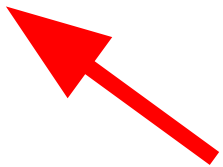
Grafos perfeitamente ordenáveis

Por Gabriel Nascarella Hishida do Nascimento

Tem muito a ver com o problema da coloração.

Computational complexity

Graph coloring is computationally hard. It is NP-complete to decide if a given graph admits a k-coloring for a given k except for the cases $k \in \{0,1,2\}$. In particular, it is NP-hard to compute the chromatic number.[29].



Porém, existem classes de grafos em que sua coloração pode ser **MUITO** mais fácil!

De fato, pode ser LINEAR!

Grafos cordais, por exemplo

Eles tem a “ordem perfeita de eliminação”!

Grafos cordais, por exemplo

Eles tem a “ordem perfeita de eliminação”!

Vamos refrescar a mente dos colegas:

Uma ordenação de um grafo `G` é dita OPE somente se:

Todo `V[i]` é simplicial no subgrafo induzido `G[ V[i..n] ]`.

Uma ordenação de um grafo `G` é dita OPE somente se:

Todo `V[i]` é simplicial no subgrafo induzido `G[ V[i..n] ]`.

Ou seja, a vizinhança de `V[i]` é clique no grafo **SEM** os vértices que vem antes dele na ordenação.

Se invertemos esta ordem,

Todo `V[i]` é simplicial no subgrafo induzido `G[ V[1..i] ]`.

Se invertemos esta ordem,

Todo `V[i]` é simplicial no subgrafo induzido `G[ V[1..i] ]`.

Ou seja, a vizinhança de `V[i]` é clique no grafo **SEM** os vértices que vem **DEPOIS** dele na ordenação.

Se invertemos esta ordem,

Todo `V[i]` é simplicial no subgrafo induzido `G[ V[1..i] ]`.

Ou seja, a vizinhança de `V[i]` é clique no grafo **SEM** os vértices que vem **DEPOIS** dele na ordenação.

(Está sempre numa clique!)

O algoritmo guloso de coloração

```
1    Para cada vértice V[i], na ordem, faça:
2    Atribua a menor cor (inteiro positivo) à V[i] que não tenha sido atribuída a vizinhos V[j] onde “j < i”.
```

Por quê que esse algoritmo é ótimo nos cordais?

Se um vértice foi pintado com a cor $k > 1$, ele pertence à uma clique de tamanho pelo menos k.

Por quê que esse algoritmo é ótimo nos cordais?

Se um vértice foi pintado com a cor `k > 1`, ele pertence à uma clique de tamanho pelo menos `k`.

Existe pelo menos `k-1` vizinhos **menores** do que ele na ordenação que tem cores `k-1, k-2, …`, sendo impossível ele ser pintado com um número maior que seu número de vizinhos.

Por quê que esse algoritmo é ótimo nos cordais?

Se um vértice foi pintado com a cor `k > 1`, ele pertence à uma clique de tamanho pelo menos `k`.

Existe pelo menos `k-1` vizinhos **menores** do que ele na ordenação que tem cores `k-1, k-2, …`, sendo impossível ele ser pintado com um número maior que seu número de vizinhos.

E também, pelo fato de que é cordal e o vértice é simplicial, todos os seus vizinhos que vem antes DEVEM formar uma clique!

Curiosidade

Já que a clique de tamanho K só aparece quando a cor K aparece, e vice-versa, o número cromático é igual ao tamanho da maior clique - definição de perfeito!

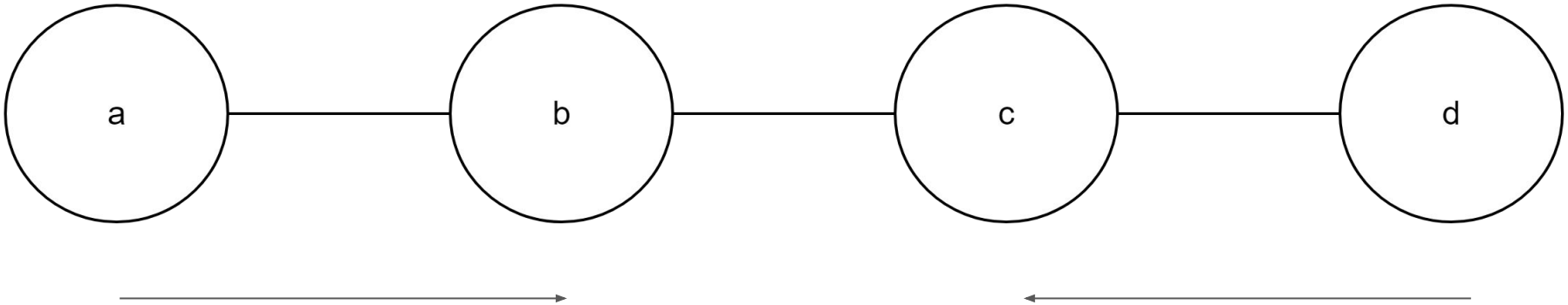
Enfim, a classe de grafos da apresentação de hoje

Os **grafos perfeitamente ordenáveis** são aqueles que admitem ordenação onde **TODOS OS SEUS SUBGRAFOS** podem ter o coloração ótima obtido pelo algoritmo guloso usando a **mesma ordenação**.

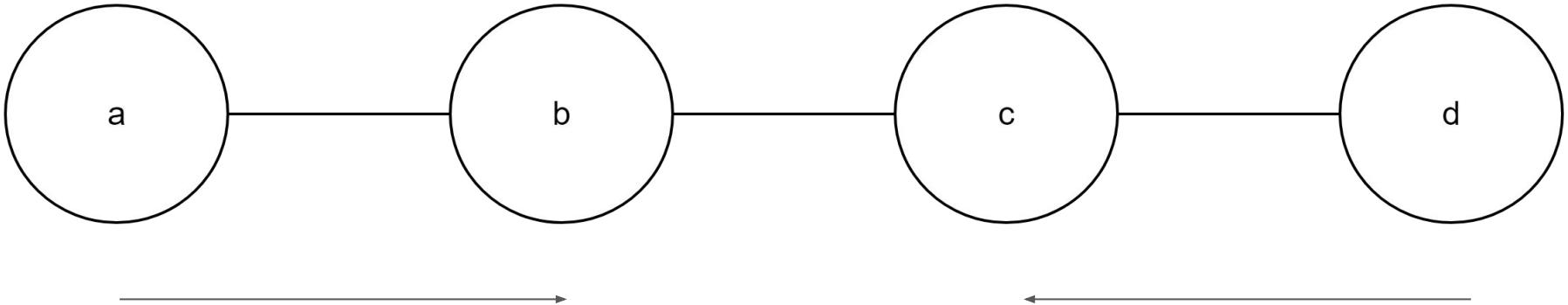
Entretanto, também temos outra definição que ajuda o reconhecimento.

Obstructions / obstruções

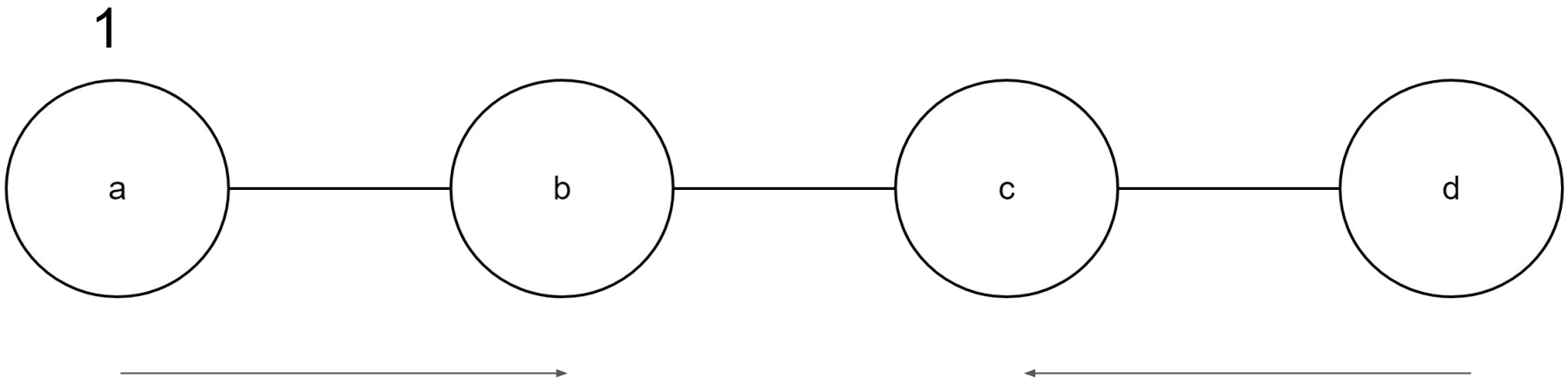
Uma obstrução é um P4 **sem cordas** - vértices (a,b,c,d)- no grafo com ordenação “<” onde, “**a < b**” e “**d < c**” ocorrem ao mesmo tempo.



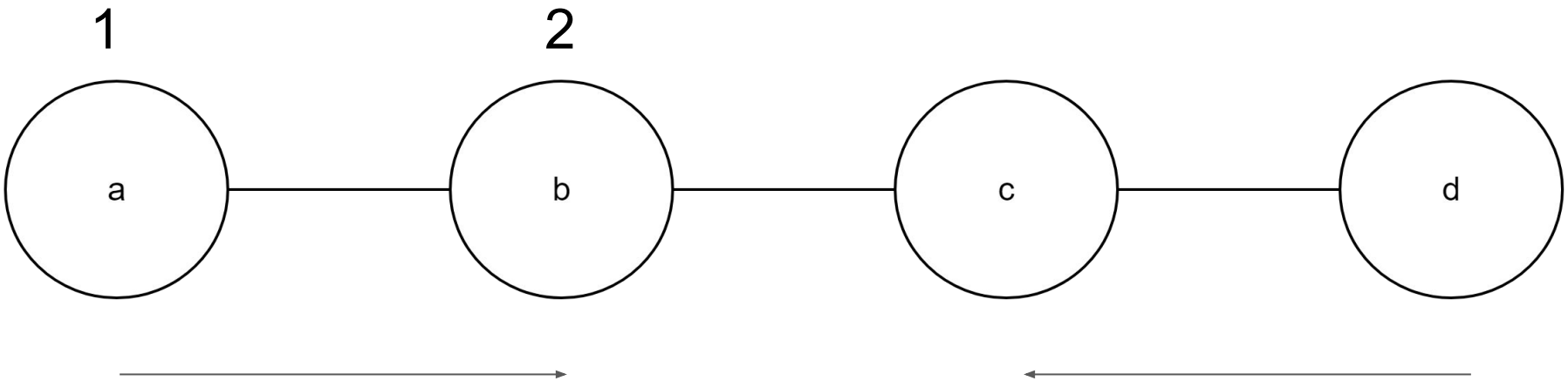
$$a < b < d < c$$



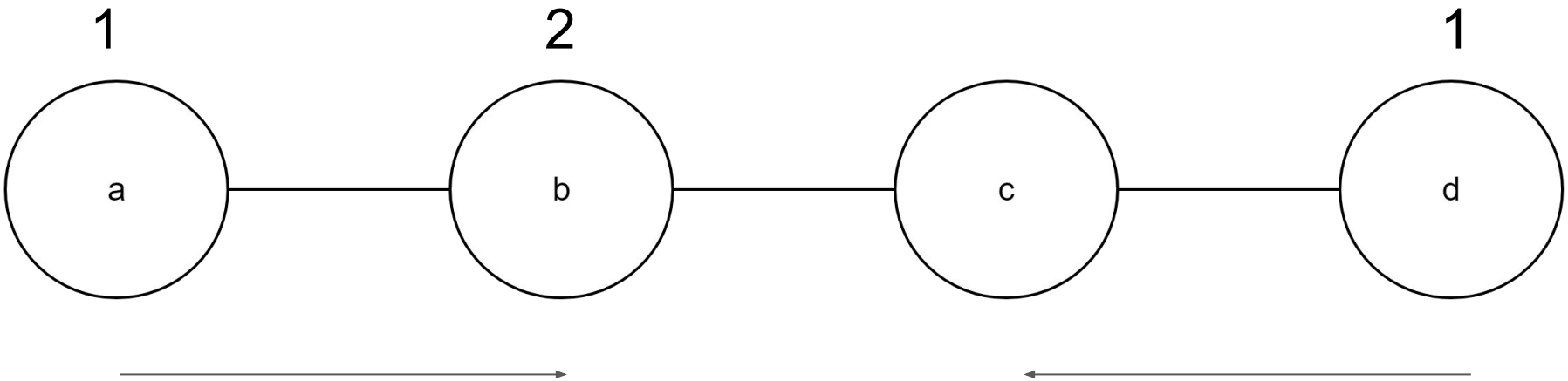
a < b < d < c



a < b < d < c

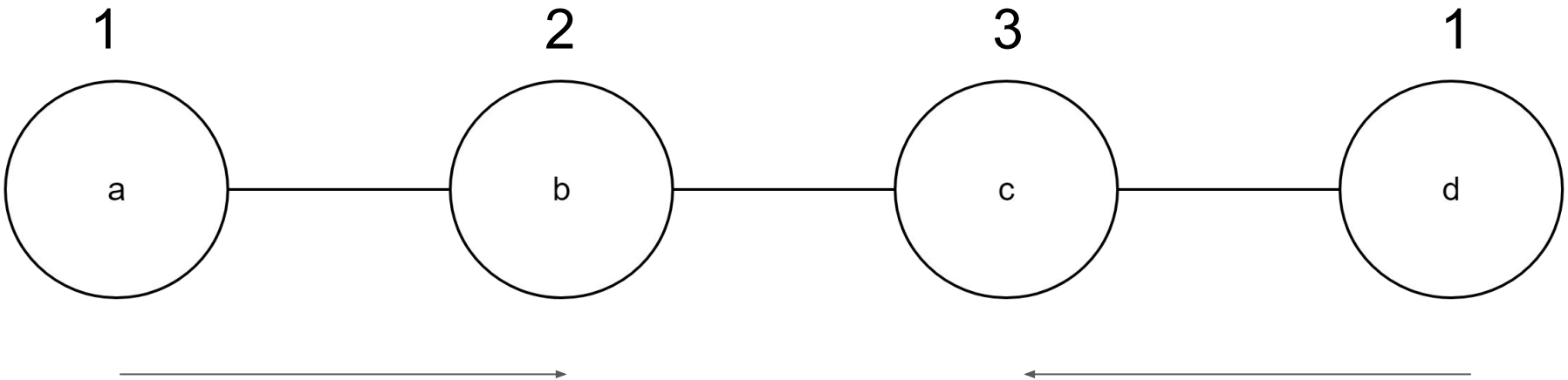


a < b < d < c



a < b < d < c

??????????????????????????



Outra definição

Os grafos perfeitamente ordenáveis são aqueles que admitem ordenação **SEM OBSTRUÇÕES** induzidas.

Outra definição

Tem ordenação em que o algoritmo funciona? Então ela não tem obstruções, pois se tivesse, a situação descrita anteriormente acontece.

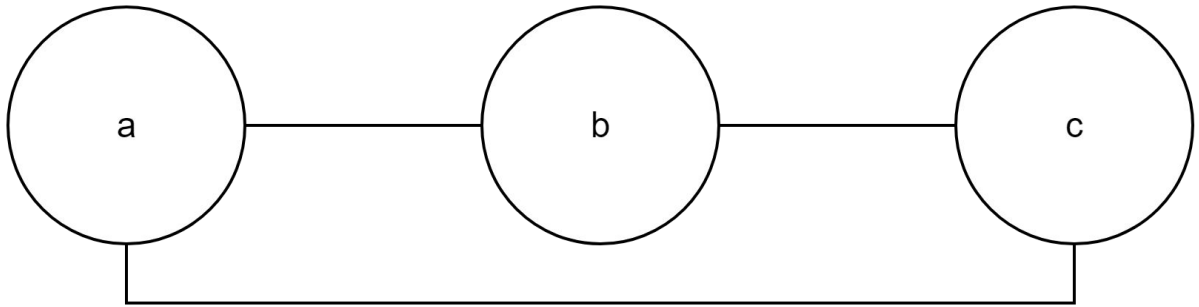
Outra definição

Tem ordenação em que o algoritmo funciona? Então ela não tem obstruções, pois se tivesse, a situação descrita anteriormente acontece.

A ordenação não tem obstruções? Então podemos atribuir cores aos vértices sem ter que dar meia volta e mesmo assim a coloração será ótima.

Algumas classes perfeitamente ordenáveis

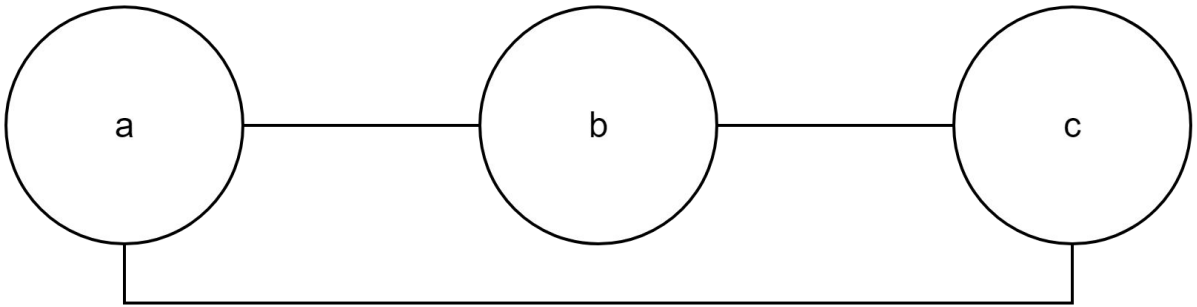
Grafos de comparabilidade não tem sequer nenhum P4 sem cordas induzido.



Algumas classes perfeitamente ordenáveis

Grafos de comparabilidade não tem sequer nenhum P4 sem cordas induzido.

Se “a < b” e “b < c”, então existe a comparação “a < c”, dando uma corda ao caminho.



MOMENTO PEGADINHA: *soft vertices*

Definition 28.24 *For a P_4 with vertices a, b, c, d, edges ab, bc, cd, the vertices a, d are* endpoints, *c, d are* midpoints *of the P_4. A vertex is* soft *if it is not a midpoint or an endpoint of a P_4.*

Perfect Graphs; Chinh T. Hoang, Wilfrid Laurier University and R. Sritharan, University of Dayton

A P_4 is a chordless path on four vertices; if w–x–y–z is a P_4, we call w and z *endpoints* and we call x and y *midpoints*. A vertex v is *soft* in G if either it does not occur as an endpoint of a P_4 or it does not occur as a midpoint of a P_4.

Recognizing brittle graphs: remarks on a paper of Hoàng and Khouzam; Alejandro A. Schiiffer, A T&T Bell Laboratories

Definition:

[...] A vertex that is not the endpoint of any P_4, or a vertex that is not the midpoint of any P_4.

https://www.graphclasses.org/classes/gc_10.html

Sim.

Pode estar em um P4.

Desde que não seja *midpoint* de um P4 ao mesmo tempo que é *endpoint* de outro P4.

Graphclass: brittle

Definition:

A graph G is brittle iff each induced subgraph H of G contains either a vertex that is not the endpoint of any P_4 of H, or a vertex that is not the midpoint of any P_4 of H.

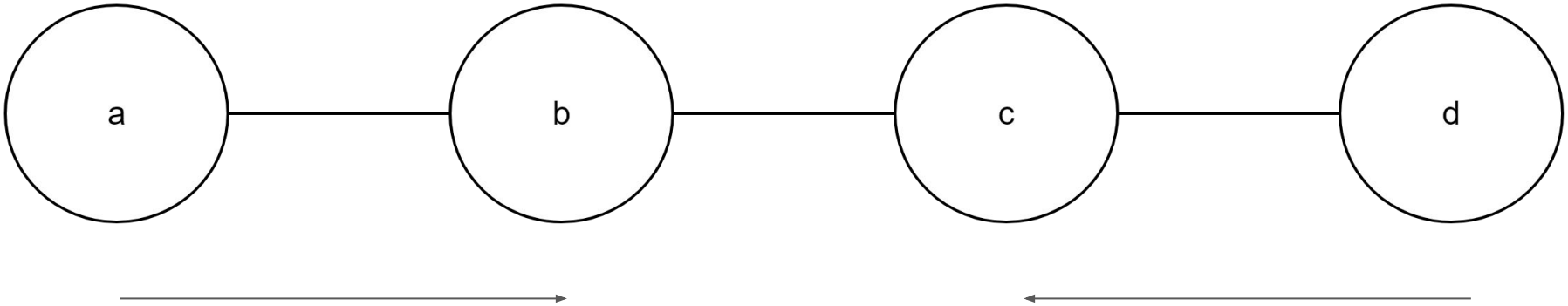
Se G tem vértice v[n] soft e (G - {v[n]}) é perfeitamente ordenável, G também é perfeitamente ordenável.

Se G tem vértice `v[n]` soft e (G - {`v[n]`}) é perfeitamente ordenável, G também é perfeitamente ordenável.

Digamos que `V[1..n-1]` é ordem perfeita.

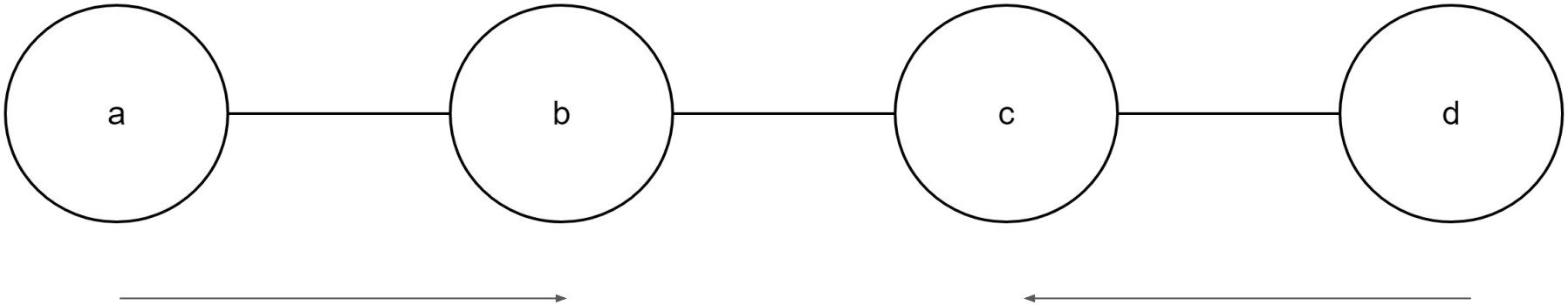
Então:

1. Se `v[n]` não é *endpoint* de um P4, a ordem (`v[n]`,`v[1]`,`v[2]`,...,`v[n-1]`) é perfeita.



Então:

2. Se `v[n]` não é *midpoint* de um P4, a ordem (`v[1]`,`v[2]`,...,`v[n-1]`,`v[n]`) é perfeita.



Grafos *brittle* são perfeitamente ordenáveis.

Indução na propriedade anterior.

Grafos *brittle* são perfeitamente ordenáveis.

Indução na propriedade anterior.

Se G é *brittle*, todo subgrafo induzido tem um vértice *soft*. Assim, para construir G perfeitamente ordenável, coloque os vértices na ordem em que a propriedade anterior seja respeitada.

Os grafos cordais são perfeitamente ordenáveis

Note que os grafos cordais são *brittle*, pois os vértices simpliciais são *soft*: eles não podem* pertencer a um P4 sem cordas pois sua vizinhança forma um completo!

Lado triste

Computational complexity

Perfectly orderable graphs are NP-complete to recognize.[2] However, it is easy to test whether a particular ordering is a perfect ordering of a graph. Consequently, it is also NP-hard to find a perfect ordering of a graph, even if the graph is already known to be perfectly orderable.

Lado positivo

Já vimos que existem classes mais fáceis de serem reconhecidas que são perfeitamente ordenáveis.

Cordais, comparabilidade, *brittle…*

Mais vantagens

Se temos a ordem perfeita de um grafo G, alguns problemas que antes eram complicados, agora ficam simples!

1) O problema da colorização é O(n + m).

No algoritmo guloso, para colorir os n vértices, temos apenas que ver os seus vizinhos que vem antes na ordem. O problema fica linear!

2) Encontrar a maior clique do grafo é um problema linear

Após a colorização, teremos o número cromático k.

Encontre o vértice de cor k e coloque ele no seu conjunto C.

Percorra a ordem perfeita ao contrário, a partir deste vértice. Vá colocando todos os vizinhos que vem antes dele e que são vizinhos deste vértice.

3) Encontrar a menor *clique cover* para um grafo co-perfeitamente ordenável é um problema linear.

Se temos G perfeitamente ordenável e sua negação ¬G, aplicamos a coloração gulosa em G (custo linear).

Essa colorização dos vértices em G forma a menor quantidade de cliques em ¬G, quando juntamos os vértices de mesma cor em G.

3) Encontrar a menor *clique cover* para um grafo co-perfeitamente ordenável é um problema linear.

Se temos G perfeitamente ordenável e sua negação ¬G, aplicamos a coloração gulosa em G (custo linear).

Essa colorização dos vértices em G forma a menor quantidade de cliques em ¬G, quando juntamos os vértices de mesma cor em G.

Se em G podemos colorir o vértice v usando a cor i, é porque em G não existe vizinho de cor i. Logo, em ¬G, este vértice é vizinho de todos os vértices de cor i.

3) Encontrar a menor *clique cover* para um grafo co-perfeitamente ordenável é um problema linear.

Se temos G perfeitamente ordenável e sua negação ¬G, aplicamos a coloração gulosa em G (custo linear).

Essa colorização dos vértices em G forma a menor quantidade de cliques em ¬G, quando juntamos os vértices de mesma cor em G.

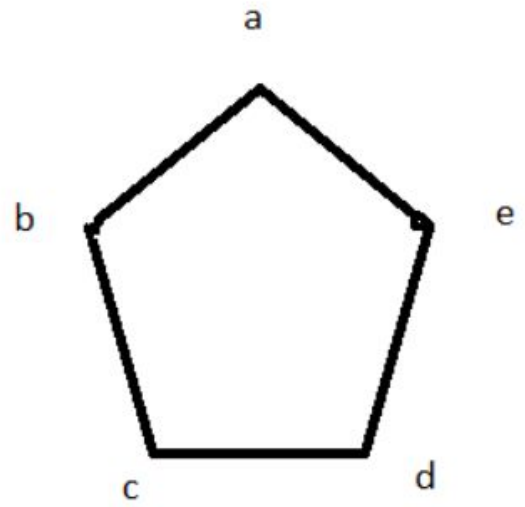
Se em G podemos colorir o vértice v usando a cor i, é porque em G não existe vizinho de cor i. Logo, em ¬G, este vértice é vizinho de todos os vértices de cor i.

Assim, ao atribuir para v o menor i tal que v é vizinho de todos os outros de sua cor, obtemos as maiores cliques.

4) Encontrar o maior conjunto independente para um grafo co-perfeitamente ordenável é linear.

Basta encontrar a maior clique de seu complemento perfeitamente ordenável, que é um problema linear, como visto anteriormente.

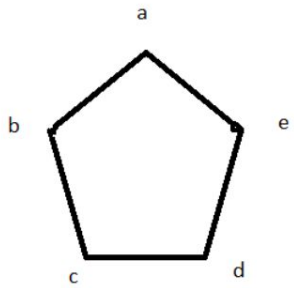
Exemplo rápido: grafo não perfeitamente ordenável



https://math.stackexchange.com/questions/3922738/perfect-graphs-and-perfectly-orderable-graphs

Dica:
Precisamos apenas checar os subgrafos induzidos de tamanho maior que 4.

(Subgrafos menores que 4 não podem ter obstrução!)



Alguém tem que ser o primeiro, e como os vértices são iguais, não importa o nome. Vamos chamá-lo de *a*.

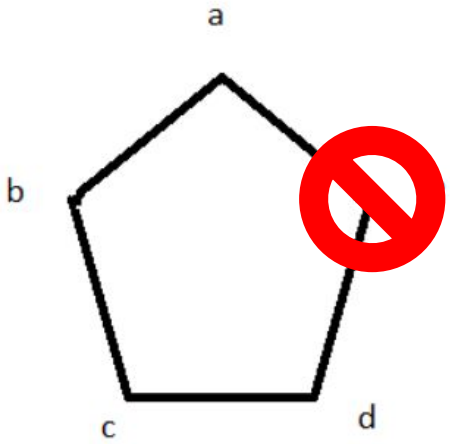
Primeiro = a.

Se c > d:

Quando tiramos o vértice “e” do grafo, a-b-c-d formam P4.

Como a é o primeiro, a < b.

(a < b) e (d < c) formam uma obstrução!



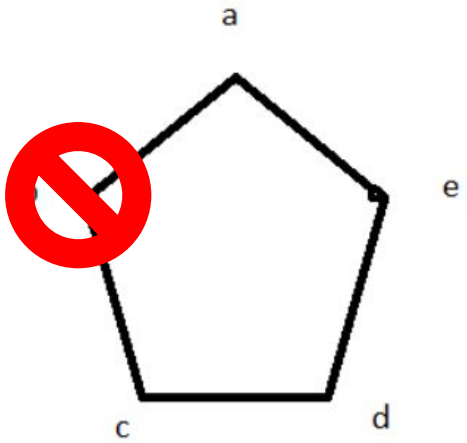
Primeiro = a.

~~Se c > d~~ Se c < d:

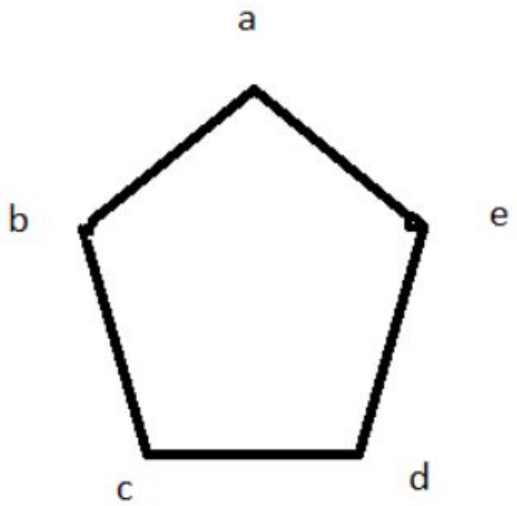
Quando tiramos o vértice “b” do grafo, a-e-d-c formam P4.

Como a é o primeiro, a < e.

(a < e) e (c < d) formam uma obstrução!



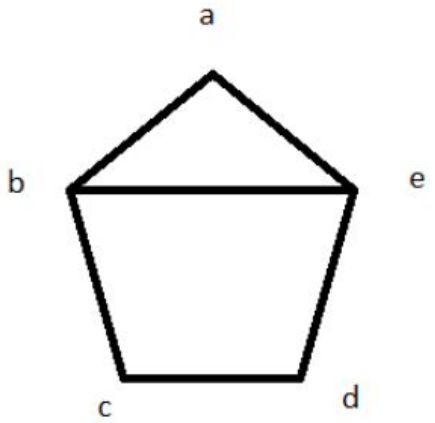
Conclusão



Não conseguimos fazer uma ordenação deste grafo que não tenha obstrução!

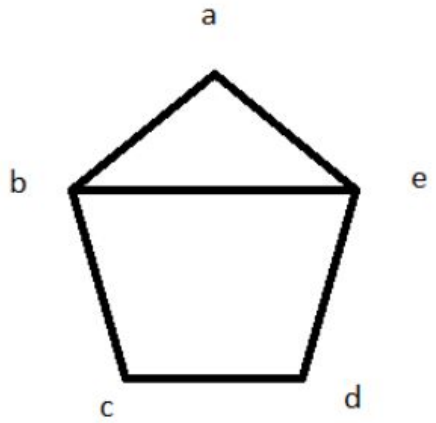
Logo, não é perfeitamente ordenável.

Exemplo rápido: grafo perfeitamente ordenável



https://math.stackexchange.com/questions/3922738/perfect-graphs-and-perfectly-orderable-graphs

Exemplo rápido: grafo perfeitamente ordenável



A ordem “c < d < e < b < a” é perfeita!

https://math.stackexchange.com/questions/3922738/perfect-graphs-and-perfectly-orderable-graphs

Dica:

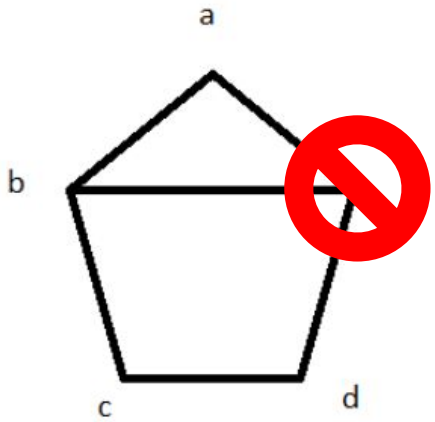
Podemos diretamente verificar apenas os caminhos sem corda e ver se existe alguma obstrução para a ordem.

No caso, apenas 2 subgrafos de tamanho 4 são caminhos sem corda!

Verificando a ordem “c < d < e < b < a”

Para o caminho a-b-c-d:

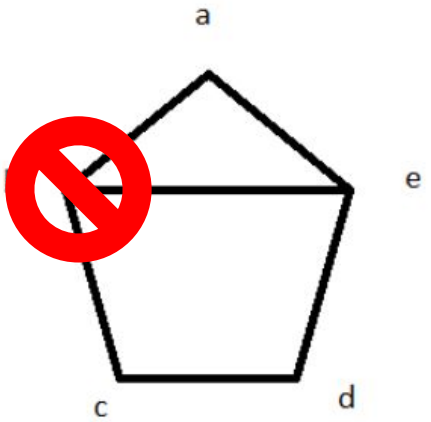
(c < d), então não temos obstrução.



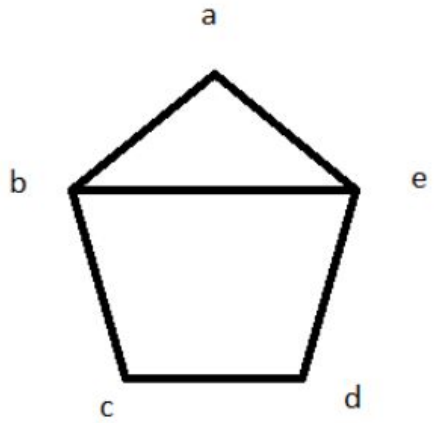
Verificando a ordem “c < d < e < b < a”

Para o caminho a-e-d-c:

(e < a), então não temos obstrução.



Conclusão



A ordem “c < d < e < b < a” não forma nenhuma obstrução, então é perfeita!

https://math.stackexchange.com/questions/3922738/perfect-graphs-and-perfectly-orderable-graphs

Referências

1. A NOTE ON PERFECT ORDERS; C.T. HOANG, Department of Computer Science, Rutgers University, New Brunswick, New Jersey 08903, USA and N.V.R. MAHADEV,
Department of Mathematics, Northeastern University, Boston, MA 02115, USA

2. https://en.wikipedia.org/wiki/Perfectly_orderable_graph
3. https://www.graphclasses.org/classes/gc_32.html

4.Chính T Hoàng, Stefan Hougardy, Frédéric Maffray, N.V.R Mahadev,
On simplicial and co-simplicial vertices in graphs,
Discrete Applied Mathematics,
Volume 138, Issues 1–2,
2004,
Pages 117-132,
ISSN 0166-218X

5. Perfect Graphs; Chinh T. Hoang, Wilfrid Laurier University and R. Sritharan, University of Dayton