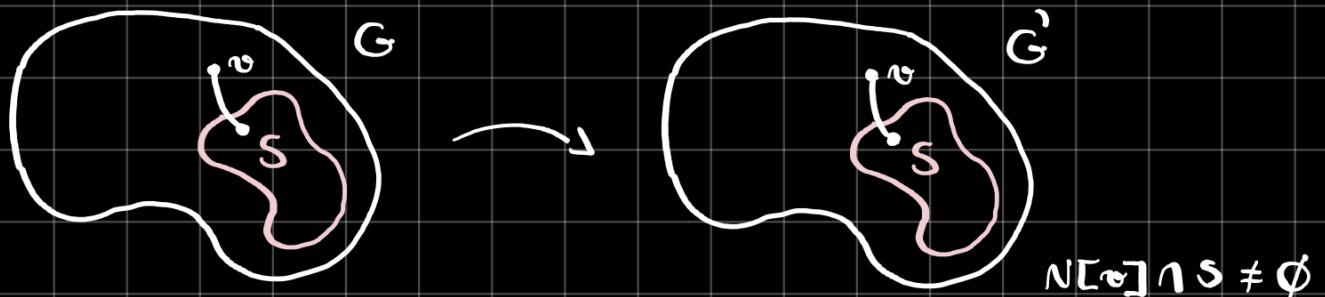
Transformar aresta da esquerda em vértice, porque na esquerda estava cobrindo arestas e na direita queremos cobrir vértices.

$$V(G') = [V(G)] \cup \{x_{uv} \mid u, v \in E(G)\}$$

$$E'(G) = [E(G)] \cup \{x_{uv} \mid u, v \in E(G)\}$$

↳ parte branca do desenho  
 ↳ parte rosa do desenho  
 ↳ p/cd aresta criou 1 vértice  
 ↳ p/cd aresta criou 2 arestas

Seja  $S$  um V.C de  $G$  com tamanho  $|S| \leq K$ . Quero mostrar que  $S$  é um CD de  $G'$ .

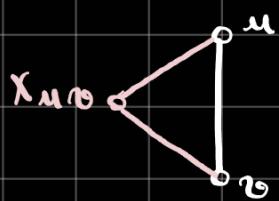


Se tenho uma aresta em  $v$  a outra extremidade tem que estar dentro de  $S$ , então para as arestas brancas originais do grafo se o  $v$  está fora ele tem que ter um vizinho dentro de  $S$ , porque se não a aresta nunca estaria coberta. Estamos assumindo que o grafo não tem nenhum vértice isolado se aresta, porque é uma instância trivial de vertex cover. (Não muda a resposta de vertex cover, pode jogar fora).

Se um vértice que estava fora de  $S$  na esquerda tem que ter um vizinho de  $S$  na esquerda então ele vai continuar tendo o mesmo vizinho de  $S$  lá fora então ele está coberto, está dominando, é verdade que a vizinhança fechada dele quando faço a intersecção  $S$  é de fato diferente de vazio.

Agora os vértices rosa, todos os vértices rosa estão fora de  $S$ .

Vamos chamar ele de  $X_{u,v}$  ele tem dois vizinhos que são brancos (extremidade do grafo original)  $u, v$  e esses dois estavam em  $G$ ,  $u$  e  $v$  existe essa aresta e é um vertex cover então  $u$  ou  $v$  estão dentro de  $S$ , se não a aresta nunca tinha sido coberta.



$$N(X_{u,v}) = \{u, v\}$$

$$N(X_{u,v}) \cap S \neq \emptyset$$

Como isso mostramos que todos os vértices são dominados por  $S$ .

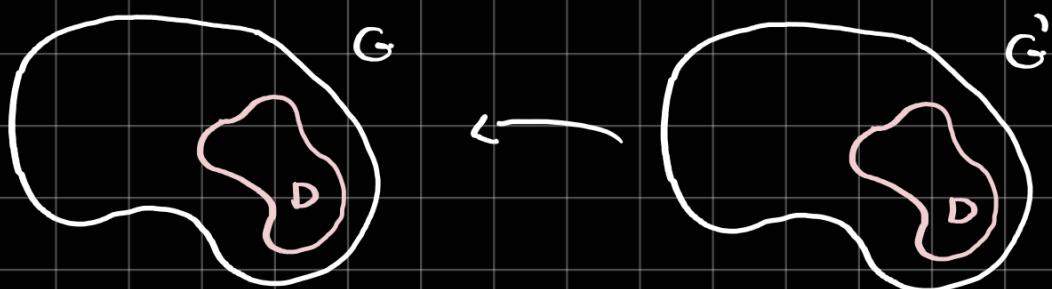
Se  $(G, K)$  é SIM p/ V.C.  $\Rightarrow (G', K')$  é SIM p/ C.D.

Fazendo a volta

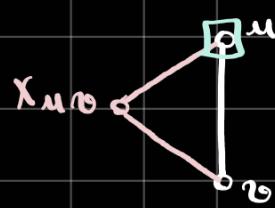
Se  $(G', K')$  é SIM p/ C.D.  $\Rightarrow (G, K)$  é SIM p/ V.C.

Seja  $D$  um conj. dominante de  $G'$  com  $|D| \leq K'$

(Não posso fazer ao contrário da ida porque os vértices rosa iam gerar um vazio no conj.  $D$ )



O vértice rosa tem dois vizinhos  $u$  e  $v$ , então o que vamos fazer é se tiver um vértice rosa troca por um dos vizinhos dele ou seja se  $x_{u,v}$  estivesse em  $D$  ele vai deixar de estar em  $D$  e colocamos por exemplo  $u$  no lugar dele. De fato não tem pro-



blema fazer isso, porque o  $x_{u,v}$  estava na solução para ser um conjunto dominante, e quem ele está vigiando? À se mesmo  $u$  e  $v$  que são os vizinhos dele, se mudar para  $u$ , continua vigiando os mesmos.

Se  $\exists x_{u,v} \in D$ , substitua-o por  $u$  e seja  $S$  o conjunto resultante após a aplicação exaustiva dessa operação.  
Note que  $S$  também é C.D., com  $|S| \leq |D|$  e  $S \subseteq V(G)$ .

Mostrar que S é V.C. de G. (Pegar uma aresta qualquer e mostrar que ela está coberta por esse conjunto). Seja  $u, v \in E(G)$  note que em  $G'$ ,  $x_{u,v}$  é dominado por S, ou seja  $\{u, v\} \cap S = \emptyset$  logo S cobre  $u, v$  em G.

O vértice  $u$  ou  $v$  que dominava o  $G'$  vai dominar uma aresta em G, como isso é verdade portanto S é um vertex cover de G.

## CICLO HAMILTONIANO

Entrada: Grafo G

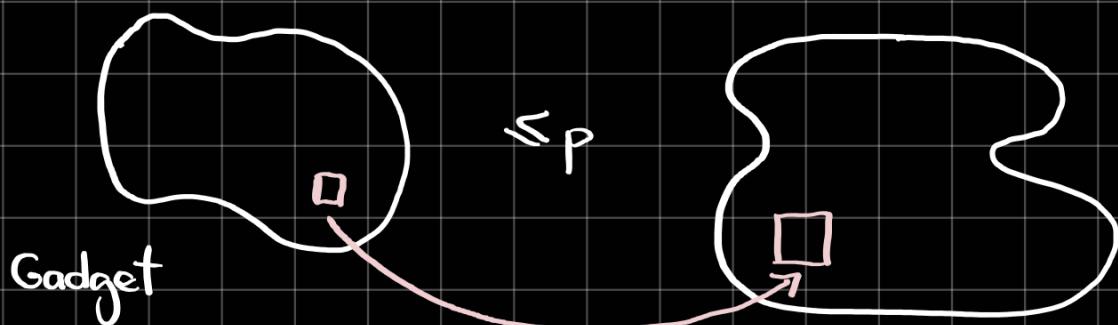
Pergunta:  $\exists$  ciclo hamiltoniano  
em G?



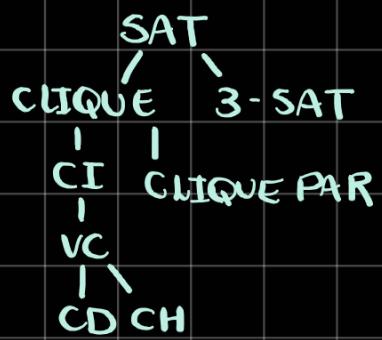
Quero encontrar um ciclo passando por todos os vértices apenas uma vez. Não pode repetir vértice e nem deixar de passar em um deles

Quando temos um problema em que a cara dele é muito diferente, algo que ajuda é pensarmos em Gadget.

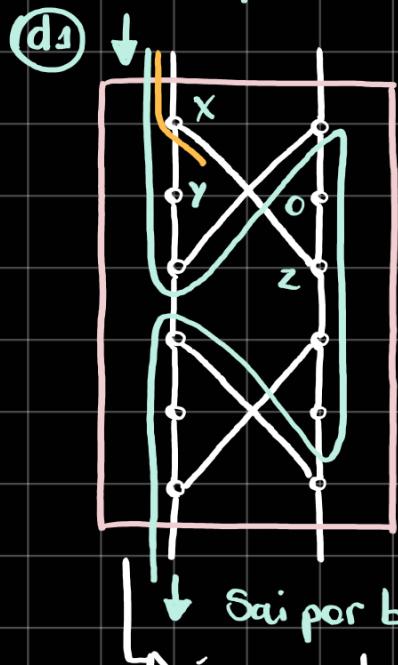
A ideia é tentar traduzir esse problema da esquerda no problema da direita, pedacinhos desse problema vão ser substituído por pedaços desse outro problema.



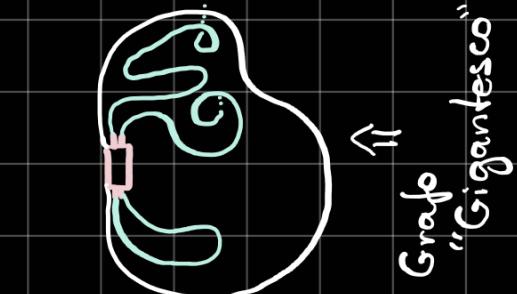
V.C.  $\leqslant_p$  C.H.  
 $(G, K)$        $(G', K')$



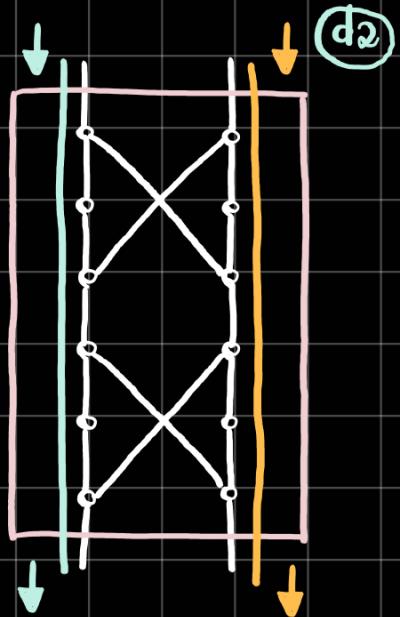
Gadget que iremos usar:  
 Entrar por cima:



↓  
 é um pedaço de um  
 grafo gigantesco



Se for um C.H. nunca  
 vai fazer essa curva,  
 porque não passa por y  
 se isso acontecer não  
 é C.H.  
 → posso ter ele "espelhado"  
 também pela direita



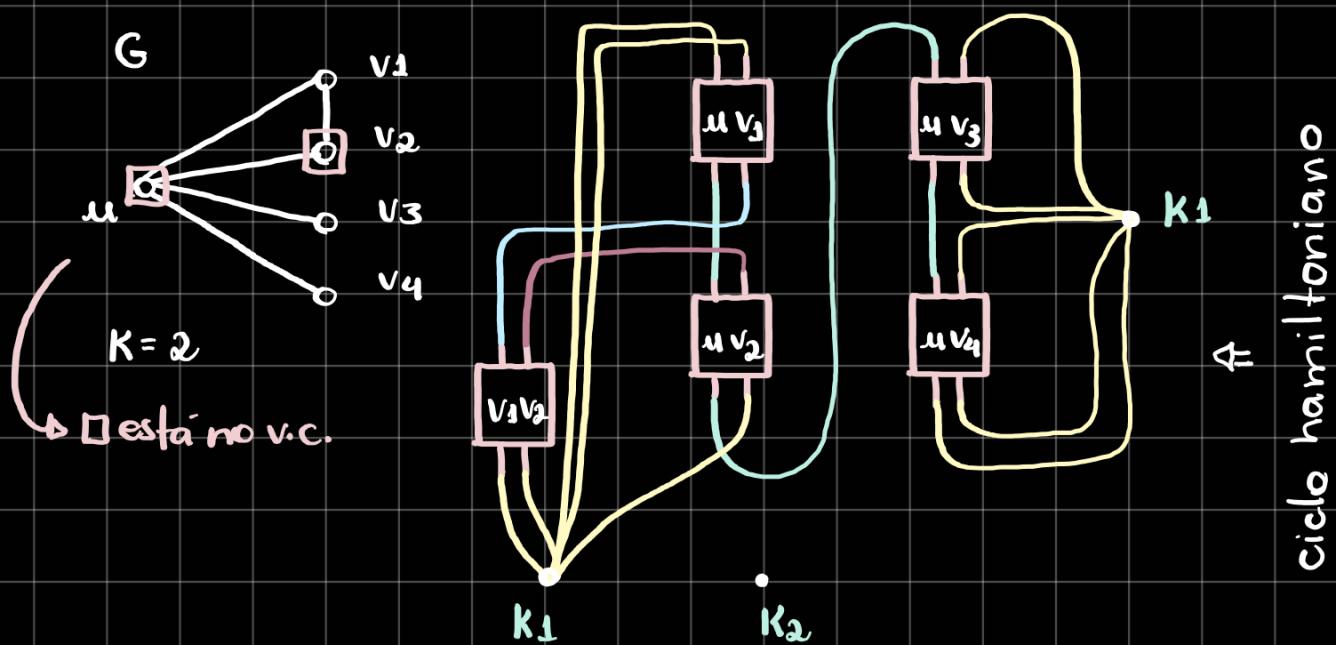
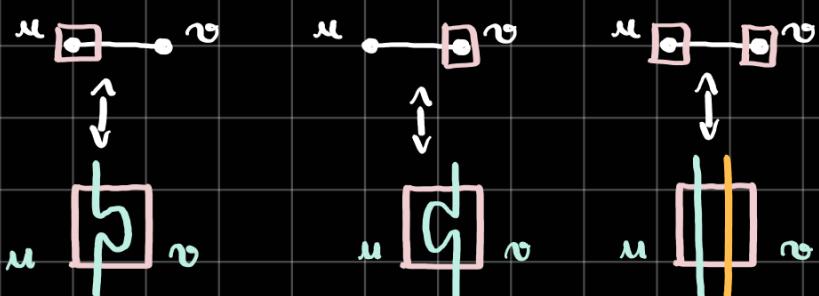
Há apenas duas formas de passar pela  
 caixinha, no desenho a esquerda que  
 visita todos os vértices de uma vez. Ou  
 o do lado direito que tem que entrar  
 duas vezes para visitar todos.  
 Referente ao desenho d1 e d2

No grafo "gigantesco" vamos procurar um C.H. que passa  
 pelos vértices apenas uma vez. Olhando para a caixinha  
 que tem 4 entradas, em algum momento ele vai passar por  
 uma dessas entradas possivelmente visitar algo e sair.

Fazendo isso em cada entrada.

- ① Cada vez que entra na caixinha ele sai, nº entrada é igual ao número de saída.
- ② Entra no máximo duas vezes por essa caixinha

Agora vamos pensar no vertex cover, quando tem uma aresta entre  $u$  e  $v$  à 3 maneiras de cobrir os vértices.



Como conectar o lado da caixinha correspondente ao  $u$ , se tiver uma aresta ligando  $v_1$  e  $v_2$  adiciono uma caixinha e ligo  $v_1$  no  $v_1$  e  $v_2$  no  $v_2$ .

Se  $K=2$ , cria-se dois vértices a mais e liga todas as "pernas" soltas no  $K_1$  e também liga as mesmas no  $K_2$ .

(No desenho tem dois  $K_1$ , porém é o mesmo vértice coloquei apenas para não ficar baguncado).

Agora argumentar que essa solução funciona

Quero mostrar se tenho um vertex cover para essa instância tenho um ciclo hamiltoniano e vice-versa.

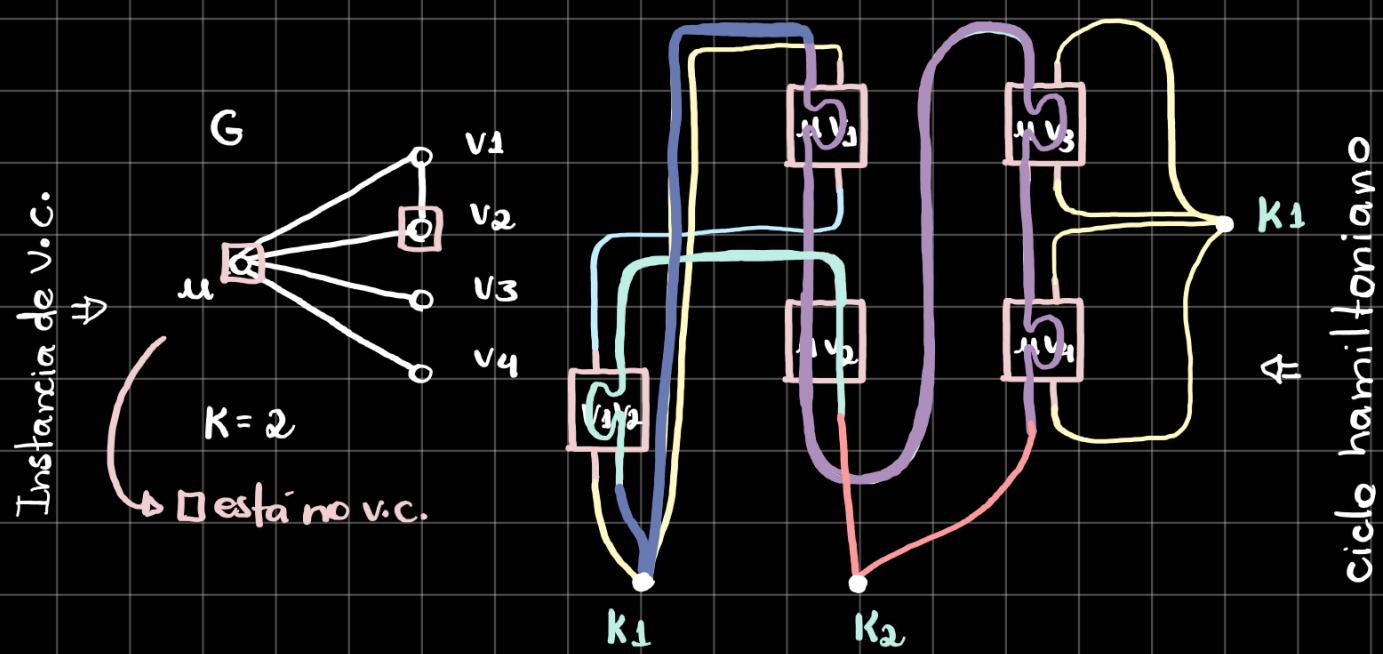
Vamos assumir que a resposta é SIM para essa instância, o que é verdade. Poderia pegar  $u$  e  $v_2$ , e com isso ocupar todas as arestas do grafo.

Como construir o ciclo hamiltoniano que passa por todos os vértices do grafo pelo  $K_1$ ,  $K_2$  e todos os vértices dentro da caixinha.

Obs.:  $\exists$  um vertex cover  $\leq K$ , porém vamos assumir  $= K$ , porque se for menor pode adicionar novos vértices sem perder nada com isso, a resposta continua sendo SIM.

Se tenho um v.c.  $\leq K$  tenho um v.c. igual à  $K$  para simplificar a argumentação.

Criar o ciclo hamiltoniano de roxo. Construindo outra solução.



Porém queria um ciclo, então vamos usar o  $K_1$  e  $K_2$  para fechar um ciclo. Como? Pega o primeiro  $K_1$ , ordena os vértices do vertex cover então o  $K_1$  vai ligar no 1º e 2º, o  $K_2$  vai ligar 2º e 3º ... o último liga no último e primeiro do vertex cover.

Se tenho v.c. de tamanho K consigo encontrar um c.h., tem que se convencer está passando por todos os vértices, de que isso forma um ciclo e que não repete vértices.

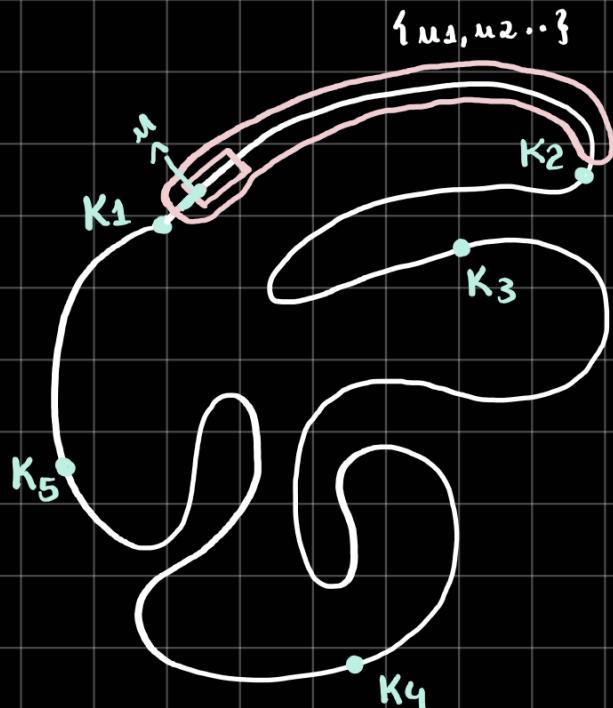
Toda aresta do grafo original G está coberta pelo v.c. e pode estar coberta por 1 vértice ou 2. A caixinha correspondente a ela pela forma que construiu, se tiver coberto por um vértice só todos os vértices vão ser cobertos pelo mesmo trecho de caminho, se tiver dois vai ser coberto por dois trechos de caminho. Então cada vértice da cobertura terá um pedaço de um caminho.

A partir da cobertura por vértice conseguimos construir o ciclo hamiltoniano.

Recapitulando o que foi feito anteriormente  
Dado o grafo que não conhecia a solução, descrevi como construir o grafo da direita, que para cada aresta faz as caixinhas depois adicione K vértices novos que não faz parte de uma caixinha depois ligue as caixinhas dessa maneira e depois ligue os K vértices na caixinha dessa outra maneira esse foi o algoritmo escrito para dado um grafo construir o grafo da direita.

Agora provar a volta, outra direção:

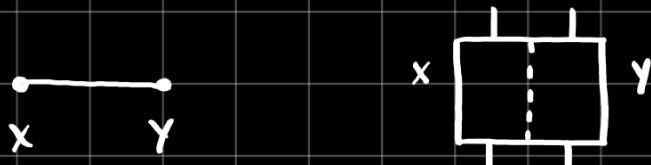
Suponha que temos um ciclo na direita quero mostrar que tenho um vertex cover. O argumento é parecido porém de trás para frente.



Desenho um pouco mais abstrato, então no meu ciclo terrei vários vértices  $K$  e entre eles teremos as caixinhas, toda entrada e saída de caixinha corresponde ao vértice  $u$ . Então adiciona ele na solução e assim por diante.

Quero mos afirmar que de fato é um vertex cover.

Podemos usar contradição, suponha que não, tem uma aresta do grafo que não foi coberta pega a caixinha correspondente aquela aresta. Suponha uma aresta no graf que nunca foi coberta  $(x, y)$  e a sua caixinha correspondente



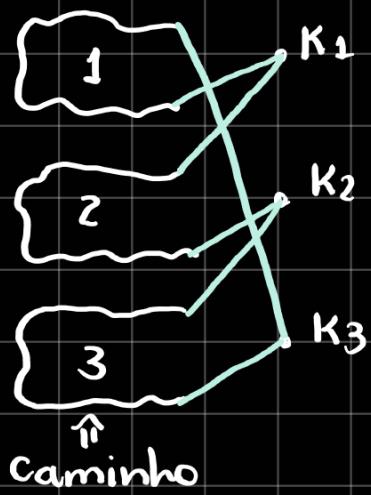
Se é um C.H. essa caixinha tem que ter sido visitada tem que ter entrado por algum lugar se foi por

$x$  ele tem que ter entrado no conjunto, então a aresta tem que estar coberta.

Esses argumentos deixam tudo bem "amarrado" e na volta consegue extrair o V.C. a partir do C.H.

C.H. é NP-Difícil

Para ver que é NP-completo, basta ver que é fácil conferir um certificado, me diz a ordem dos vértices que preci-



sa visitar, dado um grafo tem que convencer que de fato tem um caminho hamiltoniano. Pego a ordem dos vértices, verifico se todos os vértices estão realmente lá e verifico se entre cada par de vértice consecutivos realmente existe uma aresta. Com isso mostramos que esta em NP.

Mostrei que meu problema é NP-Difícil o que assumimos?

Não é possível resolver : Combinar ou flexibilizar o que

a) todas as instâncias

vamos resolver

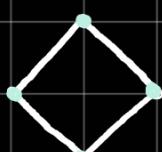
b) de maneira exata

← Sem garantias

c) em tempo polinomial

→ com garantias

Exemplo da seleção de atividade, esse problema é equivalente ao conjunto independente, ele é NP-Difícil mas para essa aplicação em específico se os objetos que quero selecionar sem conflito são intervalos no tempo para essa versão do problema não consegui fazer aparecer isso



o grafo que sai desse tipo de aplicação é bem particular e as vezes conseguimos utilizar das propriedades específicas do problemas para resolver, com isso estamos no caso a podemos

abrir mão e fazer o b e c.

Heurística estratégia tipicamente gulosa que de alguma forma usa um critério que na média vai parecer bom e que vai construir alguma solução, estratégia sem garantias.

Letra b, com garantia usando algoritmos aproximativos. Conferir dada uma solução  $S$  em que o meu valor seja pelo menos metade do ótimo. Para muitos algoritmos consegui -

$$\pi(S) \geq \frac{V}{2}$$

mos provar esse fator de 2 como por exemplo o vertex cover.

São interessante porque estamos conformando que o problema é difícil que não vamos conseguir uma solução exata, mas não está jogando para a galera vou fazer qualquer coisa aqui e ver o que vai dar. Esta fazendo um algoritmo que vai te dar alguma garantia, apesar de não ser o ótimo.

Quero abrir mão da letra C

$$\text{Ao invés } O(2^{|V(G)|} \cdot n^2) \rightarrow \text{AEE } O(c^{|V(G)|}), c < 2$$

Sei que ele vai ser exponencial mas quero garantir uma complexidade boa para ele.

Uma outra direção é a complexidade parametrizada (CP) ao invés de mudar a base (também pode mudar a base) mas tenta mudar o expoente também.

CP  $O(c^k)$  vai ser exponencial mas não no tamanho da entrada, vai ser em outra coisa tipicamente menor.

Essas duas são abordagem por garantia, pois ainda estamos pensando no pior caso.

Uma outra abordagem muito interessante e usada na prática.

PLI (Programação Linear Inteira) tem todo um

arcabouço de desigualdades lineares e modela o seu problema de maneira específica e usa um conjunto de técnicas e solvers. Resolvedores de problemas desse tipo como cplex.

Existe outras técnicas sem garantia chamadas SAT-SOLVER.

Uma família de problemas gigantesco que consegue modelar com o problema SAT.

SAT solver são software que usam várias estratégia para resolver versões da SAT da maneira mais rápida possível.

Tese de Church Turing, tudo que um computador pode fazer uma máquina de turing também pode.