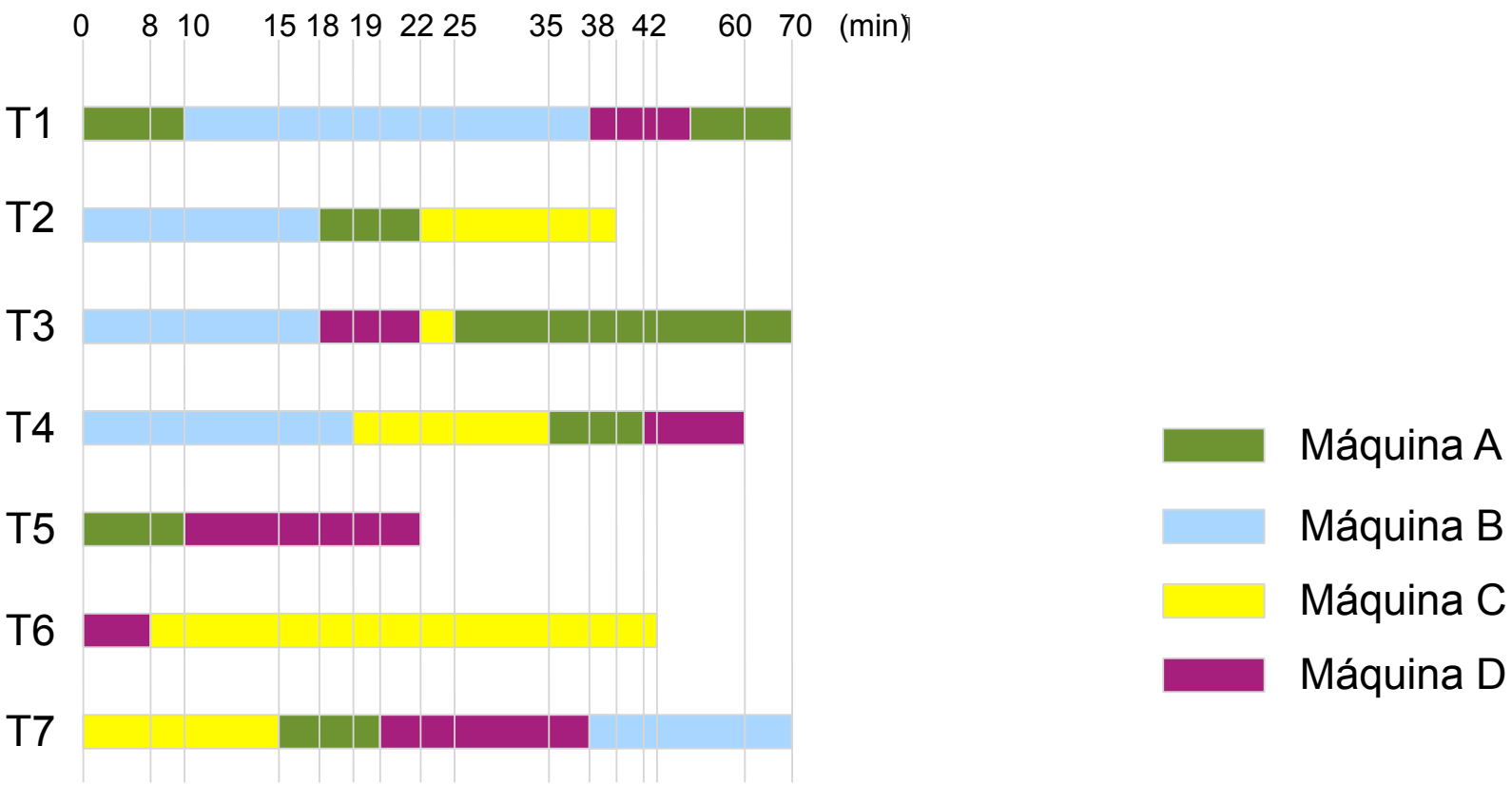
Algoritmos aproximativos: o problema do caixeiro-viajante

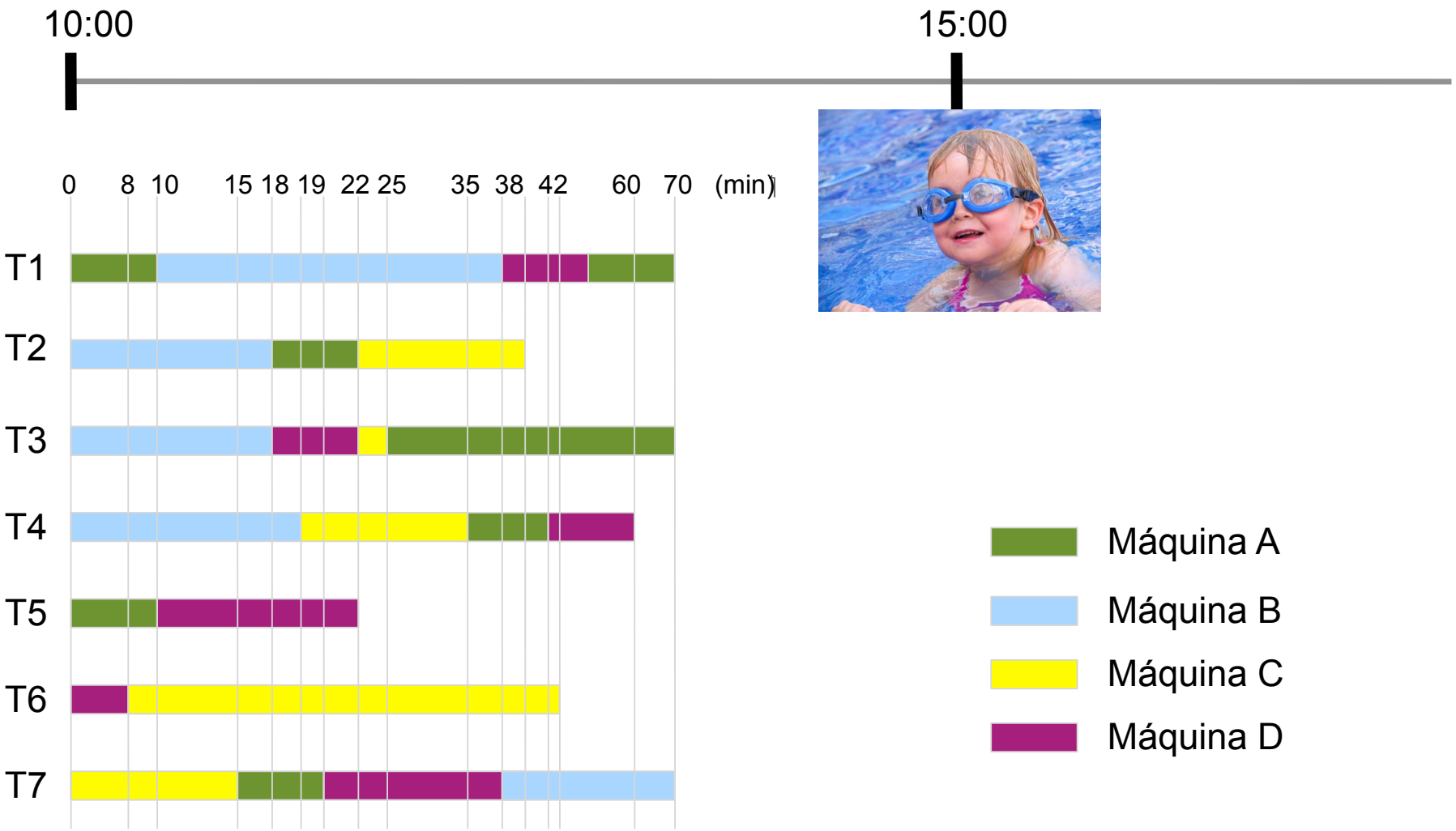
Universidade Federal do Rio de Janeiro

Vinícius Gusmão Pereira de Sá

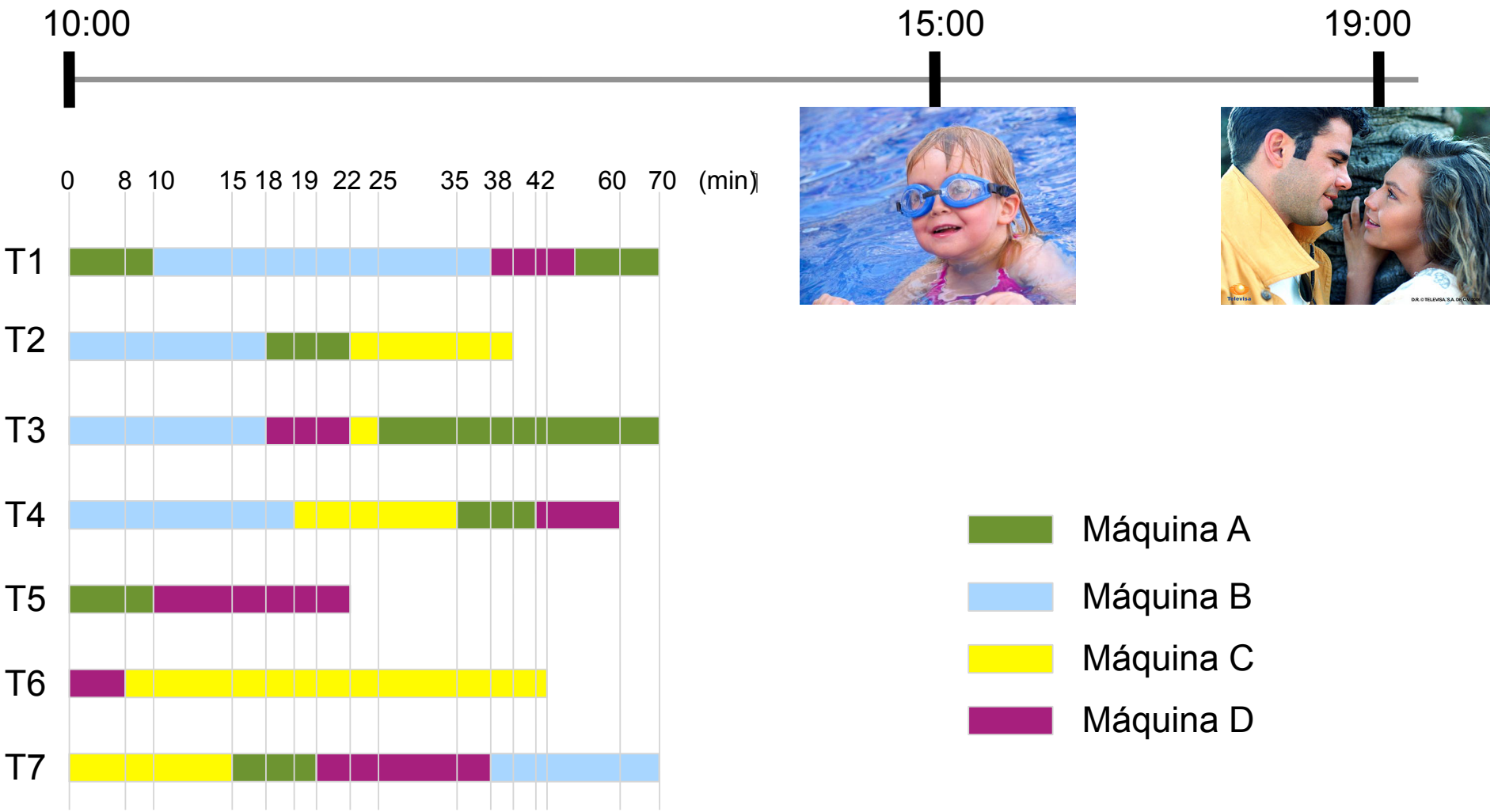
Algoritmos aproximativos



Algoritmos aproximativos



Algoritmos aproximativos



Problemas NP-difíceis

→ pelo menos tão difíceis quanto qualquer problema em NP

→ provavelmente não é possível resolvê-los em tempo polinomial (a menos que P = NP)

→ força bruta demanda tempo exorbitante

→ heurísticas e algoritmos aproximativos

→ mochila, cobertura mínima por conjuntos, floresta de Steiner, conjunto independente máximo, escalonamento, caixeiro-viajante etc.

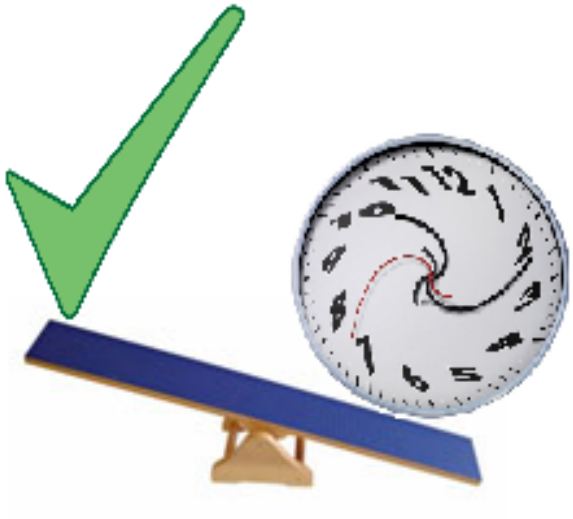
Algoritmos aproximativos

Problemas de minimização:

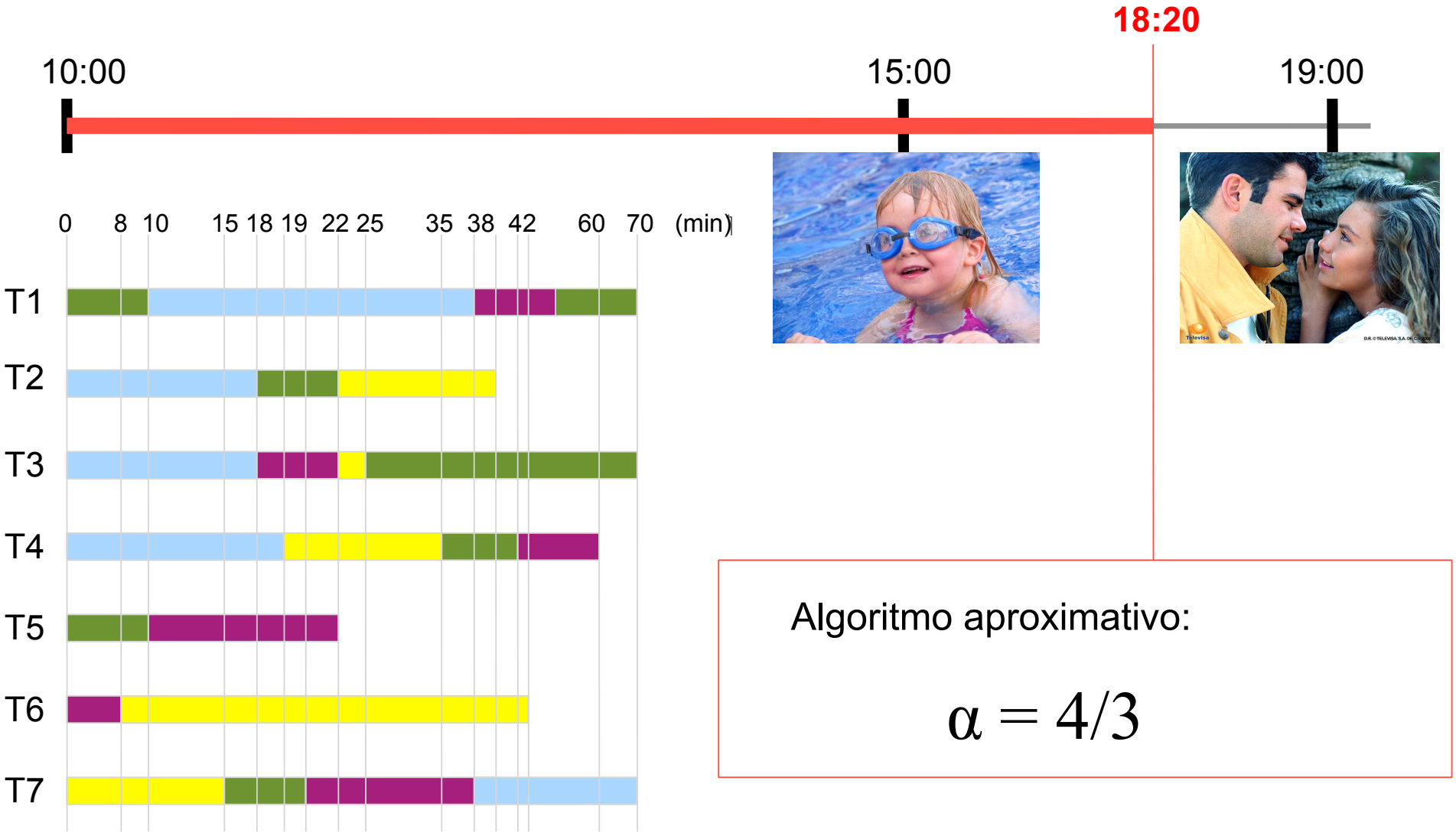
$$A(I) \leq \alpha \cdot opt(I) , \qquad \alpha \geq 1$$

Problemas de maximização:

$$A(I) \geq \alpha \cdot opt(I) , \qquad 0 < \alpha \leq 1$$



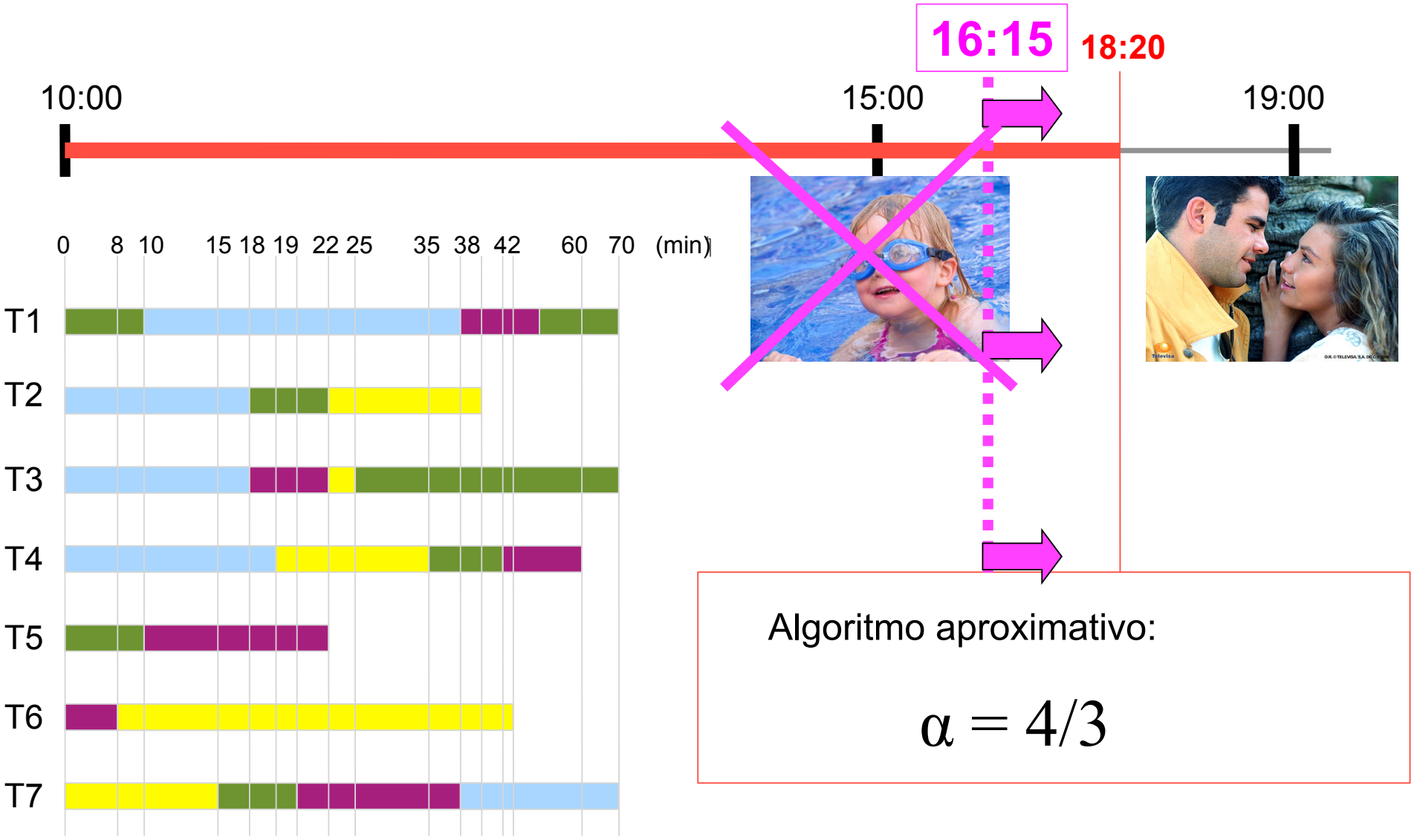
Algoritmos aproximativos



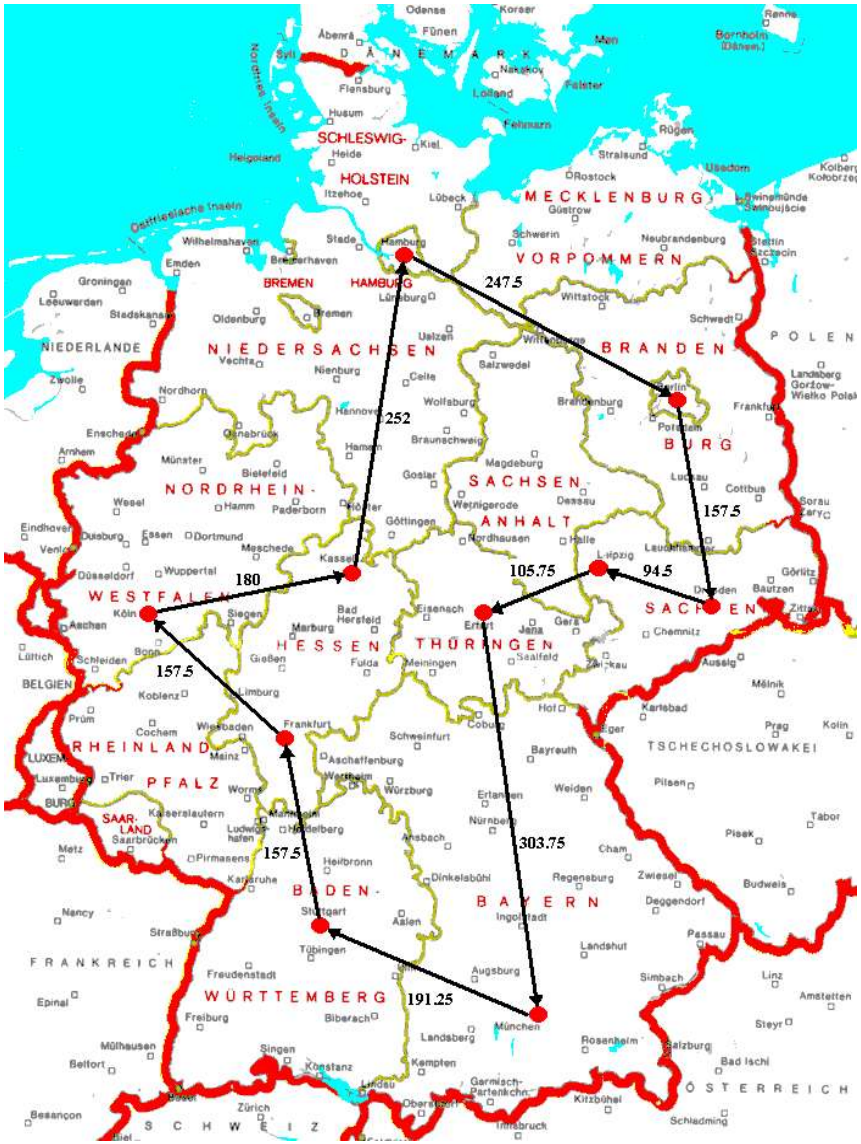
Algoritmo aproximativo:

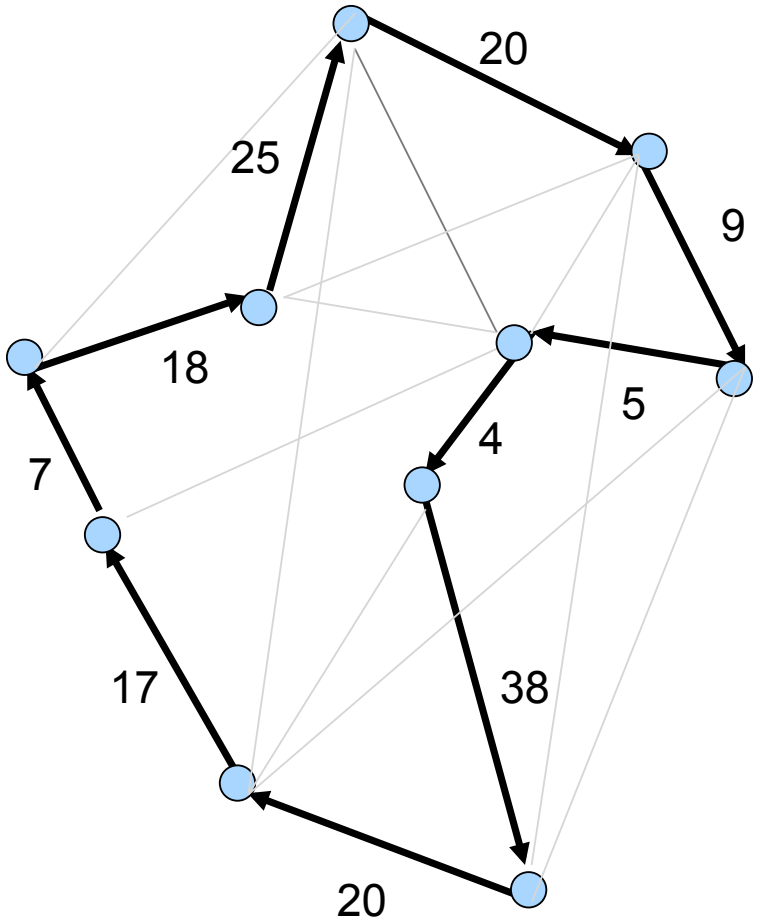
$$\alpha = 4/3$$

Algoritmos aproximativos



O problema do caixeiro-viajante





O problema do caixeiro-viajante

→ o mais famoso problema de otimização combinatória

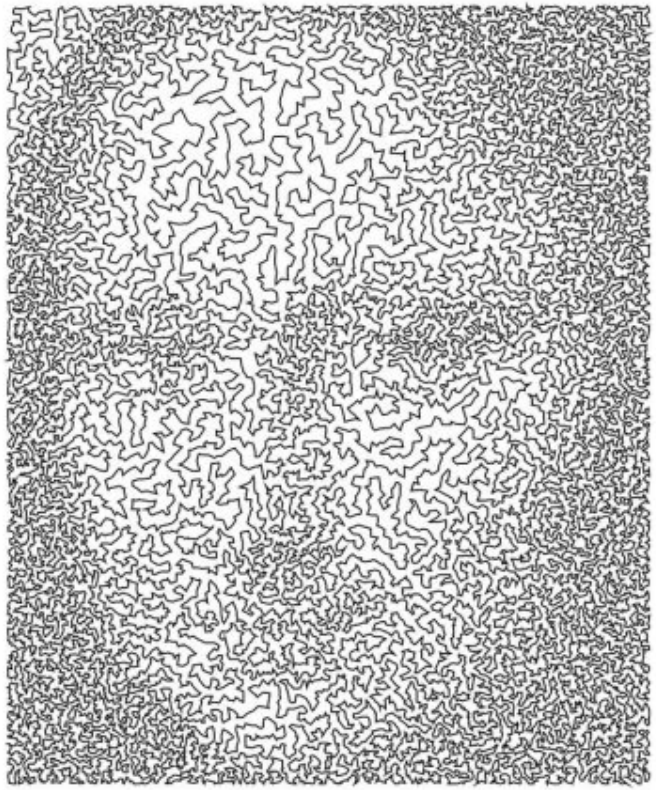
→ operações de máquinas em manufatura, otimização do movimento de ferramentas de corte, otimização de perfurações em placas de circuito impresso, roteamento de veículos, distribuição de tarefas etc.

→ ciclo hamiltoniano de custo mínimo

→ NP-difícil, mesmo se as arestas tiverem custos em {1,2}

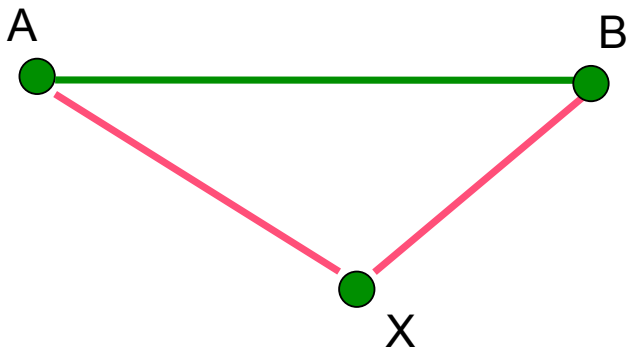
→ gama enorme de heurísticas

→ diversas formulações e problemas correlatos

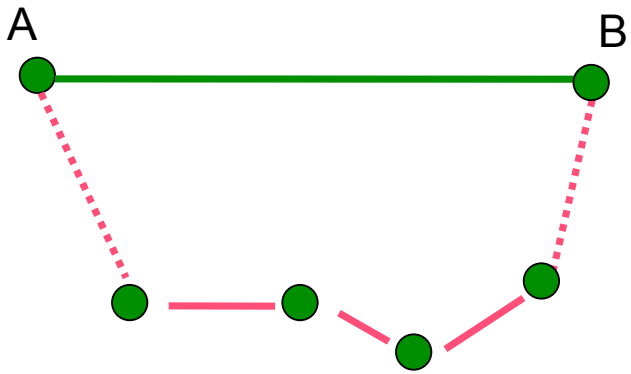


Caixeiro-viajante métrico (euclidiano)

→ grafo completo
→ custos das arestas satisfazem a desigualdade triangular



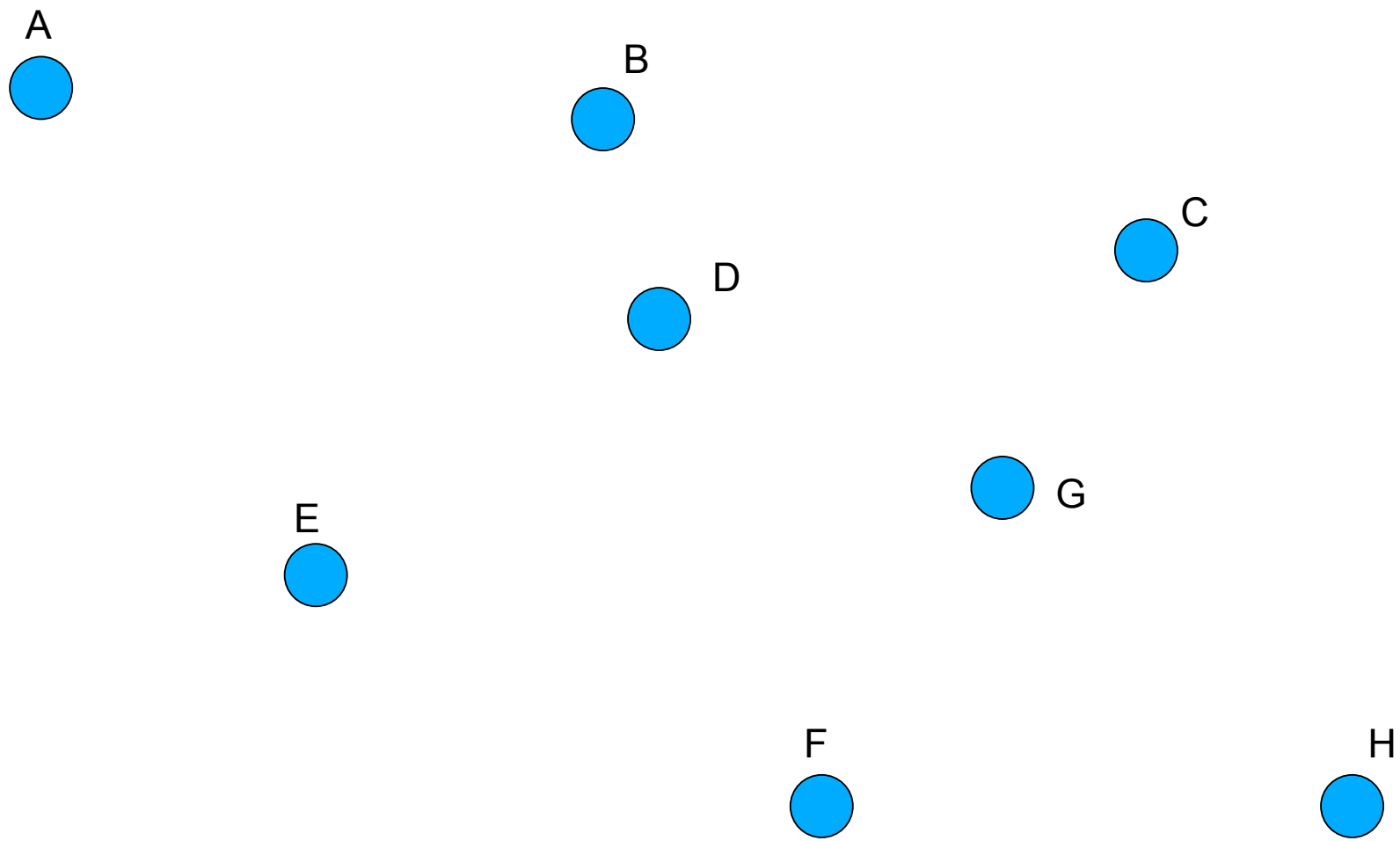
$$c(A,B) \leq c(A,X) + c(X,B)$$



O menor caminho entre A e B é sempre a própria aresta (A,B)

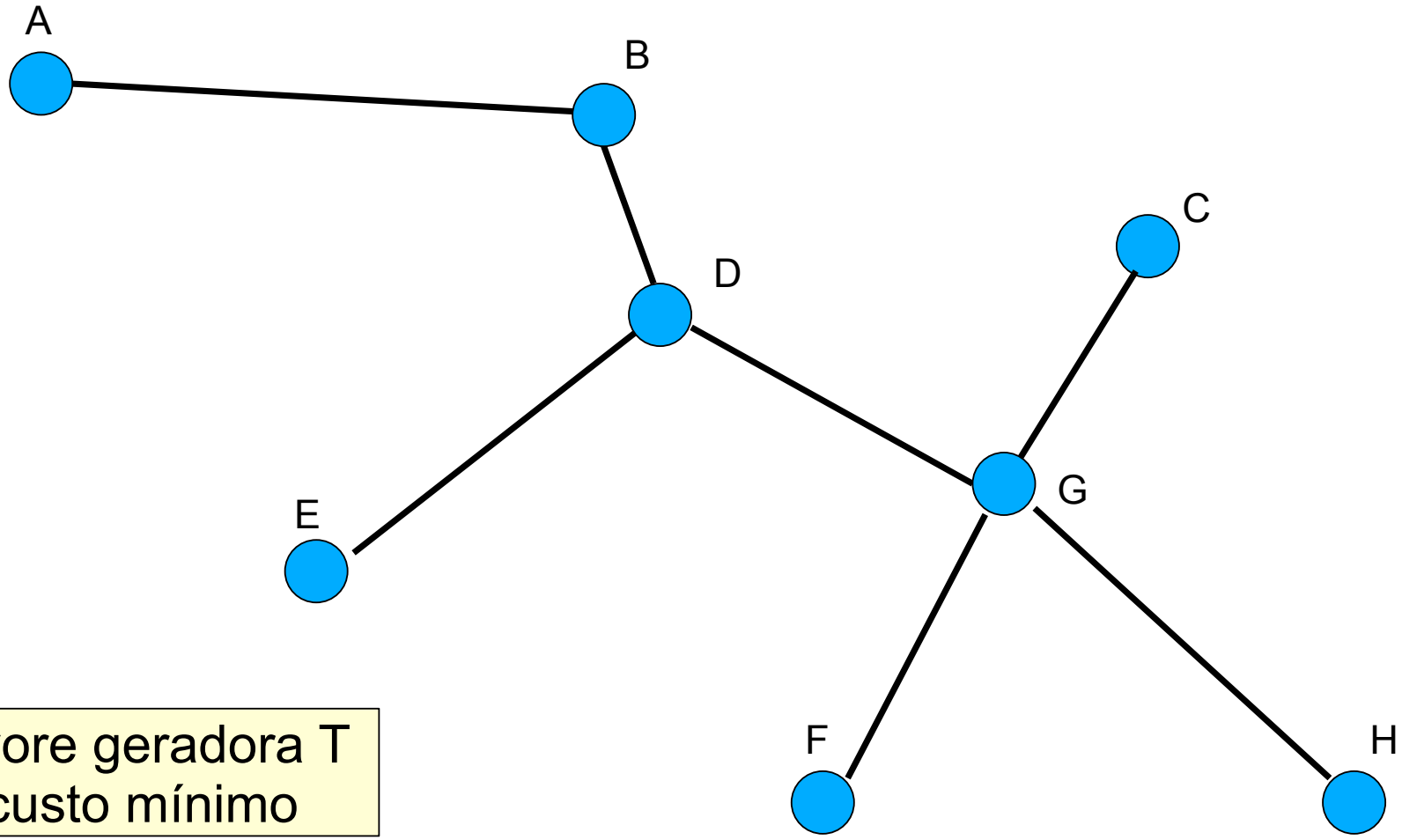
Algoritmo RSL

(Rosenkrantz, Stearns e Lewis)



Algoritmo RSL

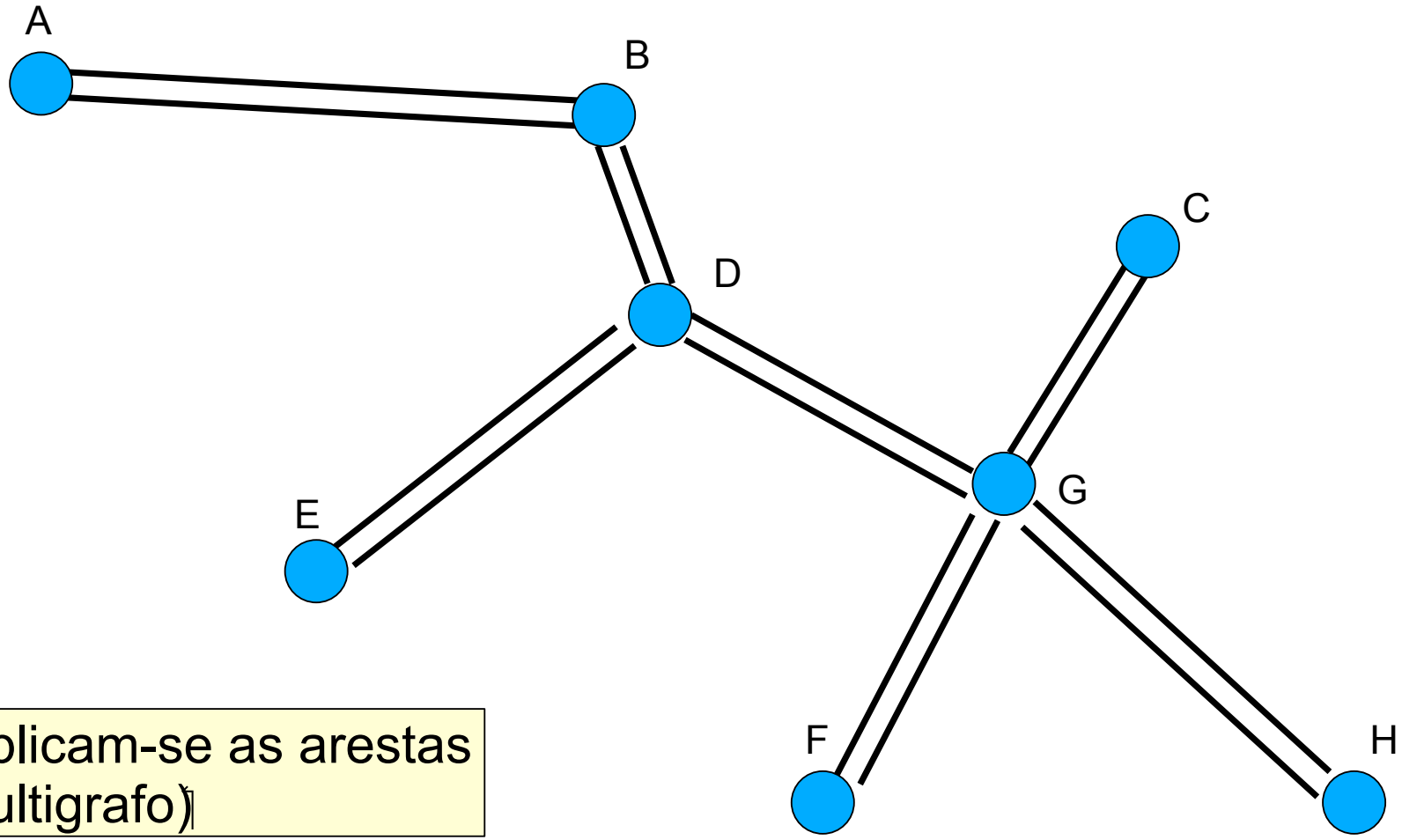
(Rosenkrantz, Stearns e Lewis)



1) árvore geradora T de custo mínimo

Algoritmo RSL

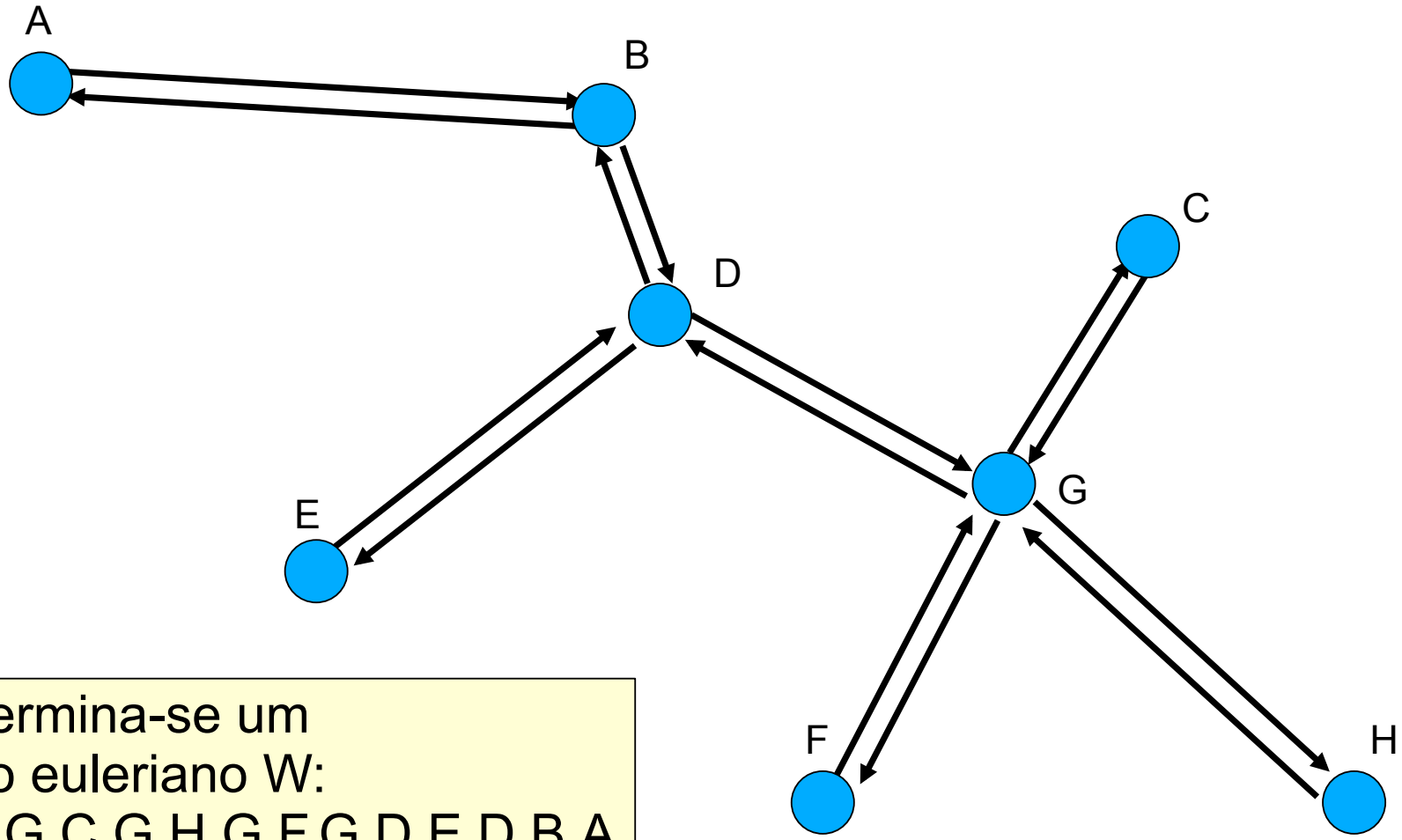
(Rosenkrantz, Stearns e Lewis)



2) duplicam-se as arestas (multigrafo)

Algoritmo RSL

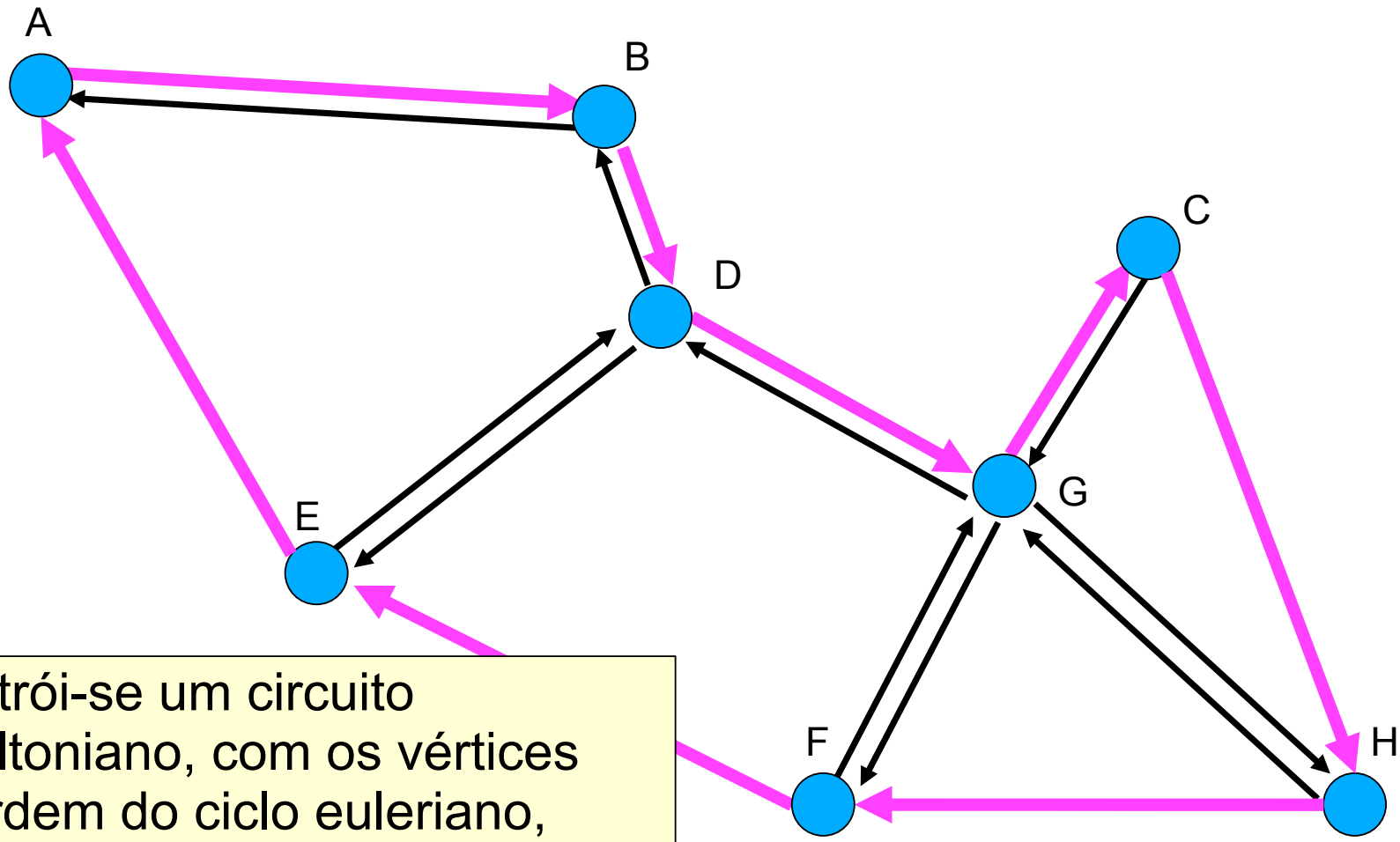
(Rosenkrantz, Stearns e Lewis)



3) determina-se um
ciclo euleriano W:
A,B,D,G,C,G,H,G,F,G,D,E,D,B,A

Algoritmo RSL

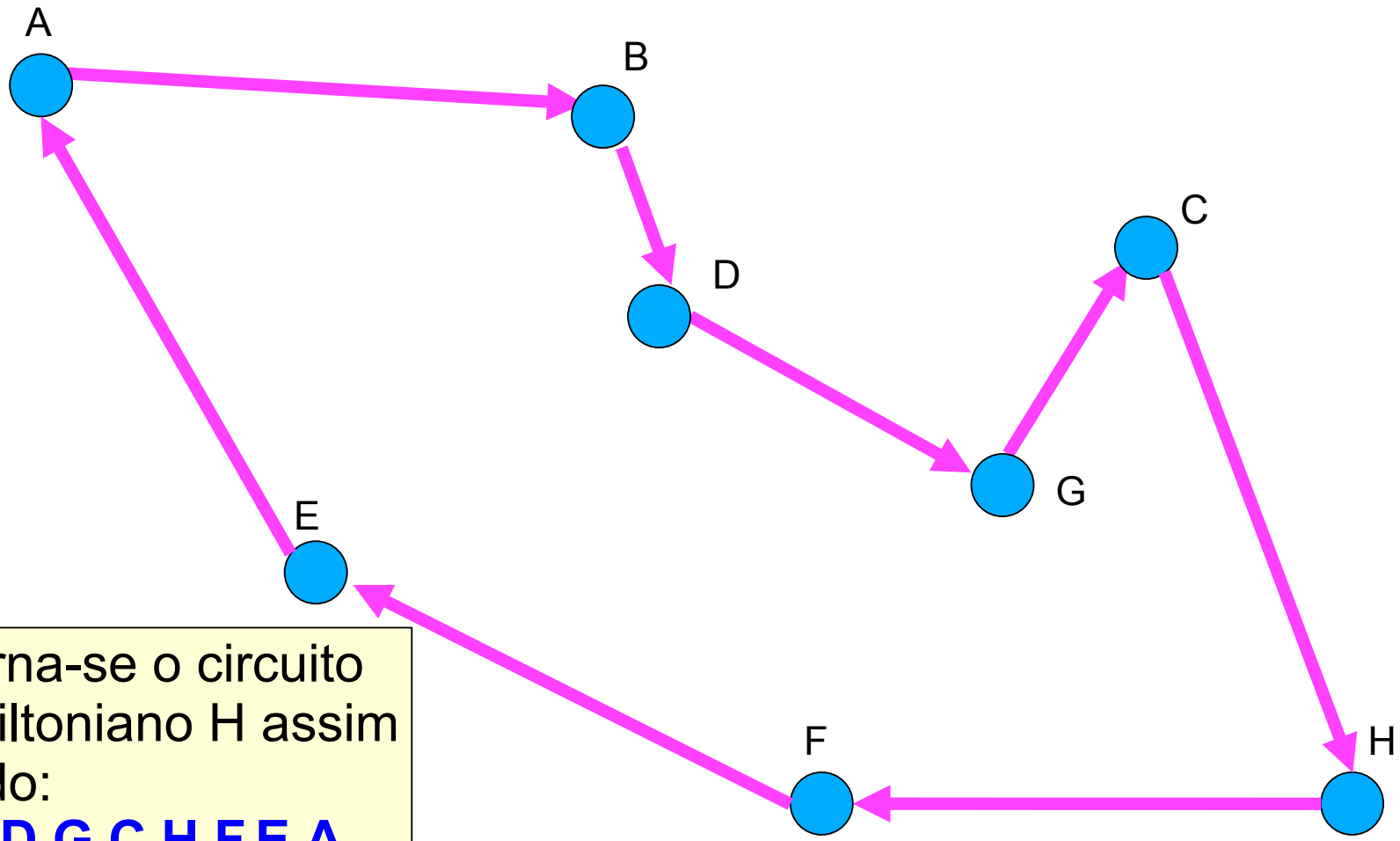
(Rosenkrantz, Stearns e Lewis)



4) constrói-se um circuito hamiltoniano, com os vértices na ordem do ciclo euleriano, sem repetições:
A,B,D,G,C,G,**H,**G,**F,**G,D,**E,**D,B,**A**

Algoritmo RSL

(Rosenkrantz, Stearns e Lewis)

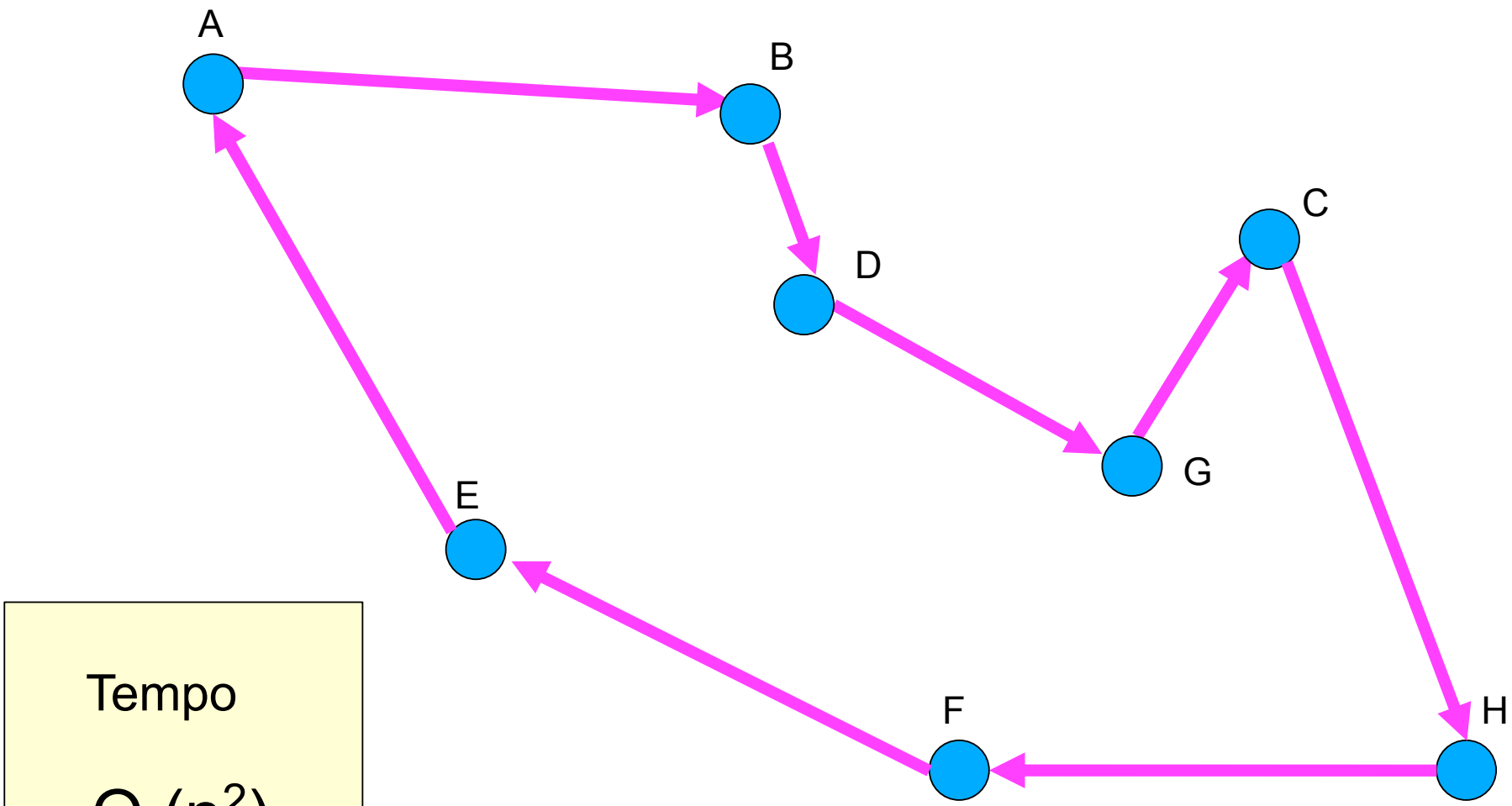


5) retorna-se o circuito hamiltoniano H assim obtido:
A,B,D,G,C,H,F,E,A

$c(H) \leq \mathbf{2} \cdot c(H^*)$

Algoritmo RSL

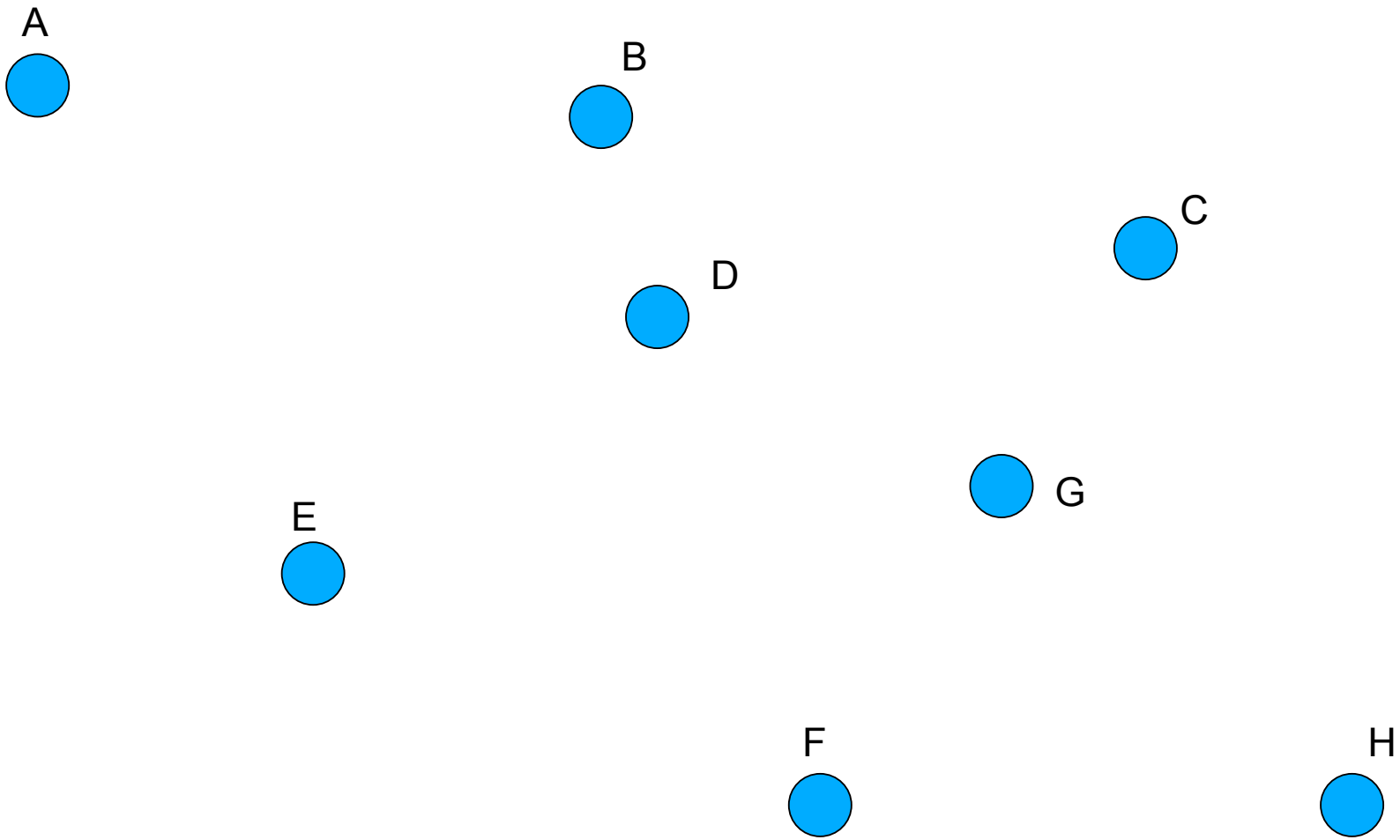
(Rosenkrantz, Stearns e Lewis)



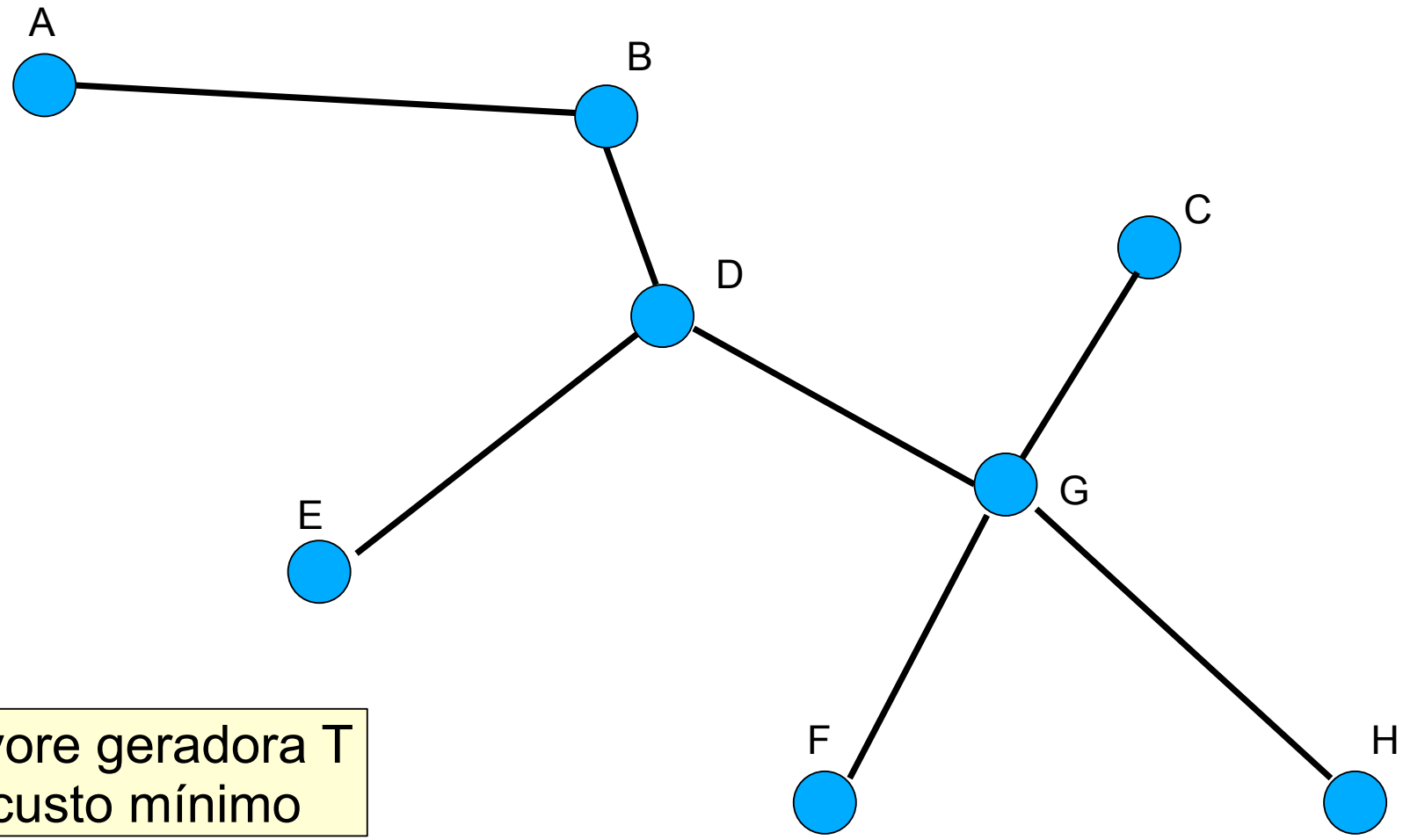
Tempo

$O(n^2)$

Algoritmo de Christofides

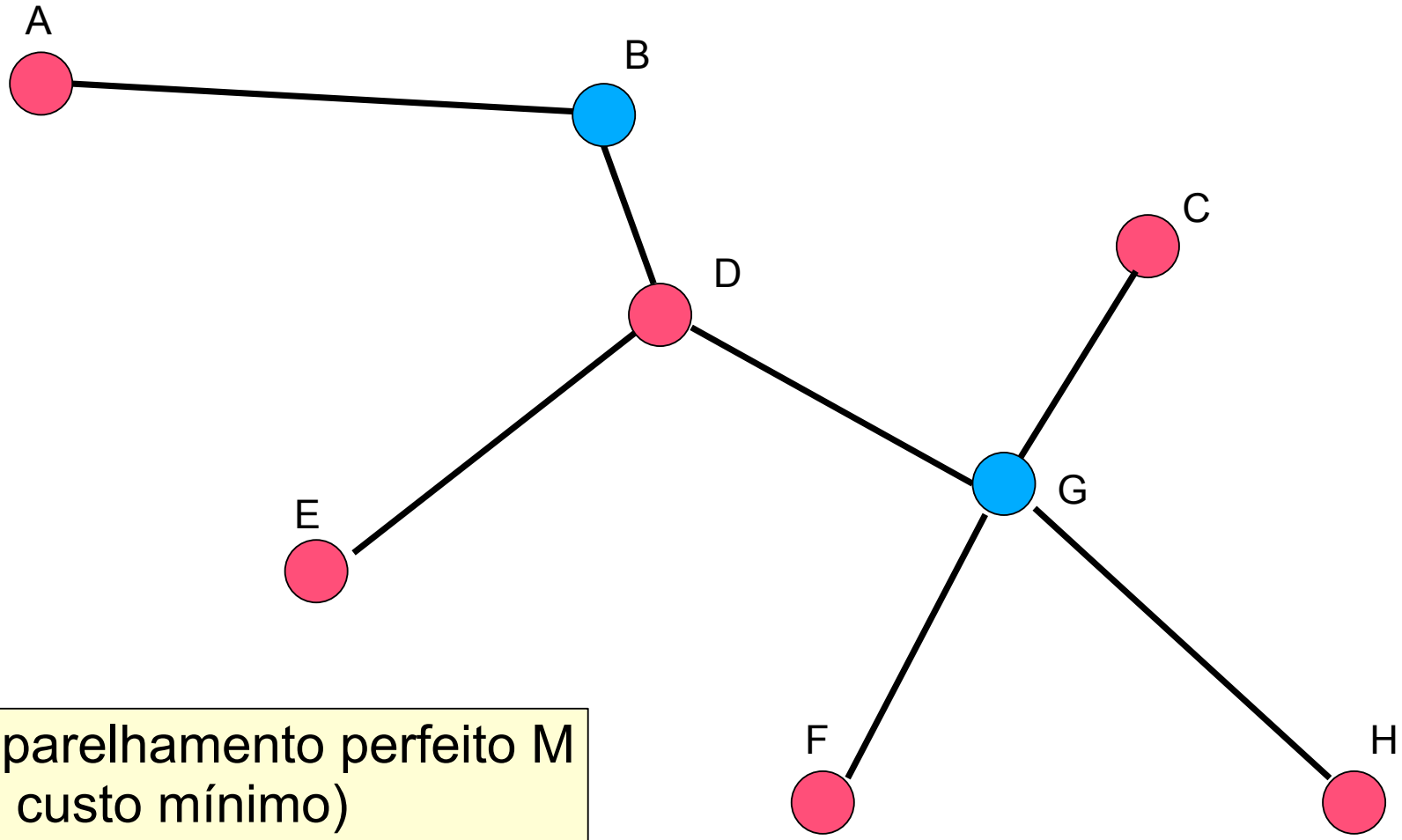


Algoritmo de Christofides



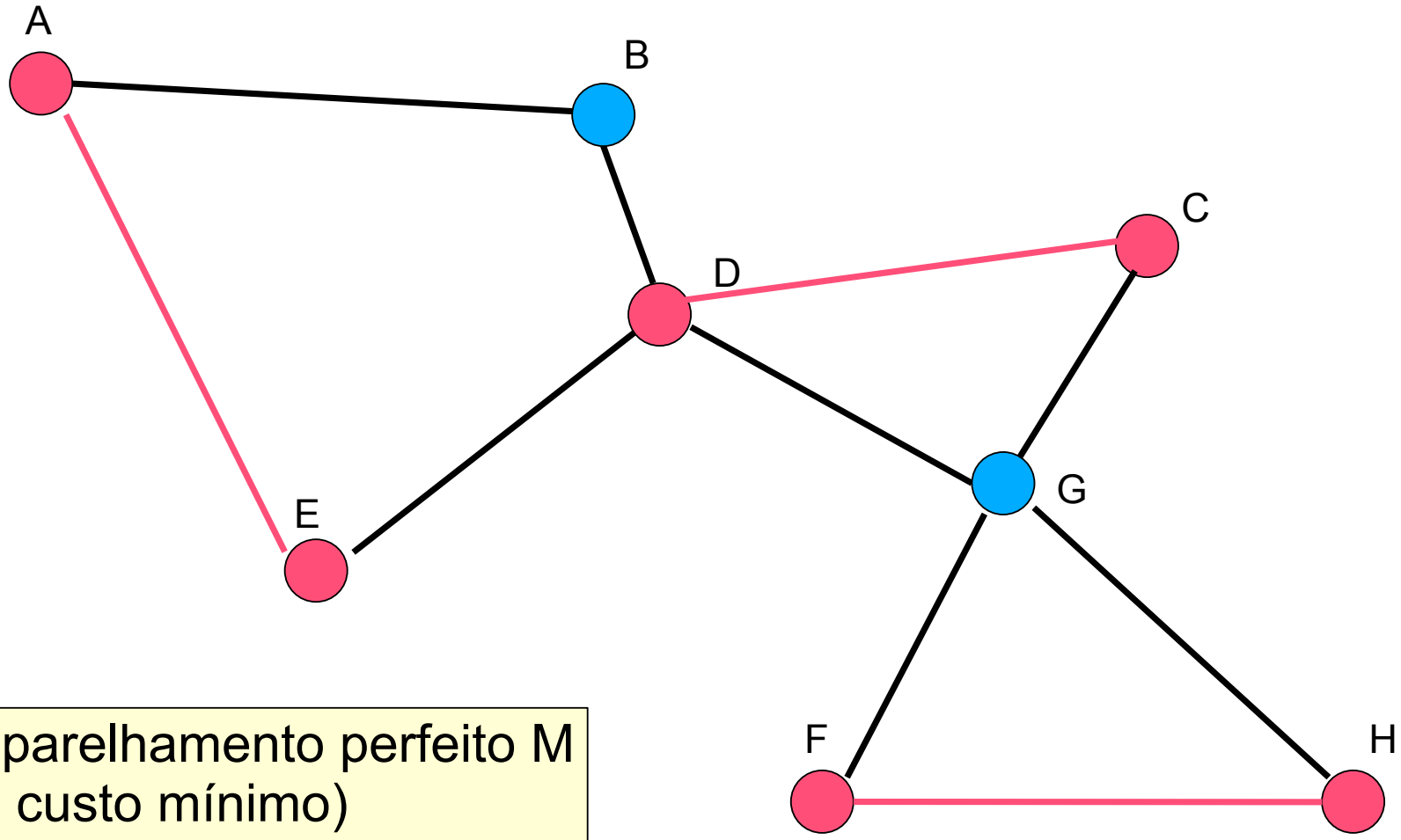
1) árvore geradora T de custo mínimo

Algoritmo de Christofides



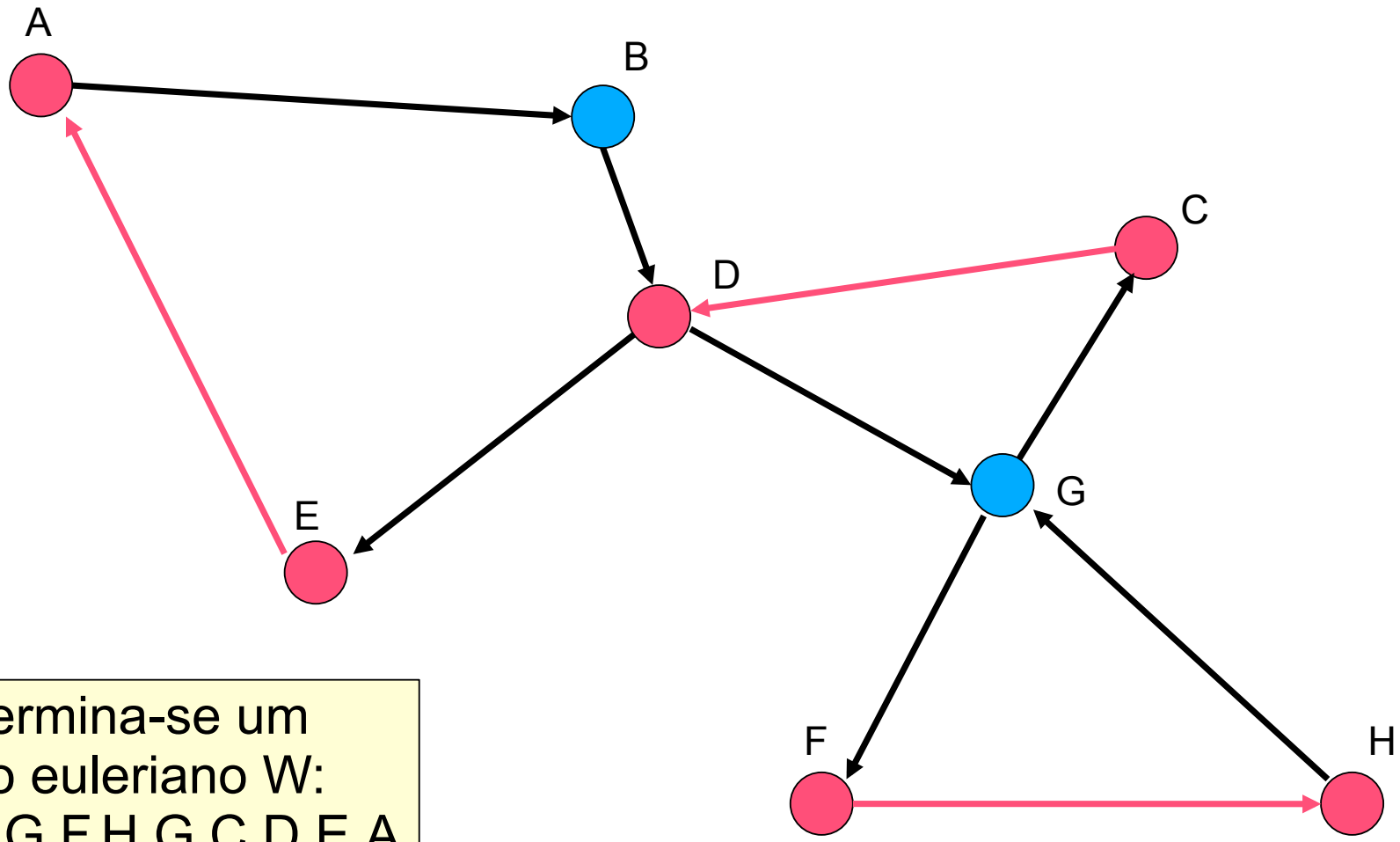
2) emparelhamento perfeito M
(de custo mínimo)
dos vértices de grau ímpar

Algoritmo de Christofides



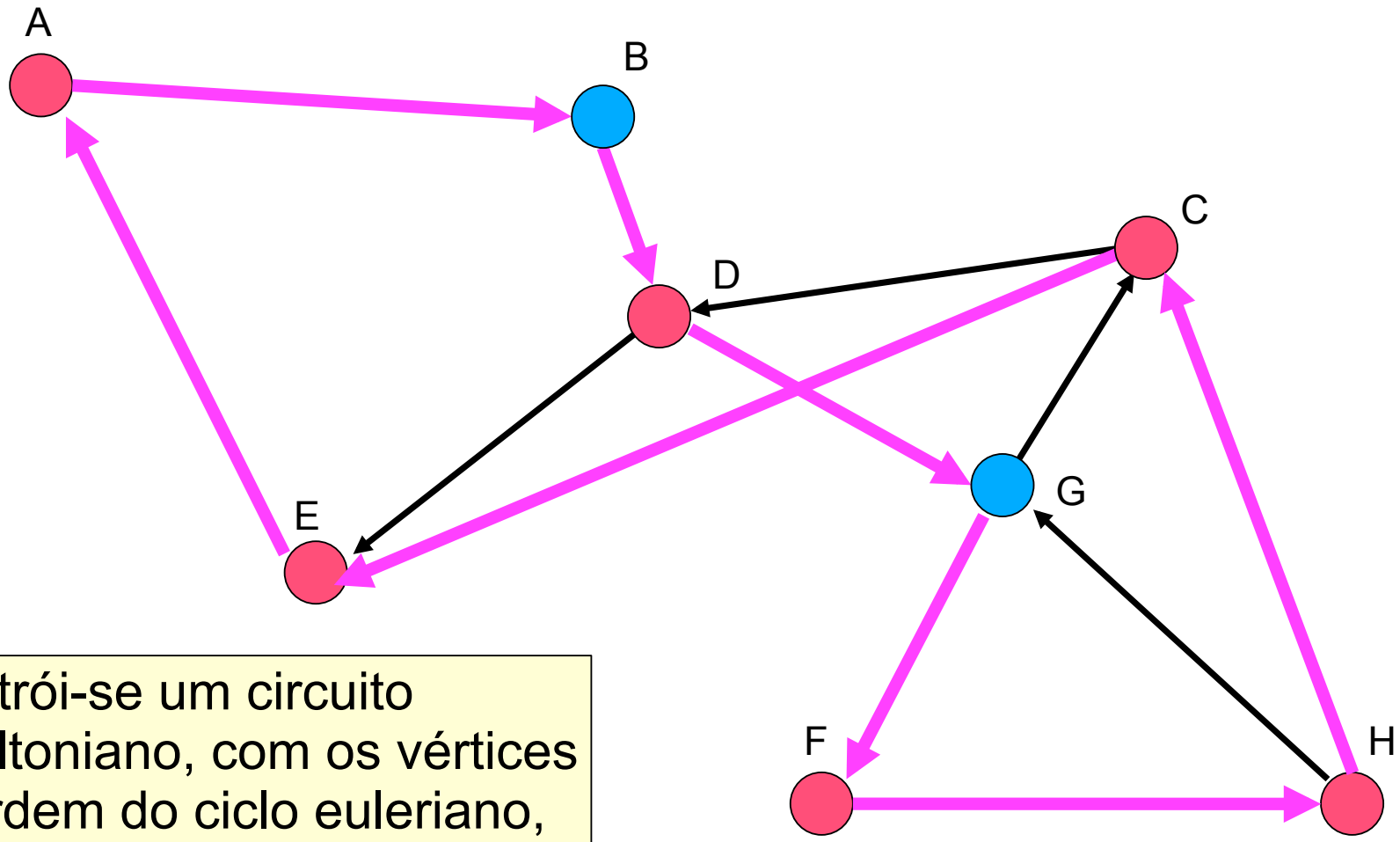
2) emparelhamento perfeito M
(de custo mínimo)
dos vértices de grau ímpar

Algoritmo de Christofides



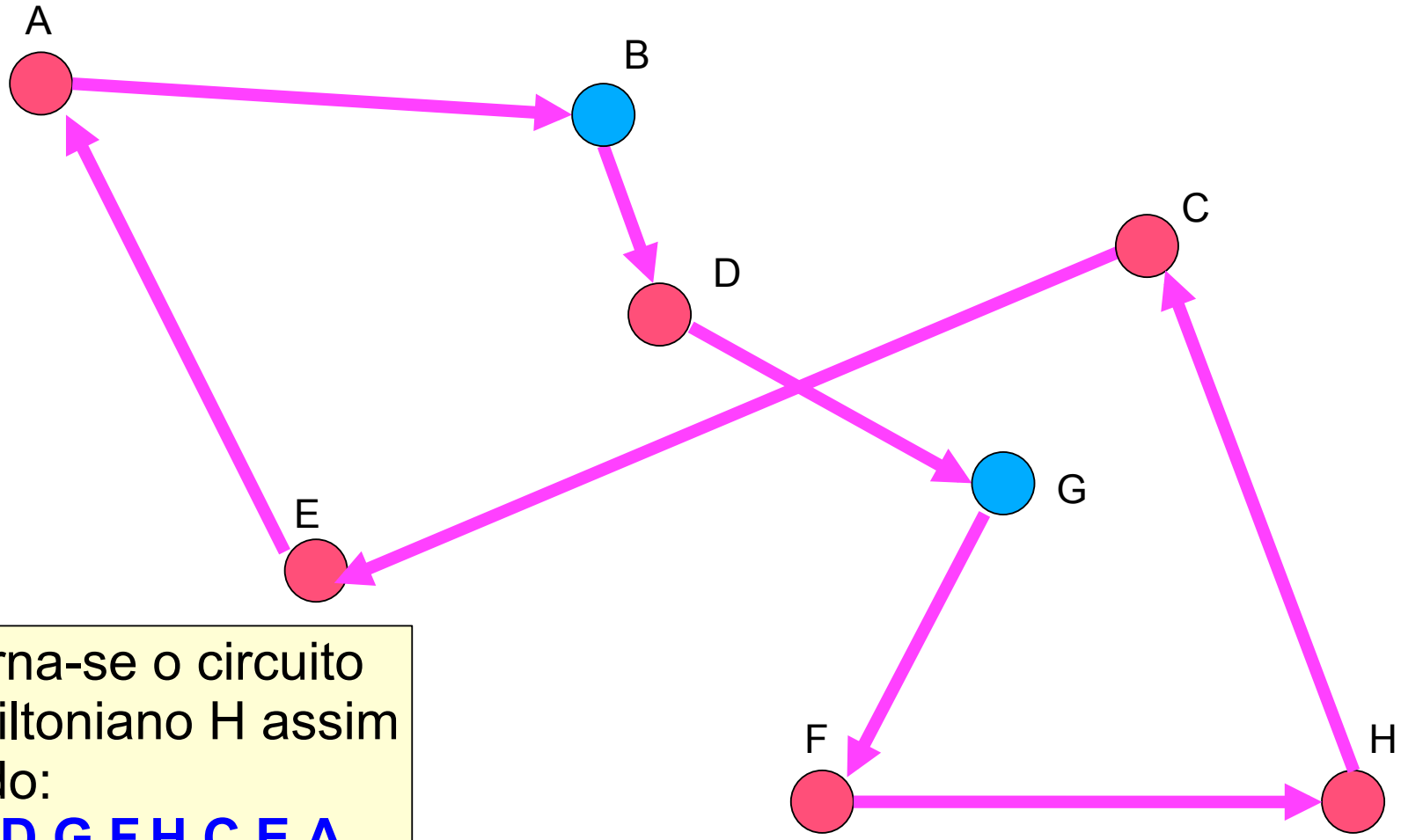
3) determina-se um ciclo euleriano W:
A,B,D,G,F,H,G,C,D,E,A

Algoritmo de Christofides



4) constrói-se um circuito hamiltoniano, com os vértices na ordem do ciclo euleriano, sem repetições:
A,B,D,G,F,H,G,**C,**D,**E,A**

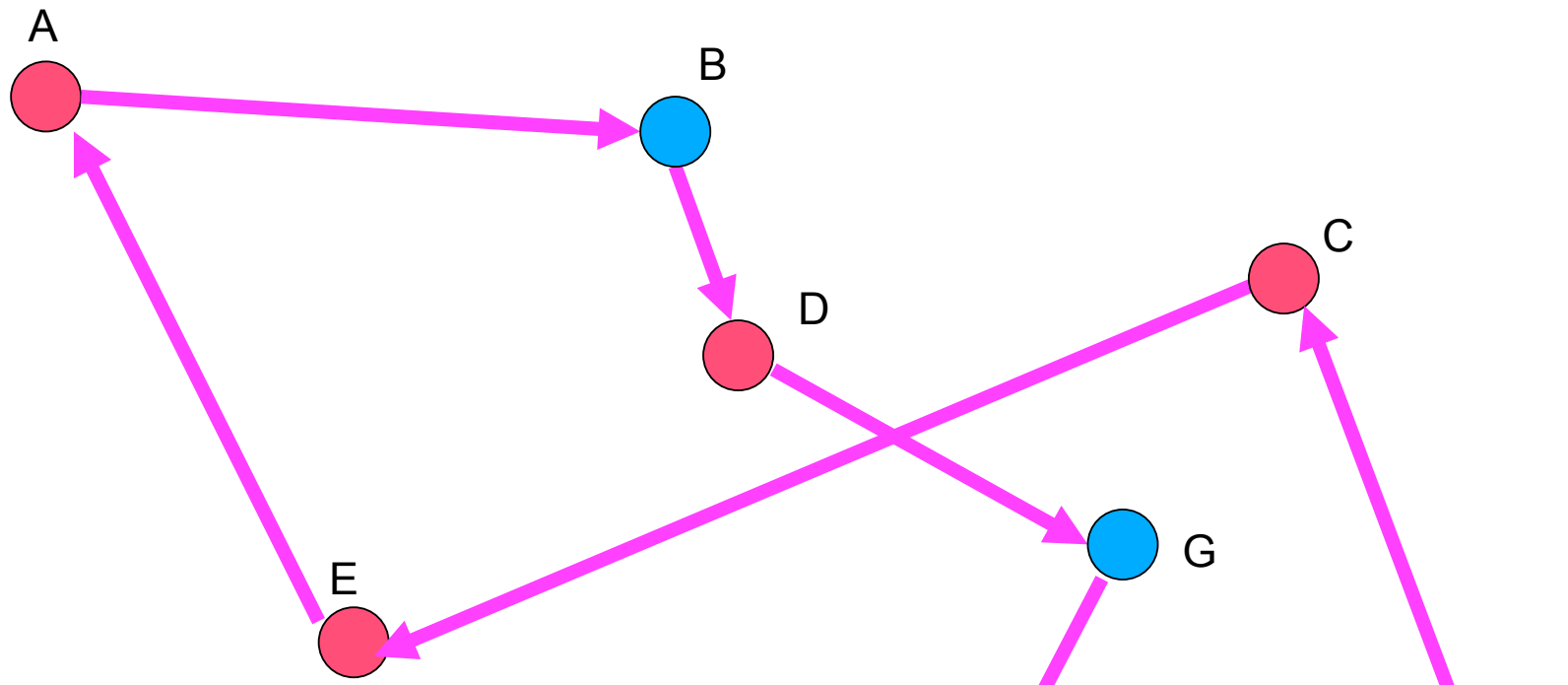
Algoritmo de Christofides



5) retorna-se o circuito hamiltoniano H assim obtido:
A,B,D,G,F,H,C,E,A

c(H) ≤ **1,5** . c(H*)

Algoritmo de Christofides



Tempo

$O(n^3)$

UFRJ