Lecture

8_9.2

Graph: Traversals & Basic Concepts

Paolo PRINETTO

Politecnico di Torino (Italy)

Univ. of Illinois at Chicago, IL (USA)

CINI Cybersecurity Nat. Lab. (Italy)

Paolo.Prinetto@polito.it

www.consorzio-cini.it

www.comitato-girotondo.org

cini Cybersecurity National Lab

License Information

Disclaimer

- We disclaim any warranties or representations as to the accuracy or completeness of this material.
- Materials are provided "as is" without warranty of any kind, either express or implied, including without limitation, warranties of merchantability, fitness for a particular purpose, and non-infringement.
- Under no circumstances shall we be liable for any loss, damage, liability or expense incurred or suffered which is claimed to have resulted from use of this material.

Goal

- This lecture aims at presenting graphs visiting techniques.

Prerequisites

- Lectures:
  - 8_9.1 Graphs: Introduction & definitions

Further readings

- **Students interested in a deeper look at the covered topics can refer, for instance, to the books listed at the end of the lecture.**

Outline

- Visite di un grafo
- Cammini e cicli
- Alberi e Foreste
- Cammini minimi
- Albero ricoprente
- Grafi Euleriani
- Grafi Hamiltoniani
- Isomorfismo
- Clique
- Planarità
- Colorabilità

Outline

- Visite di un grafo
- Cammini e cicli
- Alberi e Foreste
- Cammini minimi
- Albero ricoprente
- Grafi Euleriani
- Grafi Hamiltoniani
- Isomorfismo
- Clique
- Planarità
- Colorabilità

Visita di un grafo

Con il termine *visita di un grafo* si intende l'insieme delle operazioni che, partendo da un vertice iniziale, consentono di:

- esaminare successivamente gli altri vertici, utilizzando gli archi per passare da un vertice all'altro
- fornire in uscita l'insieme dei vertici e degli archi utilizzati.

Visita di un grafo

A seconda del criterio adottato per la scelta dei successivi vertici da visitare, si ottengono differenti algoritmi di visita.

I due criteri più usati sono noti come:

- visita in ampiezza (breadth first)
- visita in profondità (depth first).

Visita in ampiezza

Dato un vertice *s* di partenza, si assegna a ogni vertice un **livello** pari al numero minimo di archi che lo connettono a *s*.

La visita in ampiezza visita prima tutti i vertici aventi livello 1, poi tutti quelli con livello 2, e così via.

Come struttura dati ausiliaria impiega una coda FIFO per memorizzare l'insieme *Q* dei vertici utili alla ricerca.

Visita in ampiezza

- Inizialmente la coda contiene solo *s*.
- Al generico passo viene prelevato dalla coda l'elemento *j*.
- Successivamente vengono visitati tutti i vertici adiacenti a *j* non ancora visitati che, nel contempo, vengono inseriti nella coda.
- In questo modo i nodi visitati per primi sono quelli più vicini al nodo iniziale.

Visita in ampiezza: pseudo codice

```
breadth_first (vertex)
{
     visit(vertex);
     enqueue(vertex);
     while( la coda non è vuota)
     {    x = dequeue();
          for( ogni vertice w adiacente ad
                x e non ancora visitato)
          {
          visit(w);
               enqueue(w);
          }
     }
}
```


Rel. 07.04.2019

Visita in ampiezza

L'ordine di visita dei vertici dipende dall'ordine in cui questi sono stati memorizzati nelle liste di adiacenza.

Visita in ampiezza: esempio

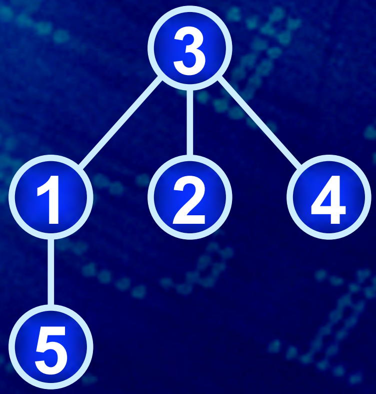
Visita in ampiezza: esempio

Assumendo che 3 sia il nodo di partenza e che i nodi a lui adiacenti siano stati memorizzati nell'ordine 1, 2, 4, si ha:
Visited Vertices:

Visita in ampiezza: esempio

Assumendo che 3 sia il nodo di partenza e che i nodi a lui adiacenti siano stati memorizzati nell'ordine 1, 2, 4, si ha:

Visited Vertices: 3 1 2 4 5

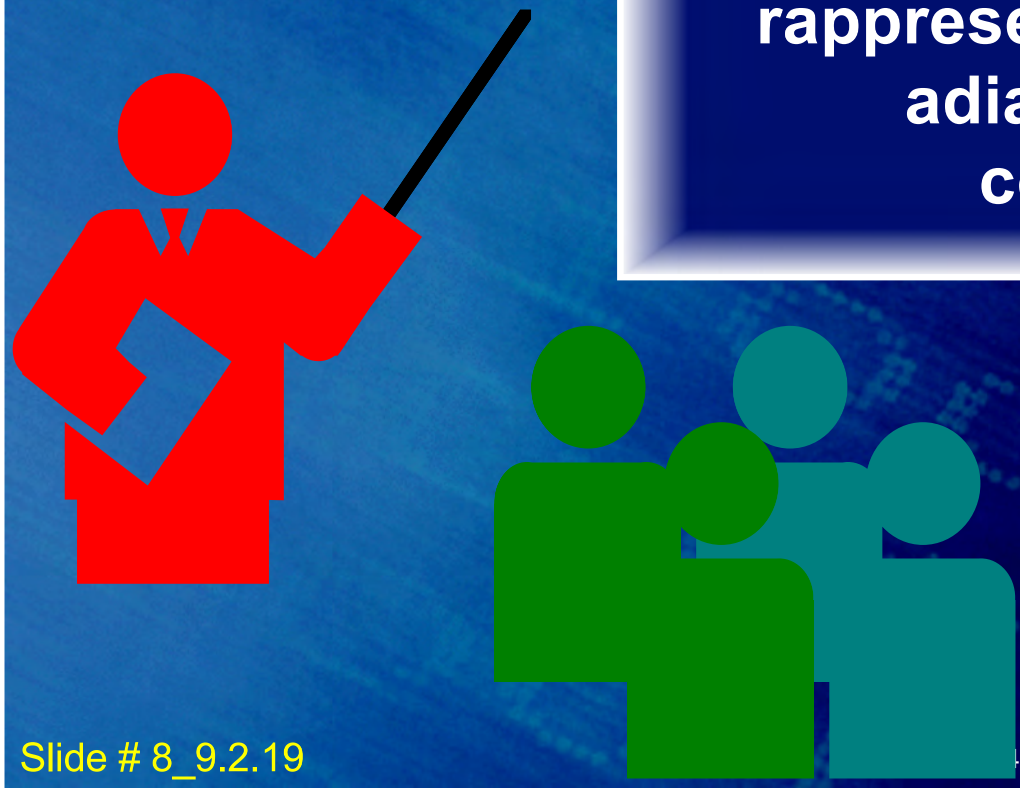


Teorema

L'algoritmo di visita in ampiezza consente di visitare tutti i vertici di un grafo se e solo se questi è connesso.

Teorema

L'algoritmo di visita in ampiezza di un grafo rappresentato come liste delle adiacenze ha complessità $O(max(|V|,|E|))$.
Nel caso in cui il grafo sia rappresentato come matrice delle adiacenze l'algoritmo ha complessità $O(|V|^2)$.



Visita in profondità

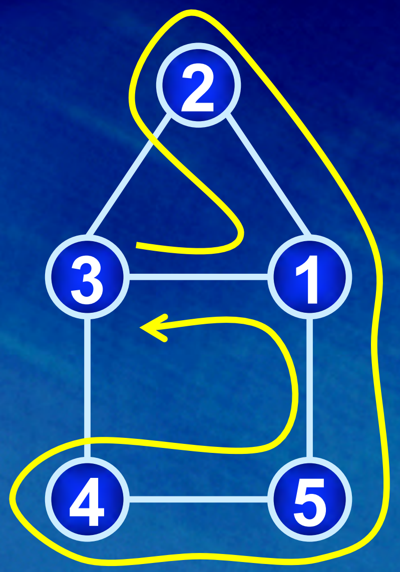
1. **Tra i vari vertici utili per il proseguimento della ricerca viene sempre scelto quello visitato più recentemente.**
2. **Al generico passo, appena visitato il nodo *k* si cerca un nodo non ancora esaminato adiacente a *k*; se un tale nodo esiste la visita prosegue, altrimenti si ritorna indietro al nodo *j* dal quale si era partiti per arrivare in *k* e si riprende a cercare nodi adiacenti a *j* non ancora visitati.**
3. **Se ne esiste almeno uno si prosegue, altrimenti si ritorna indietro di un altro passo.**

Visita in profondità: pseudo codice

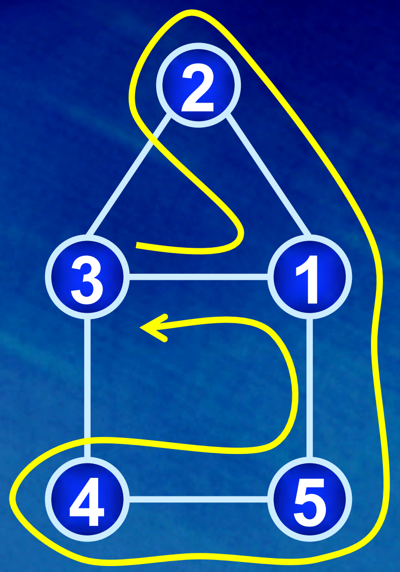
```
depth_first(vertex)
{     visit(vertex);
      for( ogni vertice w adiacente a
           vertex e non ancora visitato)
           {
           depth_first(w);
           }
}
```

Visita in profondità: esempio

Visita in profondità: esempio



Visita in profondità: esempio



Visited Vertices: 3 1 2 5 4

Visita in profondità

- È possibile implementare la visita in profondità in modo iterativo, con una procedura del tutto analoga a quella della visita in ampiezza, semplicemente sostituendo la coda con uno stack.
- Questa soluzione permette, tra l'altro, la realizzazione tramite linguaggi di programmazione che non ammettono la ricorsione.

Visita in profondità con stack

- **Inizialmente lo stack contiene solo s.**
- **Se, da un generico nodo j che si trova al top dello stack, è possibile raggiungere un nuovo nodo k non ancora visitato, allora k viene inserito in cima allo stack, altrimenti j viene eliminato dallo stack.**

Teorema

- L'algoritmo di visita in profondità di un grafo rappresentato come liste delle adiacenze ha complessità *O(|E|)* nella versione ricorsiva e *O(max(|V|,|E|))* in quella iterativa.
- Nel caso in cui il grafo sia rappresentato come matrice delle adiacenze l'algoritmo ha complessità *O(|V|^2)*.

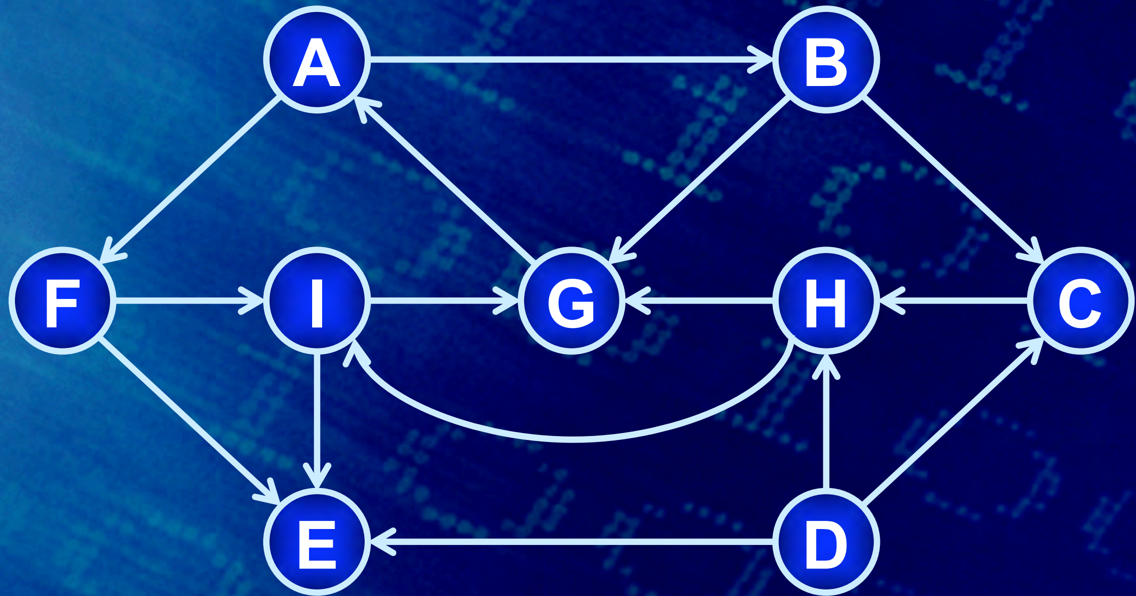
Visita in profondità: Note

- **Nel caso in cui il grafo sia un albero e il nodo s coincida con la radice dell'albero, una visita in profondità del grafo è equivalente a una visita pre-order dell'albero.**

- **Una visita in profondità è particolarmente adatta, ad esempio, nella ricerca di un cammino d'uscita in un labirinto.**

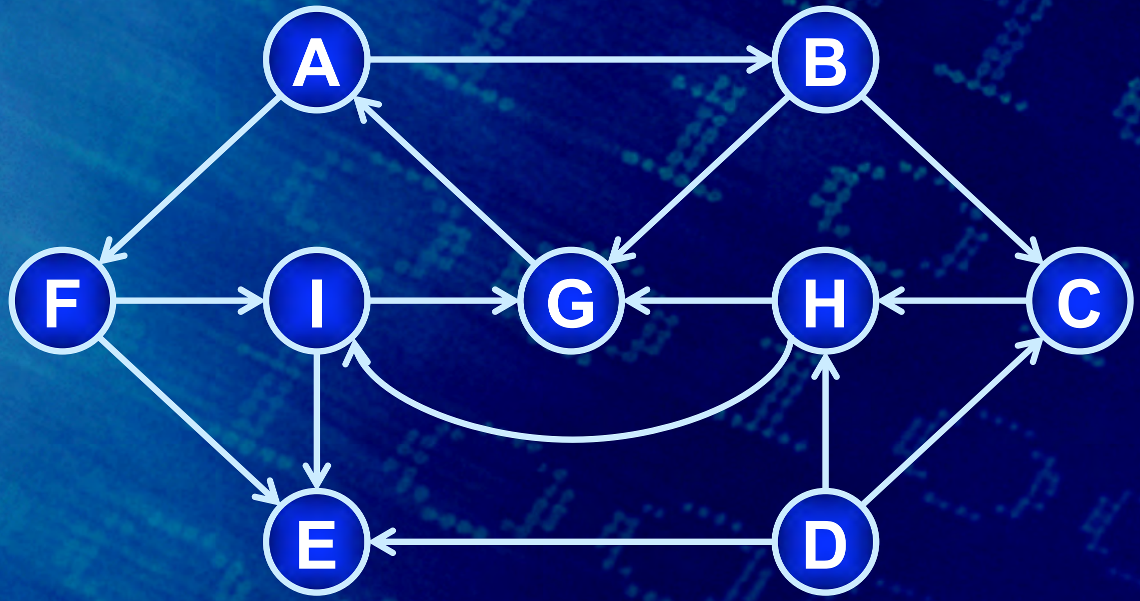
Esempio di visita

- Visita in profondità partendo da A:
  ???
- Visita in ampiezza partendo da A:
  ???



Esempio di visita

- Visita in profondità partendo da A:
  A - B - C - H - G - I - E - F
- Visita in ampiezza partendo da A:
  A - B - F - C - G - E - I - H
- Il nodo D non è raggiungibile da A.



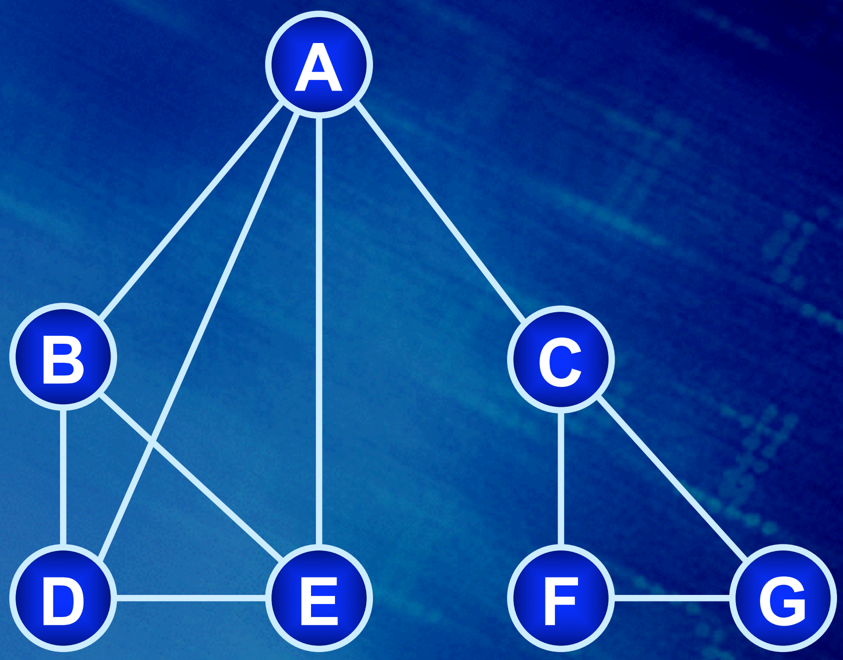
Albero di visita

Visitando un grafo con un algoritmo di visita in profondità si costruisce implicitamente un albero i cui archi sono un sottoinsieme degli archi del grafo.

Tali archi si dicono **tree edges**.

Se il grafo non è orientato, gli archi che non sono dei tree edges connettono un nodo con un suo predecessore nell'albero di visita, e per questo prendono il nome di **back edges**.

Albero di visita: esempio



Outline

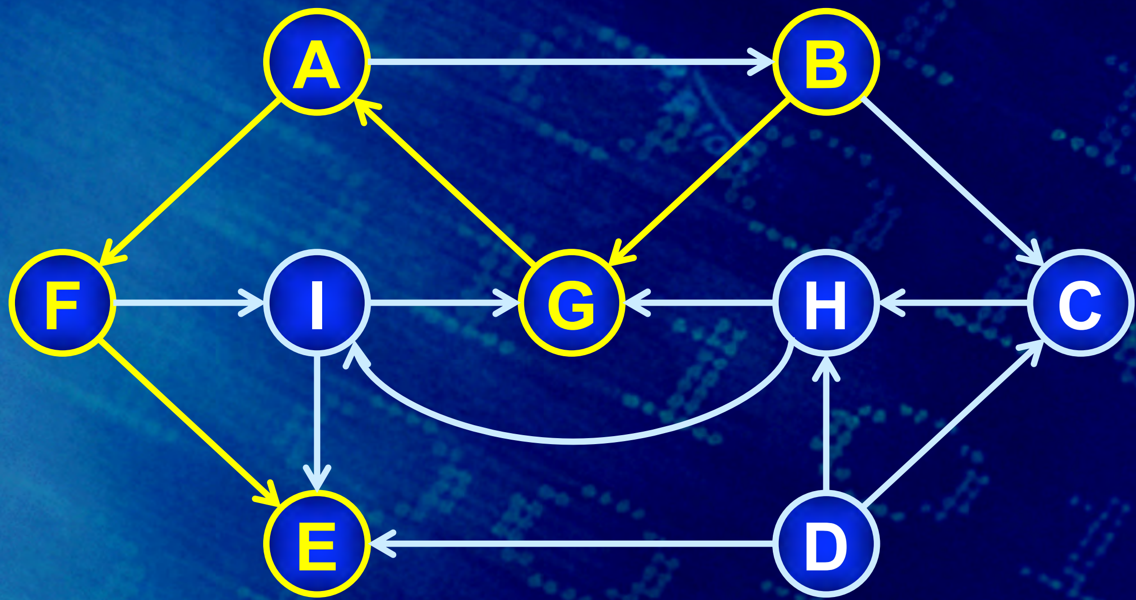
- Visite di un grafo
- Cammini e cicli
- Alberi e Foreste
- Cammini minimi
- Albero ricoprente
- Grafi Euleriani
- Grafi Hamiltoniani
- Isomorfismo
- Clique
- Planarità
- Colorabilità

Cammino

Un cammino (path) in un grafo è una sequenza di vertici (v_0, v_1, … , v_k) per cui esiste un arco che collega ogni coppia di vertici successivi nella sequenza.

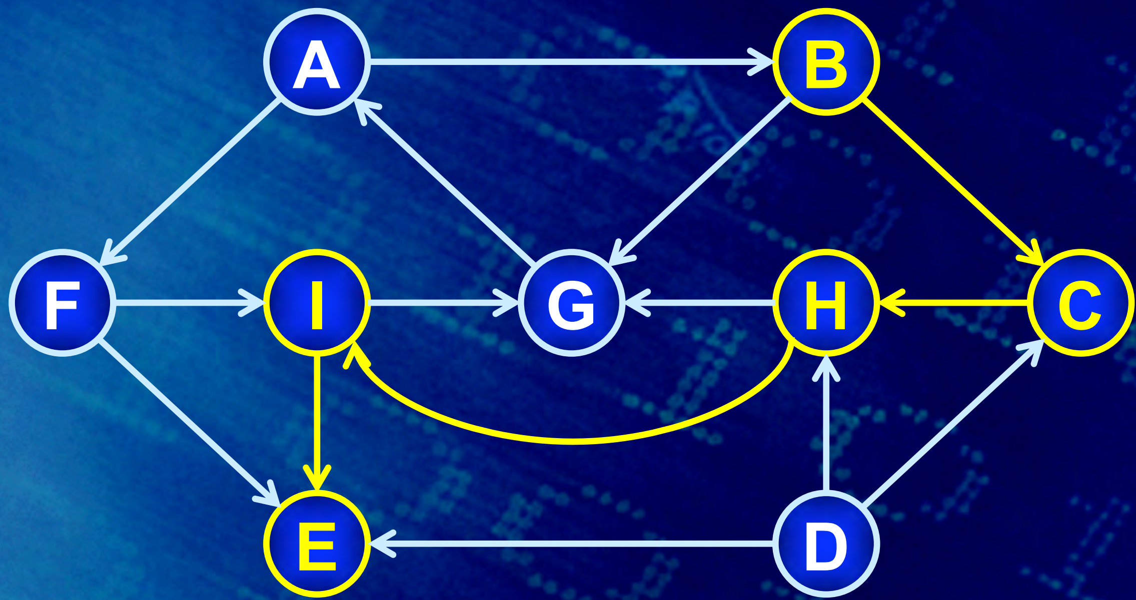
Cammino: Esempio

- **Un possibile cammino fra B ed E**



Cammino: Esempio

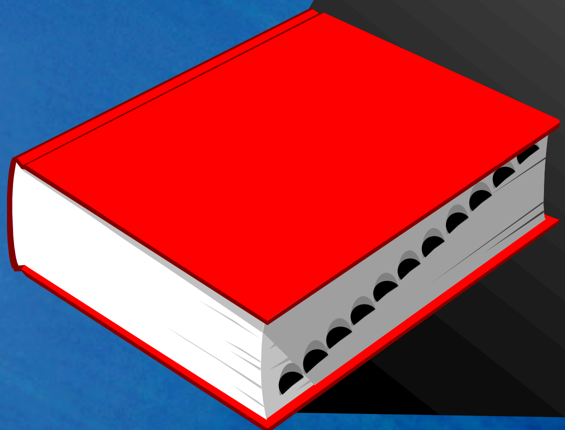
- **Un possibile cammino fra B ed E**



Lunghezza di un cammino

- Si definisce **lunghezza di un cammino** la somma dei pesi associati agli archi da cui esso è composto.
- Se un grafo non è pesato, si assume che ogni arco abbia peso pari a 1.
  - In tal caso la lunghezza del cammino è pari al numero degli archi che lo compongono.

Cammino semplice



Un cammino viene definito semplice se non usa uno stesso arco due o più volte, vale a dire se non contiene cicli.

Ciclo

Viene definito ciclo o circuito un cammino semplice per il quale i vertici di partenza e di arrivo coincidano.

Grafo aciclico

Se un grafo non contiene cicli viene detto aciclico.

Grafi diretti aciclici

Un grafo orientato e che non contenga cicli viene definito grafo diretto aciclico o **dag** (directed acyclic graph).

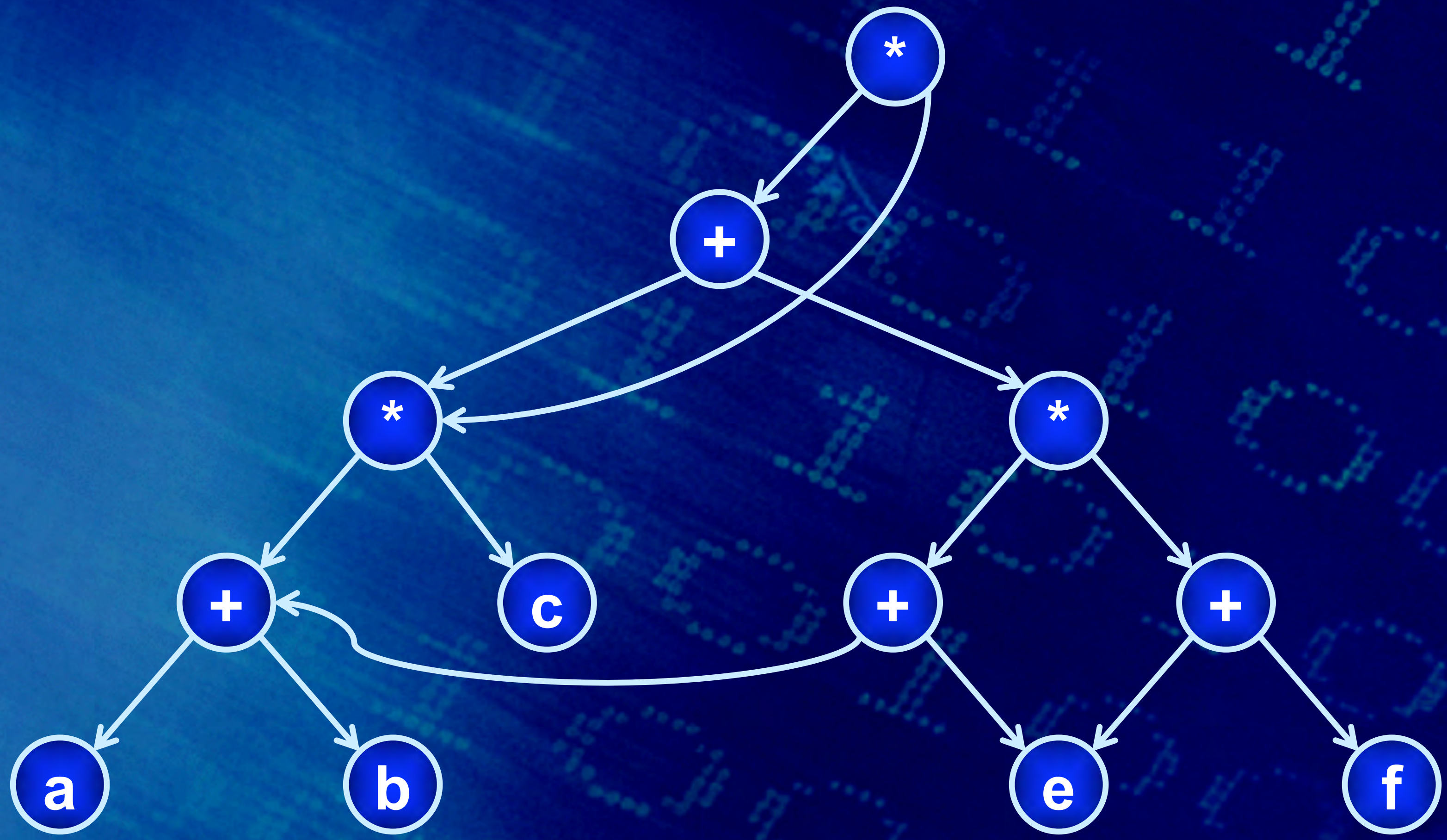
Grafi diretti aciclici - DAG

I DAG:

- sono più generali degli alberi e meno generali dei grafi orientati
- trovano applicazione, tra l'altro, nella rappresentazione della struttura sintattica di espressioni aritmetiche contenenti sottoespressioni comuni.

DAG: esempio di espressione aritmetica

((a+b) * c+((a+b)+e) * (e+f)) * ((a+b) * c)

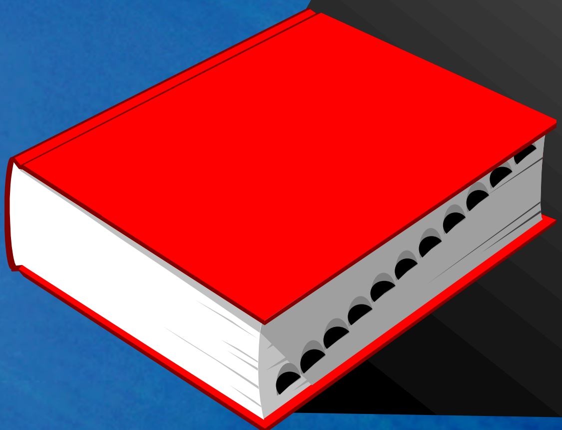


Teorema

Un grafo non orientato nel quale il grado di ogni nodo (numero di archi incidenti) è ≥ 2 possiede almeno un ciclo.

Grafo connesso

Un grafo non orientato si dice connesso (connected) se presi due qualsiasi suoi vertici v_i e v_j esiste un cammino tra v_i e v_j



Esempio di grafo non connesso

Outline

- Visite di un grafo
- Cammini e cicli
- Alberi e Foreste
- Cammini minimi
- Albero ricoprente
- Grafi Euleriani
- Grafi Hamiltoniani
- Isomorfismo
- Clique
- Planarità
- Colorabilità

Alberi

Il concetto di albero, precedentemente introdotto, viene qui riproposto come caso particolare del concetto di grafo.

Albero

Si definisce albero un grafo connesso e aciclico

Foresta

Un grafo (eventualmente orientato) le cui componenti connesse sono alberi, è detto foresta

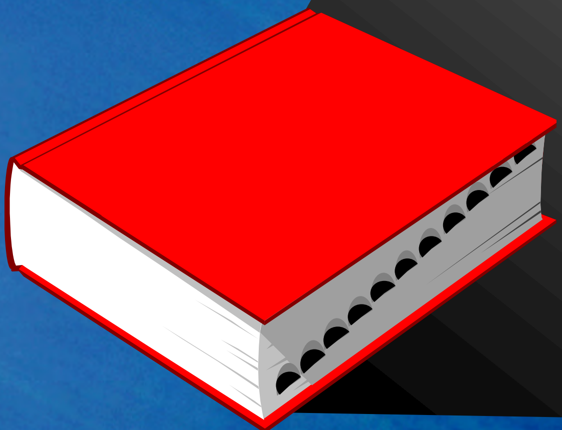
Esempio di Foresta

Outline

- – Visite di un grafo
- – Cammini e cicli
- – Alberi e Foreste
- – Cammini minimi
- – Albero ricoprente
- – Grafi Euleriani
- – Grafi Hamiltoniani
- – Isomorfismo
- – Clique
- – Planarità
- – Colorabilità

Cammino minimo

Un cammino tra due vertici si dice minimo se qualsiasi altro cammino tra gli stessi vertici ha lunghezza maggiore o uguale.



Cammini minimi

Due sono i problemi significativi relativi ai cammini minimi:

- determinare la lunghezza minima dei cammini da un vertice verso tutti gli altri
- determinare la lunghezza minima dei cammini tra tutte le coppie di vertici.

Albero dei cammini minimi

Fissato un vertice *s*, l'insieme dei cammini minimi tra *s* e qualsiasi altro vertice v da esso raggiungibile costituisce un albero, detto albero dei cammini minimi a partire da *s* (single source shortest path).

Outline

- – Visite di un grafo
- – Cammini e cicli
- – Alberi e Foreste
- – Cammini minimi
- – **Albero ricoprente**
- – Grafi Euleriani
- – Grafi Hamiltoniani
- – Isomorfismo
- – Clique
- – Planarità
- – Colorabilità

Albero ricoprente

Dato un grafo G = (V, E) non orientato, viene definito albero ricoprente (spanning tree) quel sottografo che è un albero e contiene tutti i vertici di G.

Teorema

Ogni grafo non orientato connesso contiene almeno un albero ricoprente.

Spanning Tree: esempio

Esempio di grafo e di uno dei suoi spanning tree

Minimum Spanning Tree

Dato un sottografo *G'* di un grafo *G* non orientato e pesato, si definisce peso di G' la somma dei pesi degli archi appartenenti a *G'*.

Si definisce minimum spanning tree di un grafo lo spanning tree avente peso minimo.

Minimum Spanning Tree: esempio

Un grafo può avere più minimum spanning tree:

Determinazione del Minimum Spanning Tree

La determinazione del minimum spanning tree è fondamentale in tutti quei casi in cui occorre minimizzare i costi di interconnessione tra un insieme di punti.

Outline

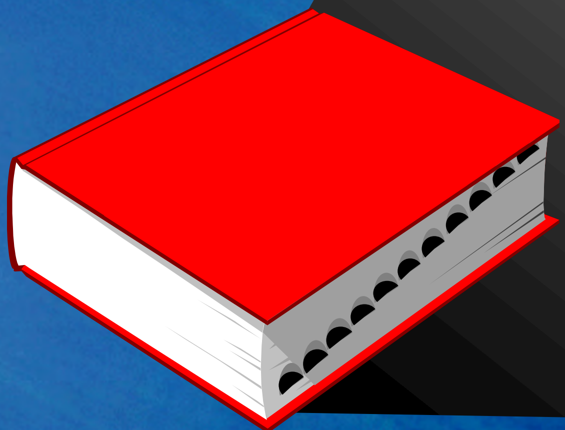
– Visite di un grafo
– Cammini e cicli
– Alberi e Foreste
– Cammini minimi
– Albero ricoprente
– **Grafi Euleriani**
– Grafi Hamiltoniani
– Isomorfismo
– Clique
– Planarità
– Colorabilità

Cammino di Eulero

Cammino che percorre tutti gli archi di un grafo esattamente una e una sola volta

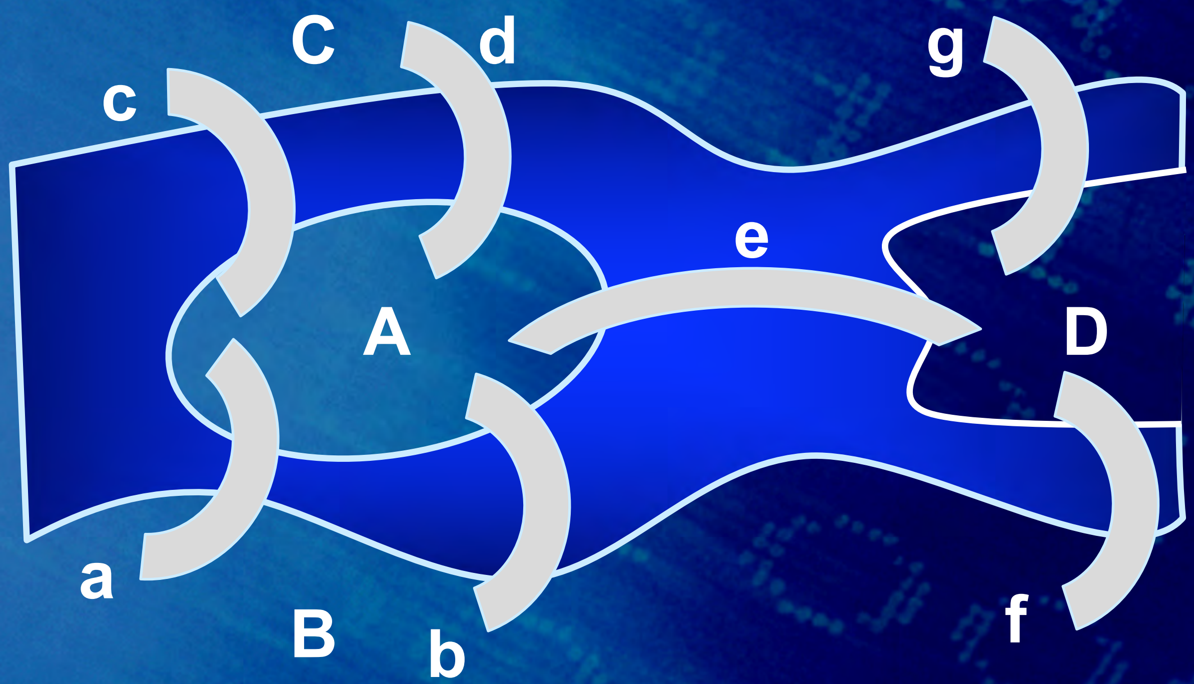
Ciclo di Eulero

Ciclo che percorre tutti gli archi di un grafo esattamente una e una sola volta



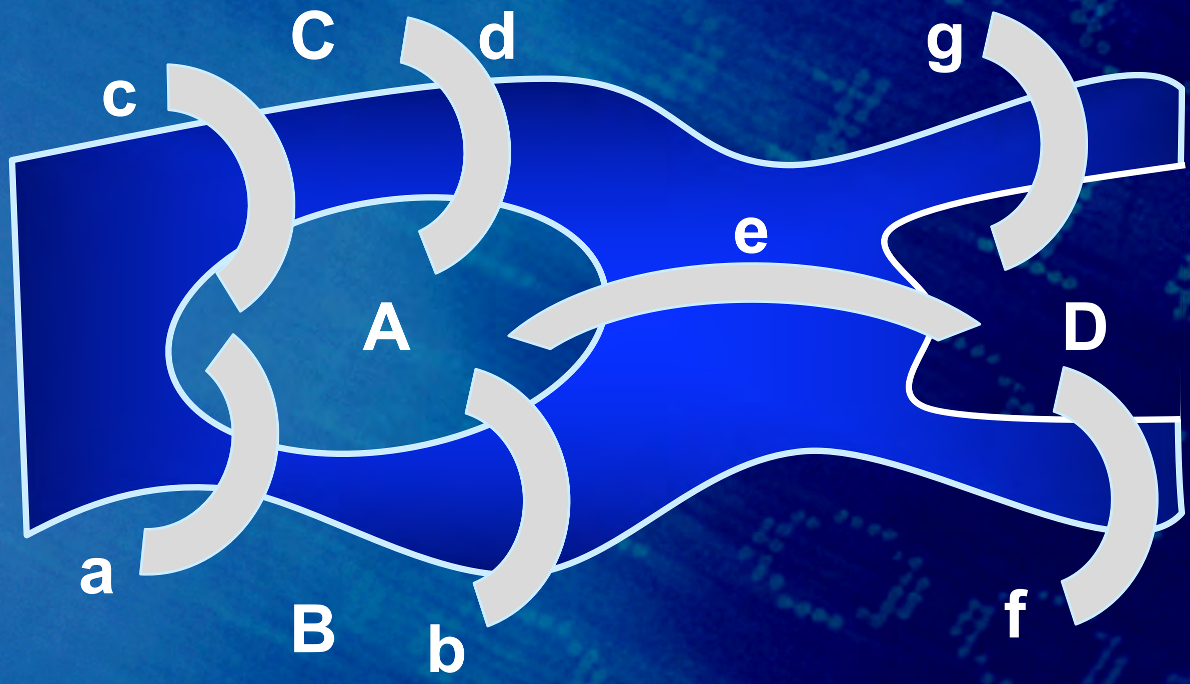
Cammino di Eulero: esempi

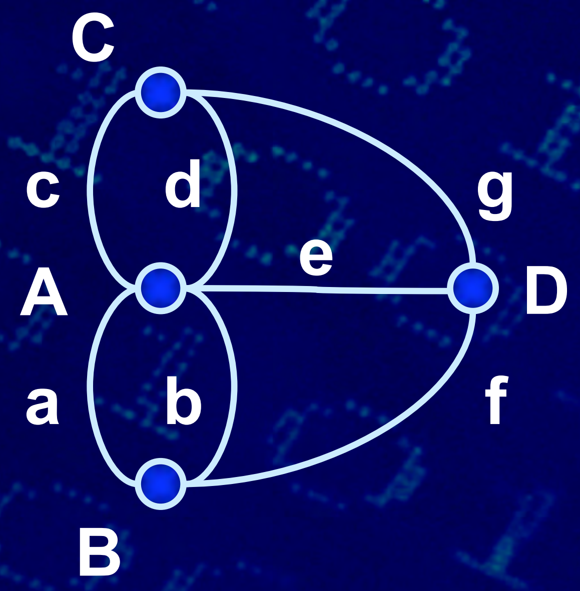
- Ponti di Königsberg
  - È possibile attraversare i sette ponti di Königsberg (oggi Kalingrad, Russia), sul fiume Pregal (oggi Pregolya) una e una sola volta?



Cammino di Eulero: esempi

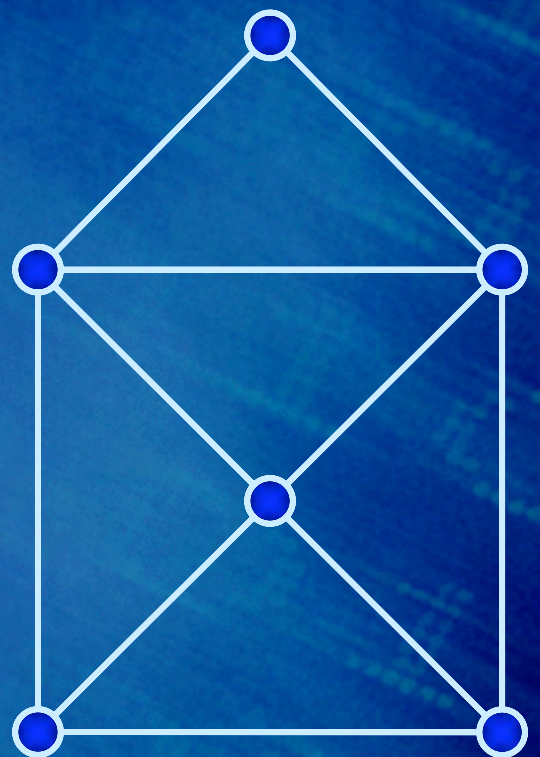
- Ponti di Königsberg
  - È possibile attraversare i sette ponti di Königsberg (oggi Kalingrad, Russia), sul fiume Pregal (oggi Pregolya) una e una sola volta?

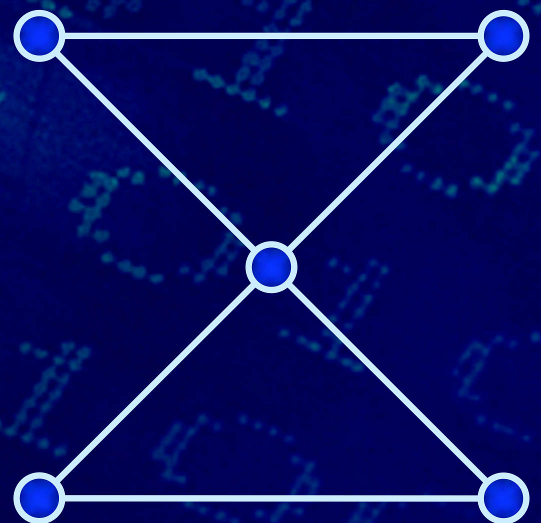




Cammino di Eulero: esempi

- **È possible disegnare una figura senza staccare la matita dal foglio e senza ripassare su una stessa linea?**





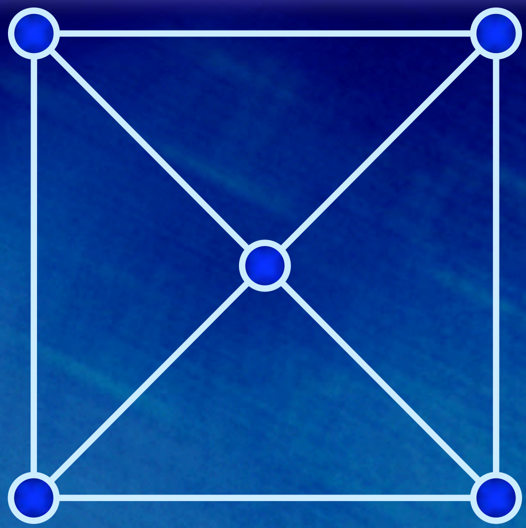
Grafo euleriano

Grafo connesso che contiene un ciclo di Eulero

Grafo semi-euleriano

Grafo connesso che contiene un cammino di Eulero

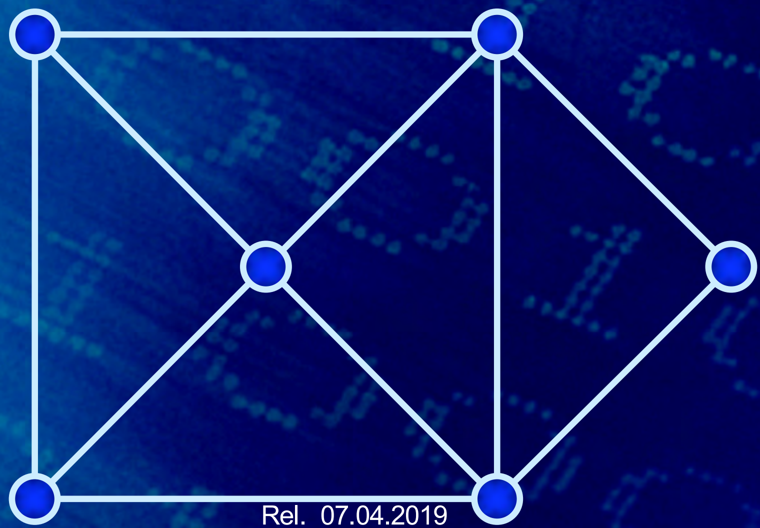
Grafi euleriani: Esempi



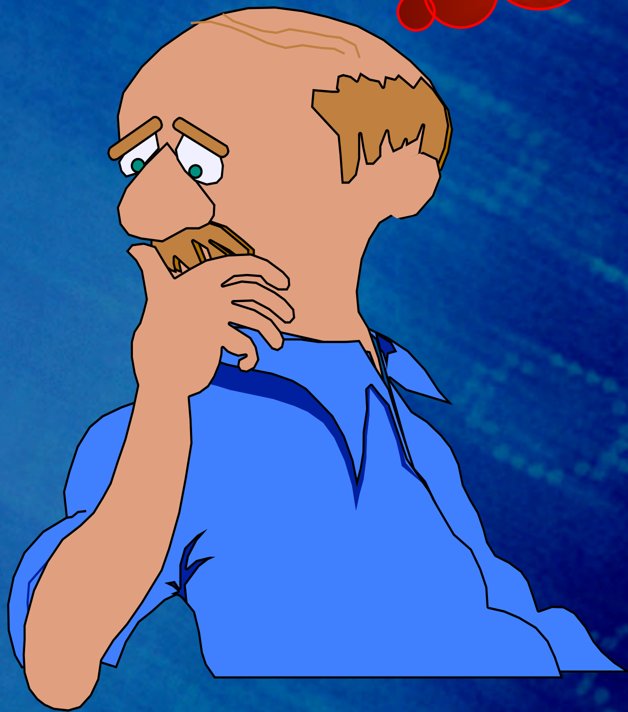
Grafo non-euleriano

Grafo euleriano

Grafo semi-euleriano

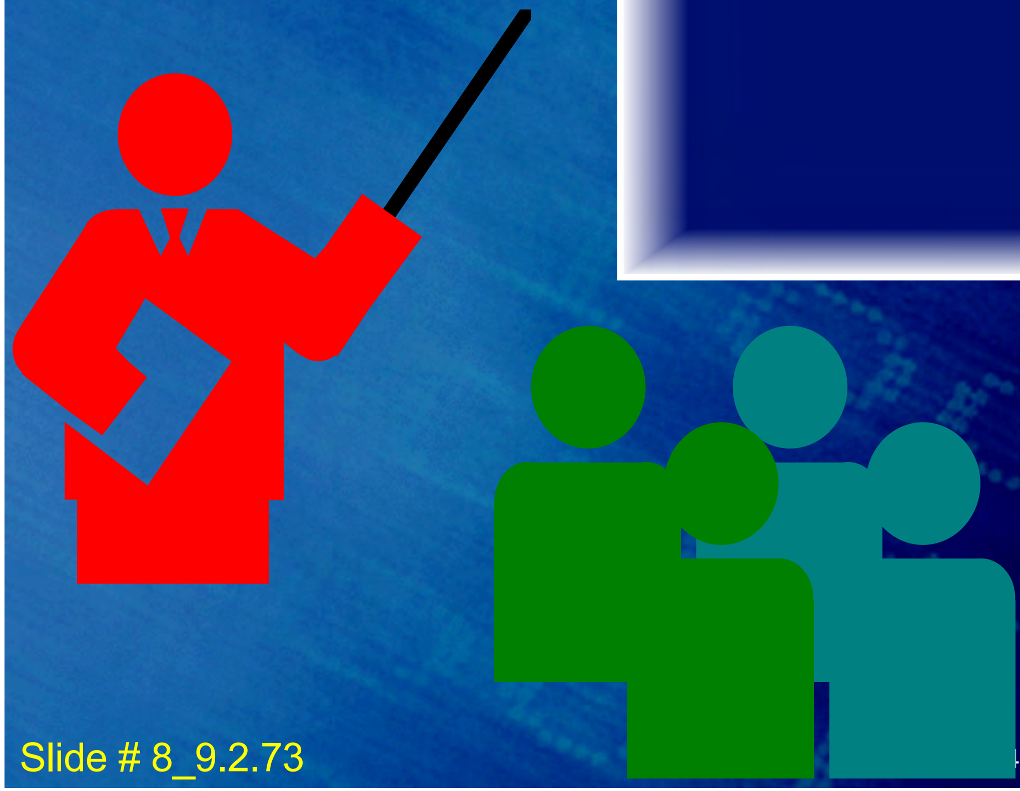


É possibile dare condizioni necessarie e sufficienti affinché un grafo sia euleriano?



Teorema

Un grafo non orientato è euleriano se e solo se è connesso e i suoi vertici sono tutti di ordine pari



Corollario

Un grafo non orientato è semi-euleriano se e solo se è connesso e possiede o nessuno o esattamente due vertici di ordine dispari

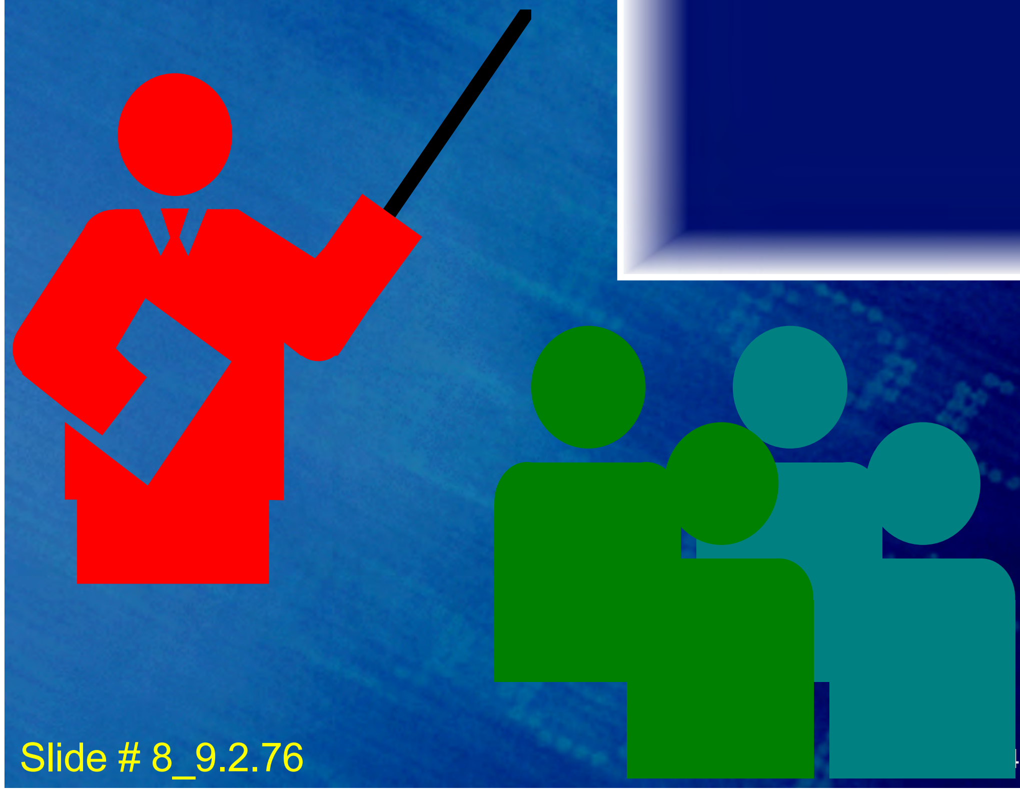
Corollario

Un grafo orientato possiede un cammino di Eulero se e solo se è connesso e, per ogni vertice (con la possibile eccezione di due), il grado in ingresso è uguale al grado in uscita. Per gli eventuali due vertici anomali, in uno il grado in ingresso deve essere di 1 superiore al grado in uscita, mentre nell'altro il grado in ingresso deve essere di 1 inferiore al grado in uscita

Corollario

Un grafo orientato possiede un ciclo di Eulero se e solo se è connesso e, per ogni vertice, il grado in ingresso è uguale al grado in uscita:

$$\rho^{\rightarrow}(i) = \rho^{\leftarrow}(i) \quad \forall i$$



Outline

- Visite di un grafo
- Cammini e cicli
- Alberi e Foreste
- Cammini minimi
- Albero ricoprente
- Grafi Euleriani
- **Grafi Hamiltoniani**
- Isomorfismo
- Clique
- Planarità
- Colorabilità

Cammino di Hamilton

Cammino che percorre tutti i vertici di un grafo esattamente una e una sola volta

Ciclo di Hamilton

Ciclo che percorre tutti i vertici di un grafo esattamente una e una sola volta

Grafi hamiltoniani: Commesso Viaggiatore

Il nome deriva dal gioco proposto nel 1857 da Sir William Hamilton, basato sulla costruzione di un ciclo che tocchi tutti i vertici di un dodecaedro.

Tipico esempio di un problema che può essere ricondotto alla ricerca di un ciclo di Hamilton su un grafo pesato è quello del **commesso viaggiatore** (Travelling Salesman Problem - TSP).

Grafi hamiltoniani: Commesso Viaggiatore

Il nome deriva dal gioco proposto nel 1857 da Sir William Hamilton, basato sulla costruzione di un ciclo che tocchi tutti i vertici di un dodecaedro.

Tipico esempio di un problema che può essere ricondotto alla ricerca di un ciclo di Hamilton su un grafo pesato è quello del commesso viaggiatore (Travelling Salesman Problem - TSP).

TSP

- The Traveling Salesman Problem, or TSP, is defined as follows:
  - Given a weigthed graph G=(V,E), find the hamiltonian cycle with the minimum weigth

Determinazione di un cammino di Hamilton

- **A differenza di quanto avviene per i grafi euleriani, l'individuazione di una condizione necessaria e sufficiente per stabilire se un grafo è hamiltoniano è uno dei maggiori problemi insoluti della teoria dei grafi.**
- **È stato comunque dimostrato che l'individuazione dell'esistenza di un cammino di Hamilton è un problema $\mathcal{NP}$-hard.**
- **Una condizione sufficiente è espressa dal seguente teorema.**

Rel. 07.04.2019

Teorema di Dirac

Sia *G* un grafo semplice con |*V*| nodi. Se (|*V*| ≥ *3*) e

$$\rho(v) \geq |V| / 2$$

per ogni nodo *v*, allora *G* è hamiltoniano

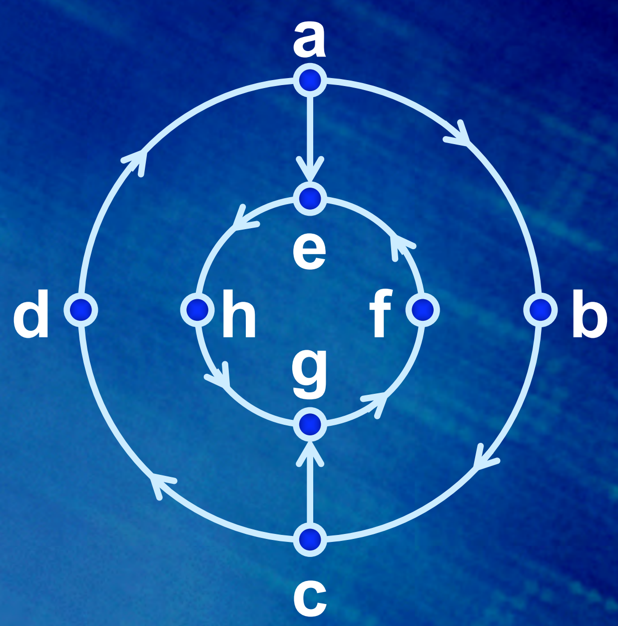
Outline

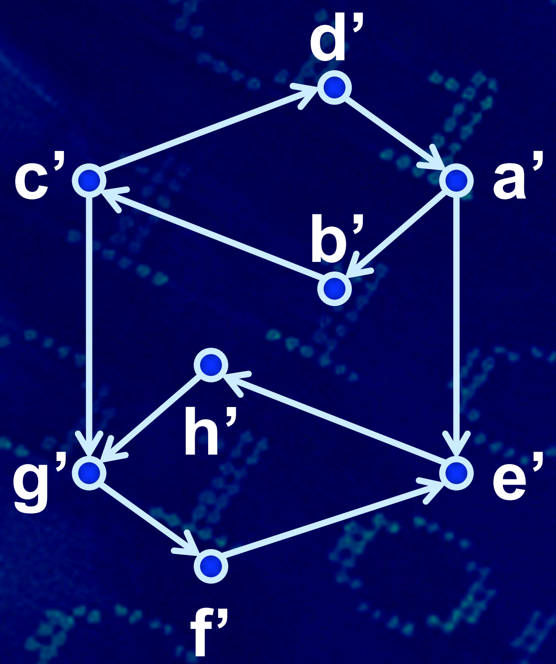
– Visite di un grafo
– Cammini e cicli
– Alberi e Foreste
– Cammini minimi
– Albero ricoprente
– Grafi Euleriani
– Grafi Hamiltoniani
– **Isomorfismo**
– Clique
– Planarità
– Colorabilità

Isomorfismo

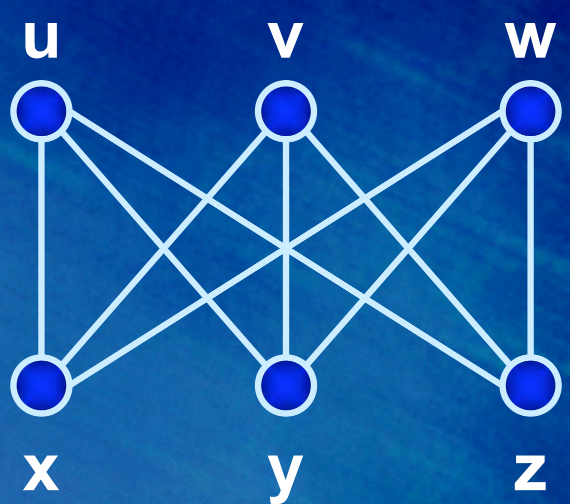
Due grafi vengono detti isomorfi se esiste una corrispondenza biunivoca tra i loro nodi, con la proprietà che due nodi sono congiunti da un arco in un grafo se e solo se i corrispondenti nodi sono uniti da un arco nell'altro.

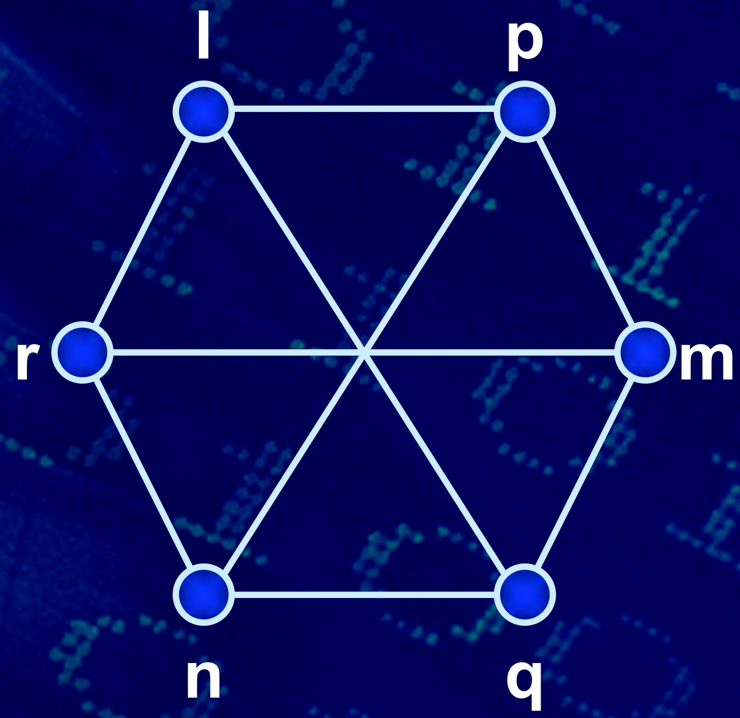
Grafi orientati isomorfi: esempio





Grafi non orientati isomorfi: esempio





Outline

- Visite di un grafo
- Cammini e cicli
- Alberi e Foreste
- Cammini minimi
- Albero ricoprente
- Grafi Euleriani
- Grafi Hamiltoniani
- Isomorfismo
- **Clique**
- Planarità
- Colorabilità

Clique

Viene denominata clique (cricca) di G un sottografo completo massimale di G.

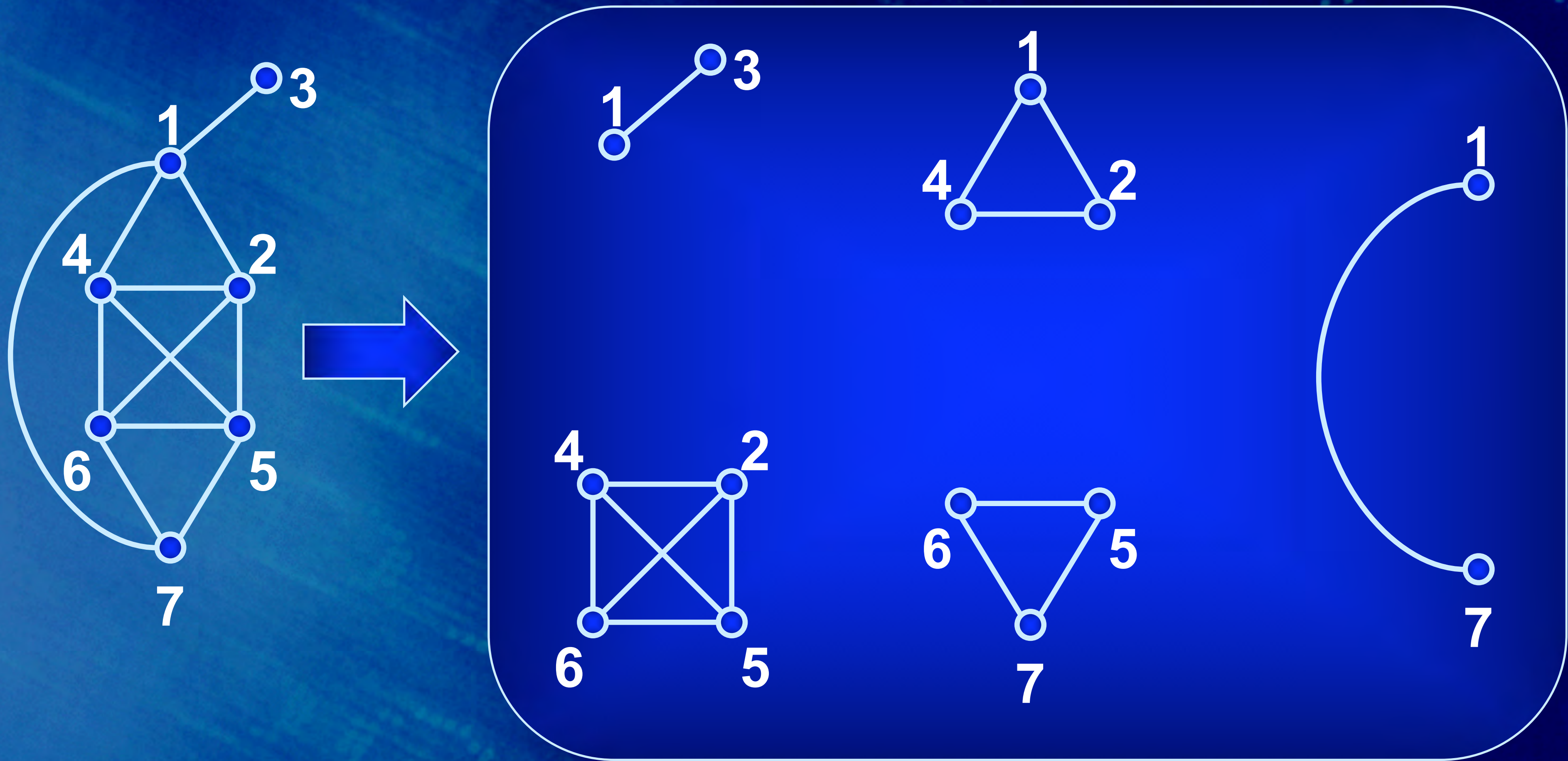
Clique

Dato un grafo *G*, la clique è un sottoinsieme dei vertici di *G* tale che:

- esista un arco tra tutte le possibili coppie di vertici
- il sottoinsieme non sia contenuto in alcun altro sottoinsieme di dimensioni maggiori e che soddisfi la stessa proprietà.

Clique di un grafo: esempio

Esempio di un grafo e di tutte le sue clique



Determinazione delle clique

- Il problema della determinazione della (o delle) clique di dimensione massima all'interno di un grafo è un problema di tipo $\mathcal{NP}$-hard.

Outline

- Visite di un grafo
- Cammini e cicli
- Alberi e Foreste
- Cammini minimi
- Albero ricoprente
- Grafi Euleriani
- Grafi Hamiltoniani
- Isomorfismo
- Clique
- **Planarità**
- Colorabilità

Planarità

- Un grafo **piano** è un grafo che può essere disegnato sul piano in modo tale che due archi qualsiasi (o meglio, le curve che li rappresentano) non si incrocino mai eccetto che a un nodo al quale sono entrambi adiacenti.
- Un grafo **planare** è un grafo isomorfo a un grafo piano.

Planarità

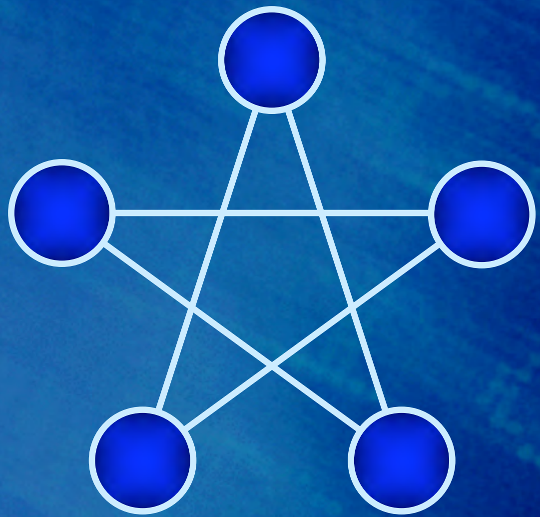
- Qualsiasi rappresentazione di mappe o informazioni topografiche è planare
  - Algoritmi relativi ai grafi sono spesso specializzati per grafi planari (ad es., Travelling Salesman Problem)
- I circuiti sono in genere rappresentati da grafi planari
  - La determinazione della planarità di un grafo riveste particolare importanza nelle fasi di placement e routing sia dei circuiti integrati sia delle piastre.

Planarità

- Nessun algoritmo efficiente per determinare la planarità di un grafo era noto fino al 1974, quando R.E.Tarjan sviluppò un algoritmo lineare, ricorrendo a una complicata visita in profondità.

Planarità: Errore comune

Il fatto che un grafo sia disegnato con degli archi che si incrocino non implica che esso non sia planare



Planarità: Errore comune

Il fatto che un grafo sia disegnato con degli archi che si incrocino non implica che esso non sia planare

Planarità: Errore comune

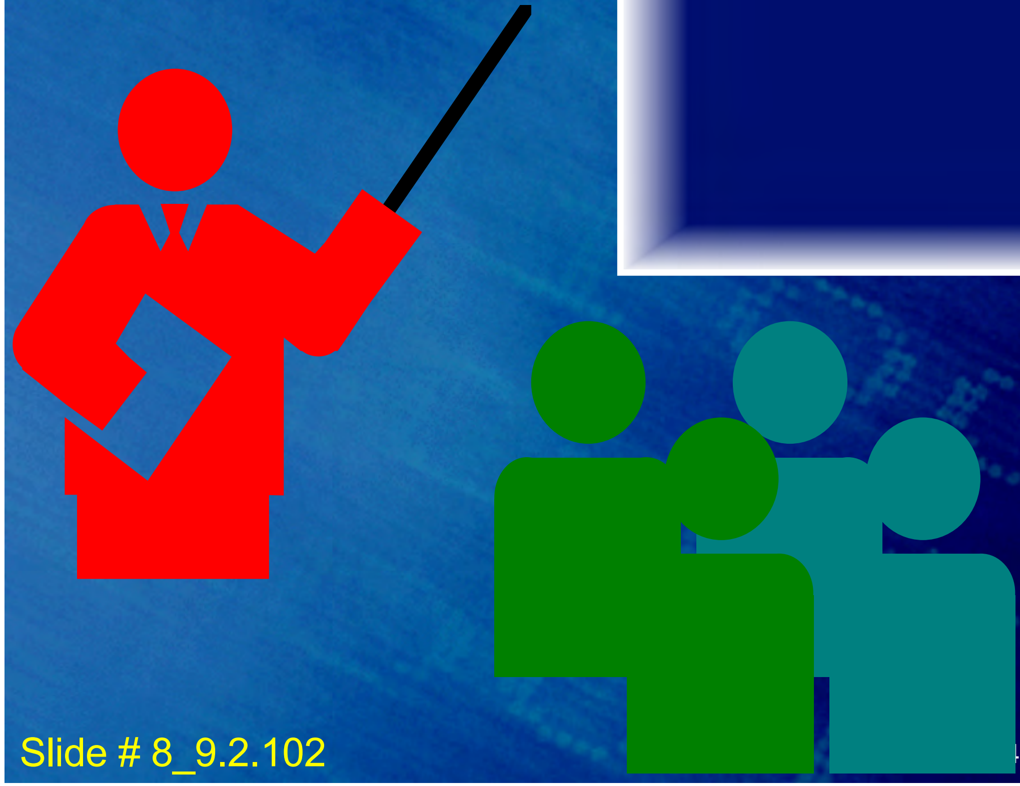
Il fatto che un grafo sia disegnato con degli archi che si incrocino non implica che esso non sia planare

Teorema

Un grafo è planare se e solo se può essere tracciato sulla superficie di una sfera.

Teorema

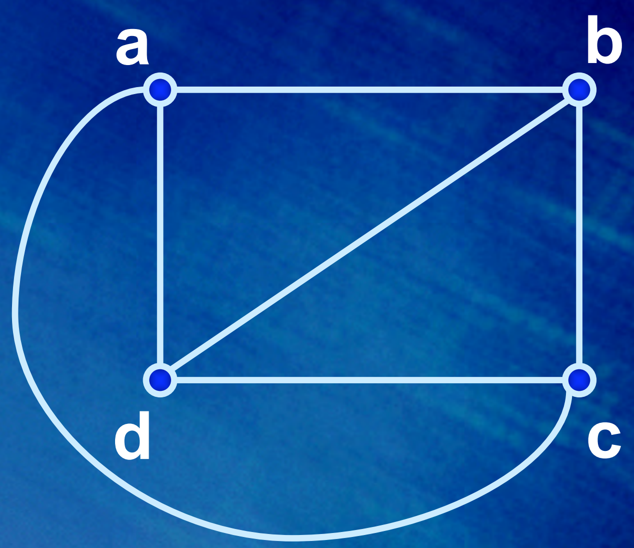
Un grafo planare semplice non contiene nodi il cui grado sia >5

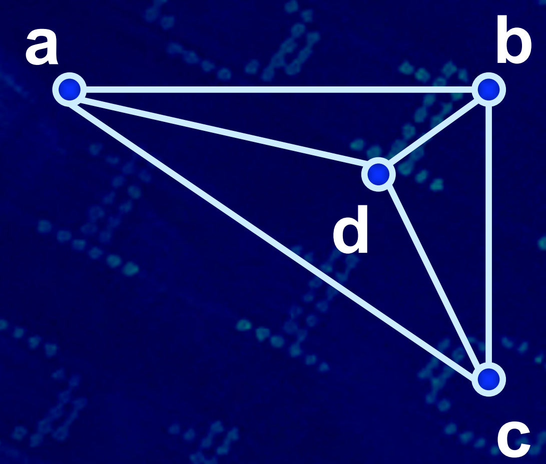


Teorema

Un grafo planare semplice può essere tracciato su un piano in modo tale che tutti i suoi archi siano rappresentati da segmenti rettilinei (Fary, 1948).

Grafo planare: esempio

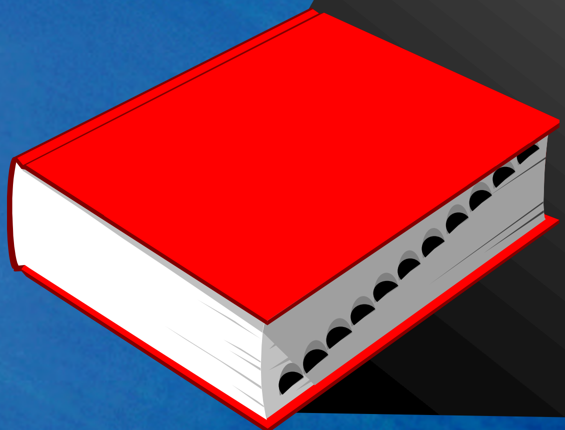




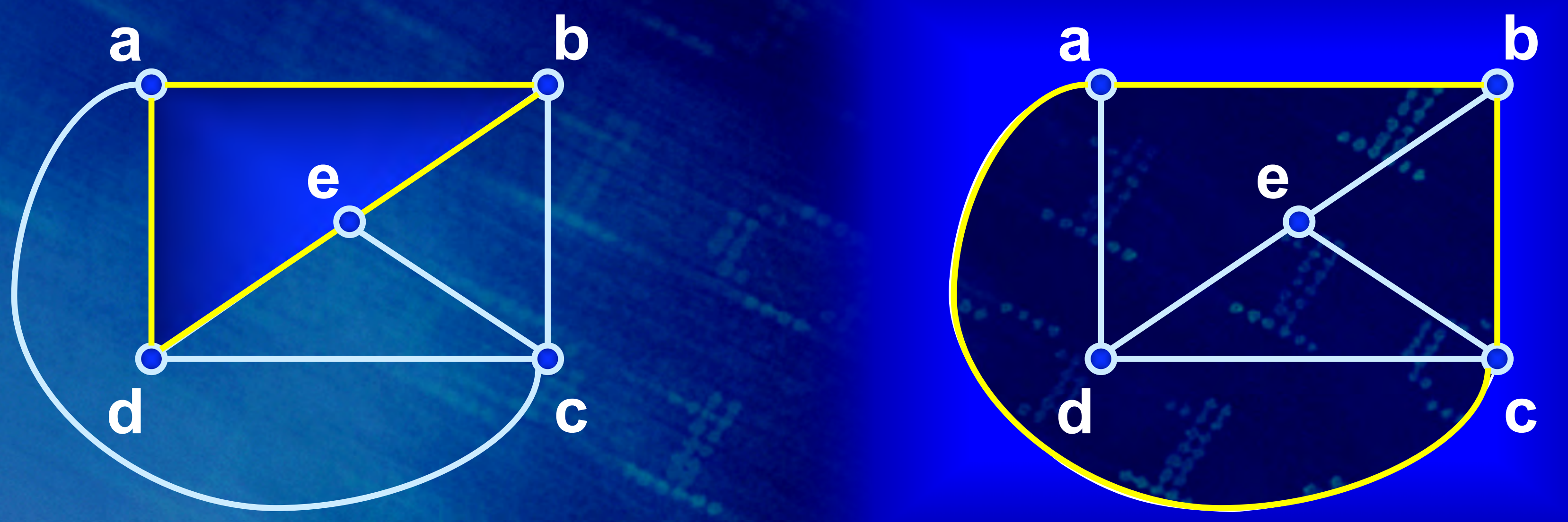
Grafo planare: esempio

Regione

Una regione di un grafo piano connesso è un'area del piano limitata da archi del grafo e che non contiene né archi né vertici.

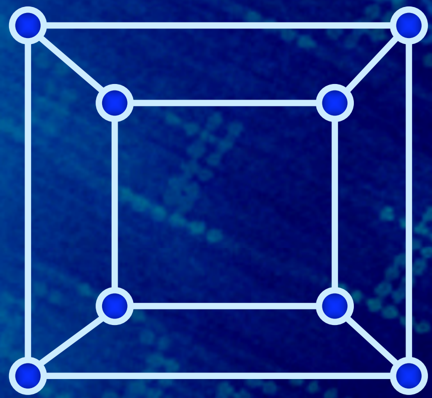


Regioni: esempio



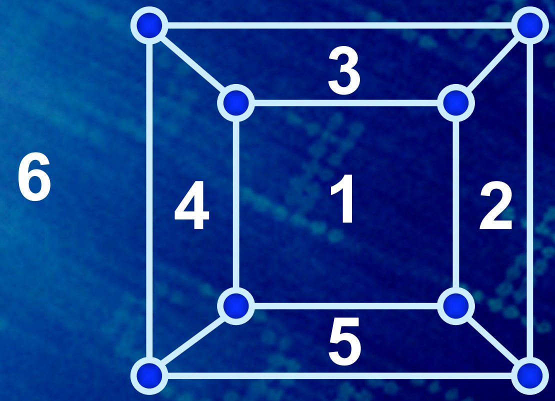
Regioni: esempio

- Quante regioni ha il seguente grafo?



Regioni: esempio

- Quante regioni ha il seguente grafo?



Teorema di Eulero (1752)

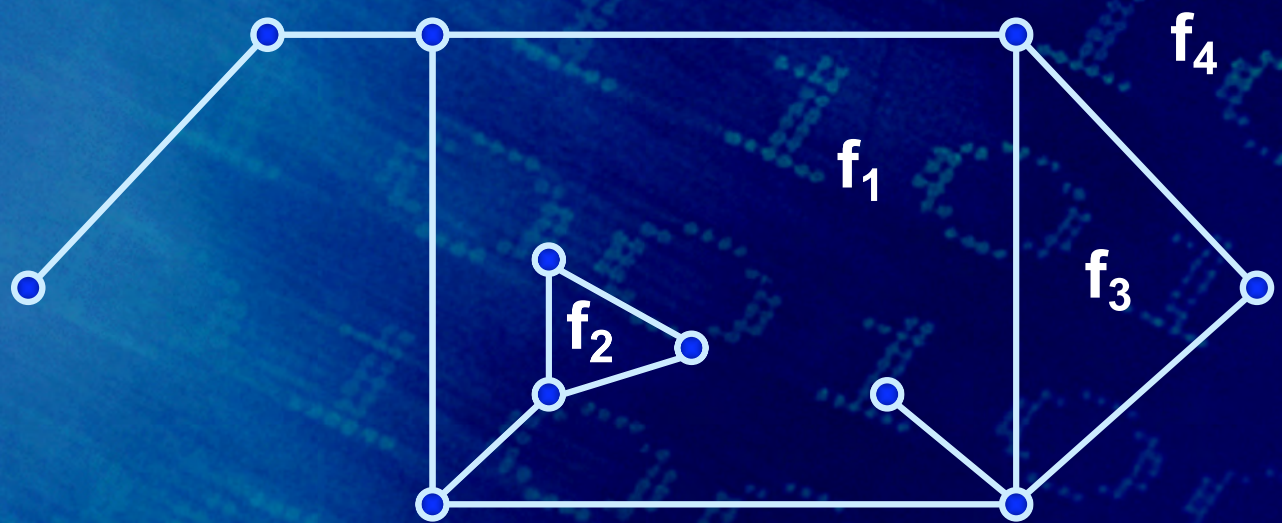
In un grafo piano connesso,

#regioni + #vertici = #archi + 2

Teorema di Eulero: esempio

#regioni + #vertici = #archi + 2

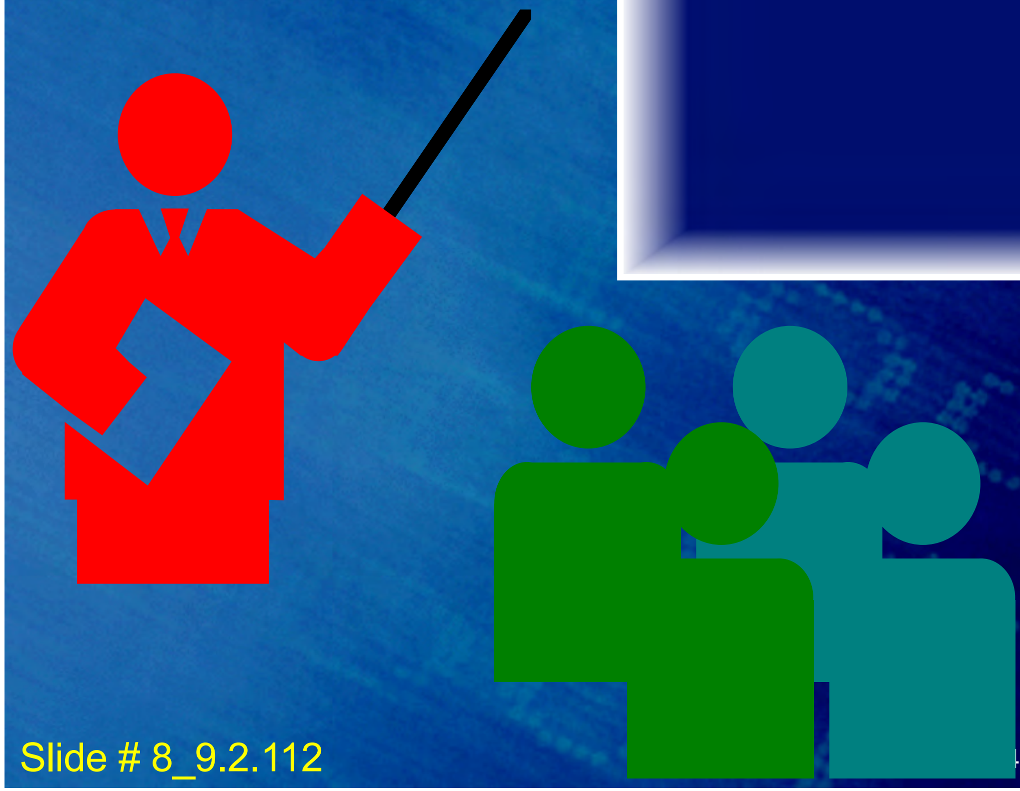
4 + 11 = 13 + 2



Corollario

In un grafo planare semplice connesso, con $|V|$ ($|V| \geq 3$) nodi e $|E|$ spigoli, si ha:

$$|E| \leq 3\,|V| - 6$$

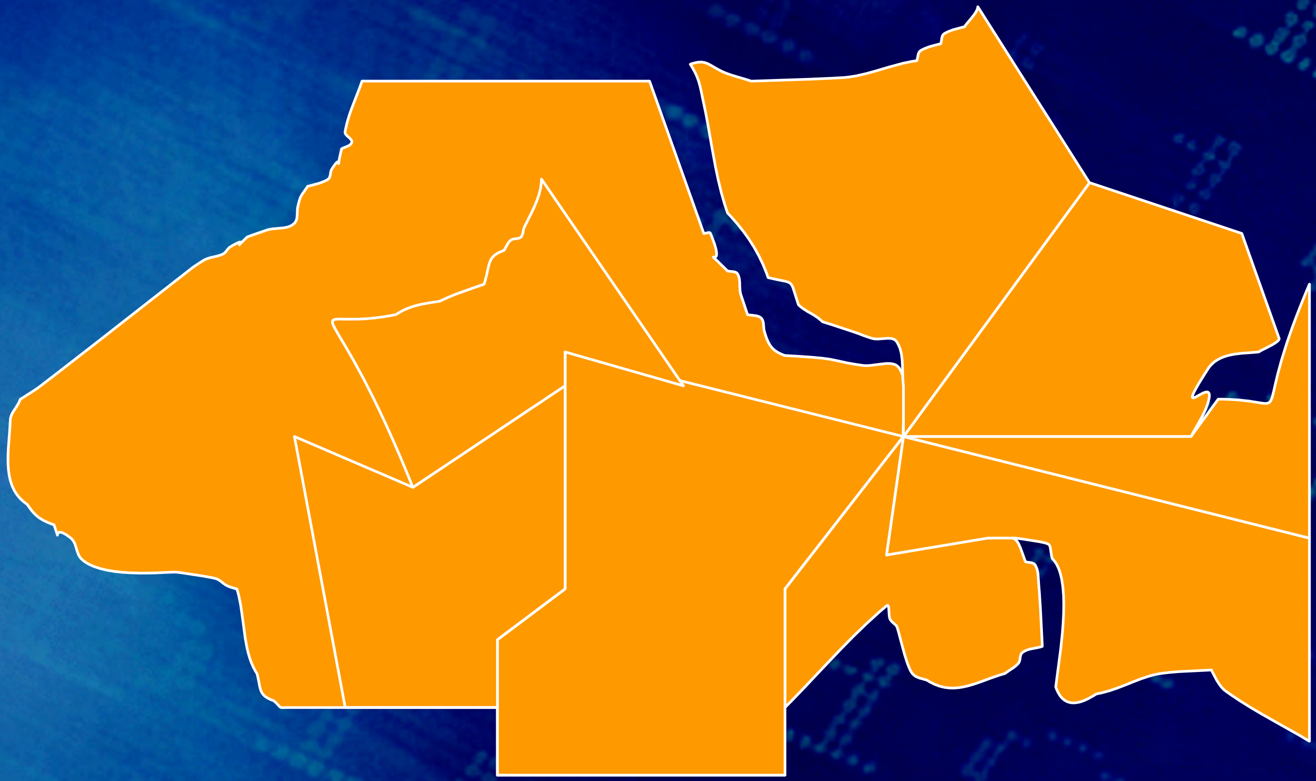


Outline

– Visite di un grafo
– Cammini e cicli
– Alberi e Foreste
– Cammini minimi
– Albero ricoprente
– Grafi Euleriani
– Grafi Hamiltoniani
– Isomorfismo
– Clique
– Planarità
– **Colorabilità**

Graph Coloring

Consider a fictional continent.



Map Coloring

Suppose removed all borders but still wanted to see all the countries. 1 color insufficient.

Map Coloring

So add another color. Try to fill in every country with one of the two colors.

Map Coloring

So add another color. Try to fill in every country with one of the two colors.

Map Coloring

So add another color. Try to fill in every country with one of the two colors.

Map Coloring

So add another color. Try to fill in every country with one of the two colors.

Map Coloring

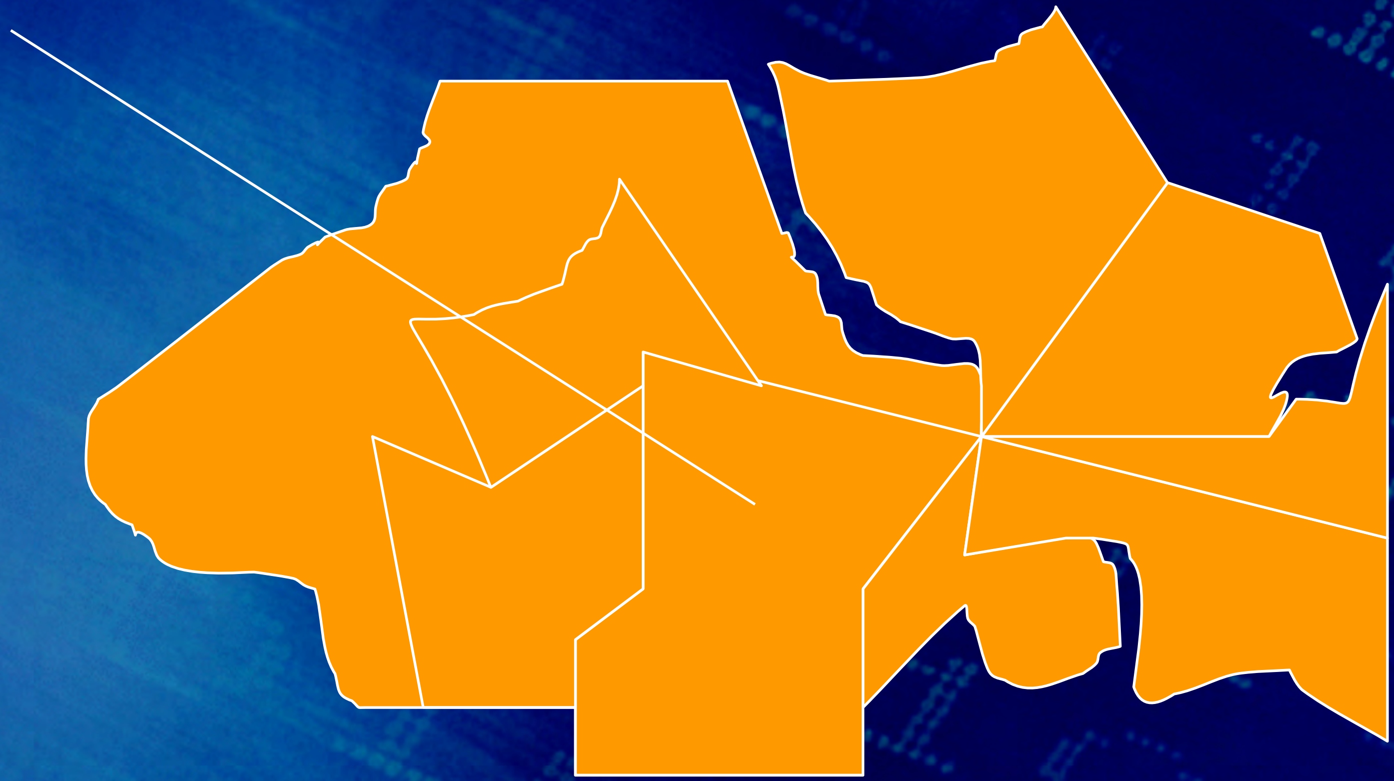
PROBLEM: Two adjacent countries forced to have same color. Border unseen.

Map Coloring

So add another color:

Map Coloring

Insufficient. Need 4 colors because of this country.



Map Coloring

With 4 colors, could do it.

Coloring a Graph - Applications

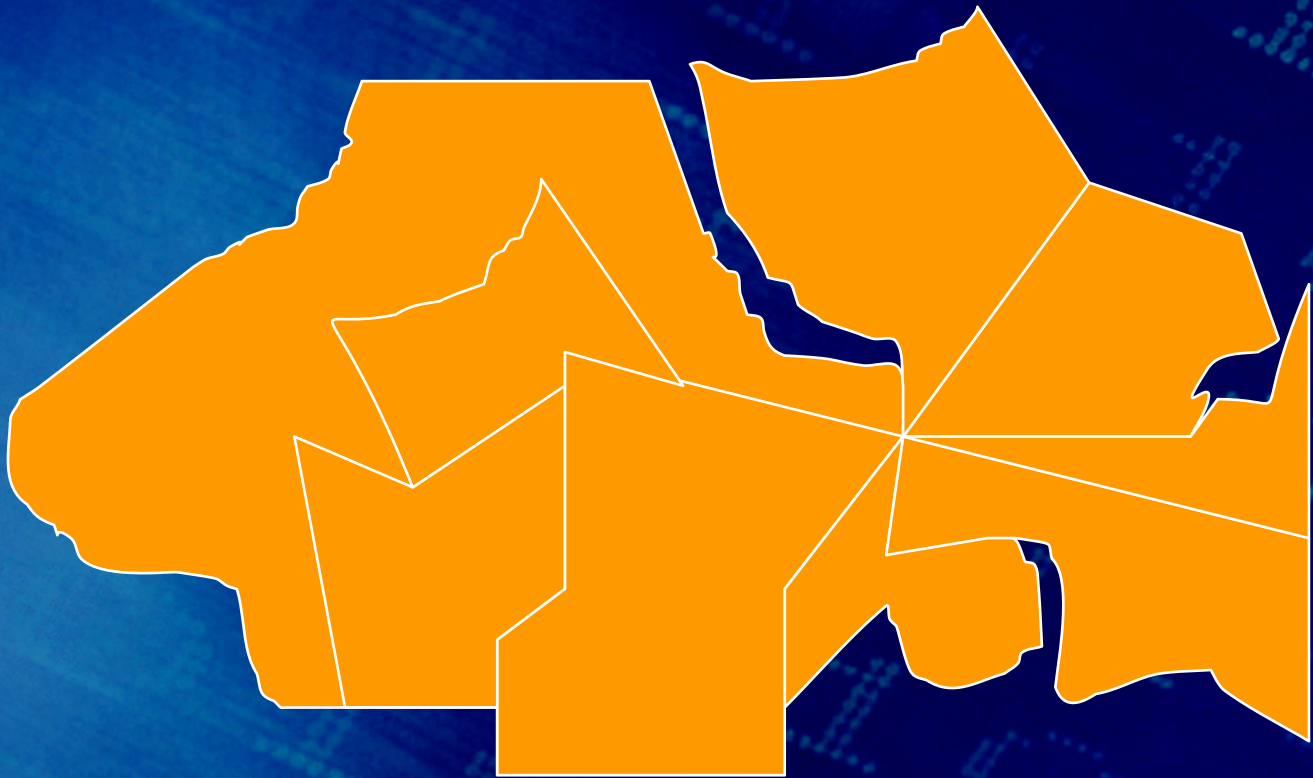
- Sudoku
- Scheduling
- Mobile radio frequency assignment
- Pattern matching
- Register Allocation
- …

Coloring a Graph - Applications

- Sudoku
- Scheduling
- Mobile radio frequency assignment
- Pattern matching
- Register Allocation
- …

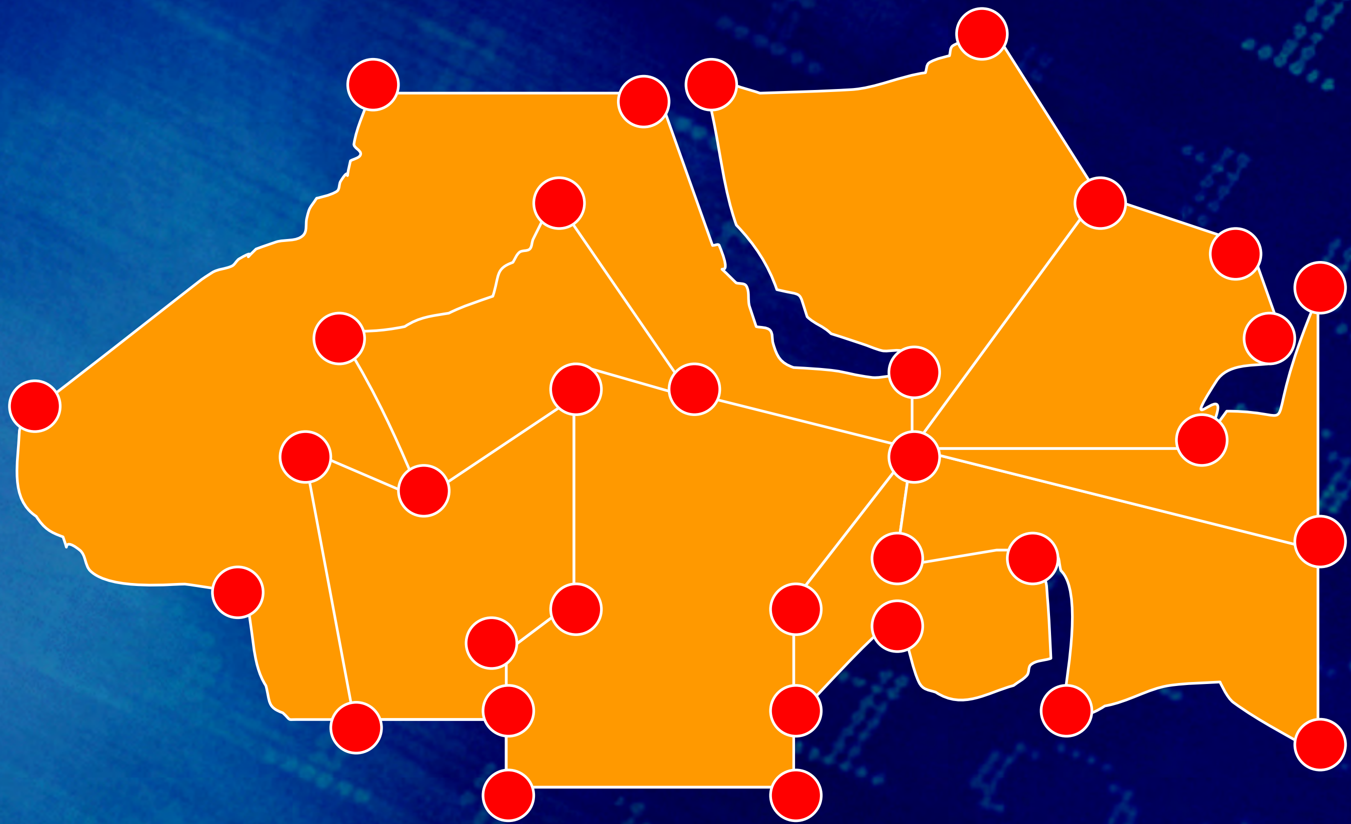
From Map Coloring to Graph Coloring

The problem of coloring a map, can be reduced to a graph-theoretic problem:



From Maps to Graphs to Dual Graphs

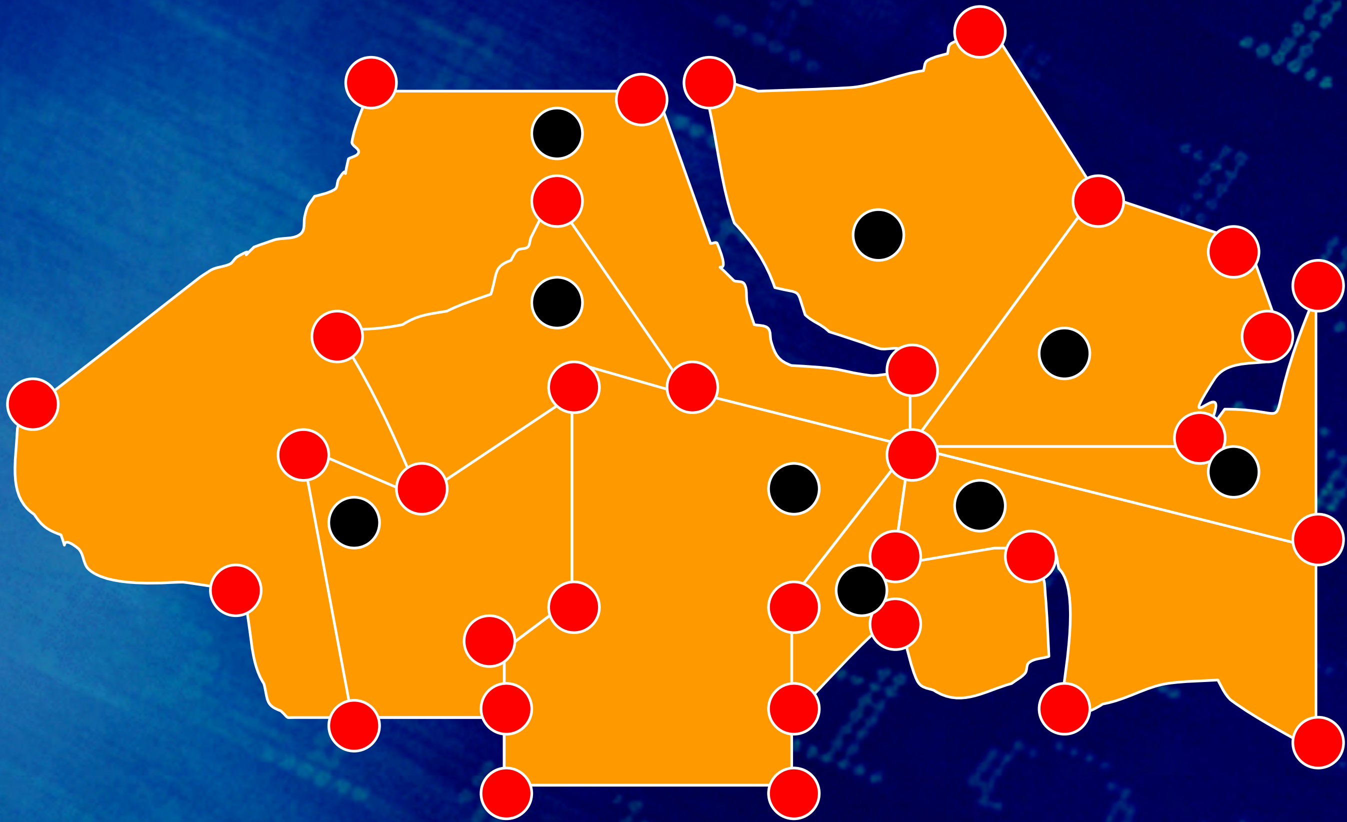
Really, could think of original map as a graph, and we are looking at dual graph:



From Maps to Graphs to Dual Graphs

Dual Graphs :

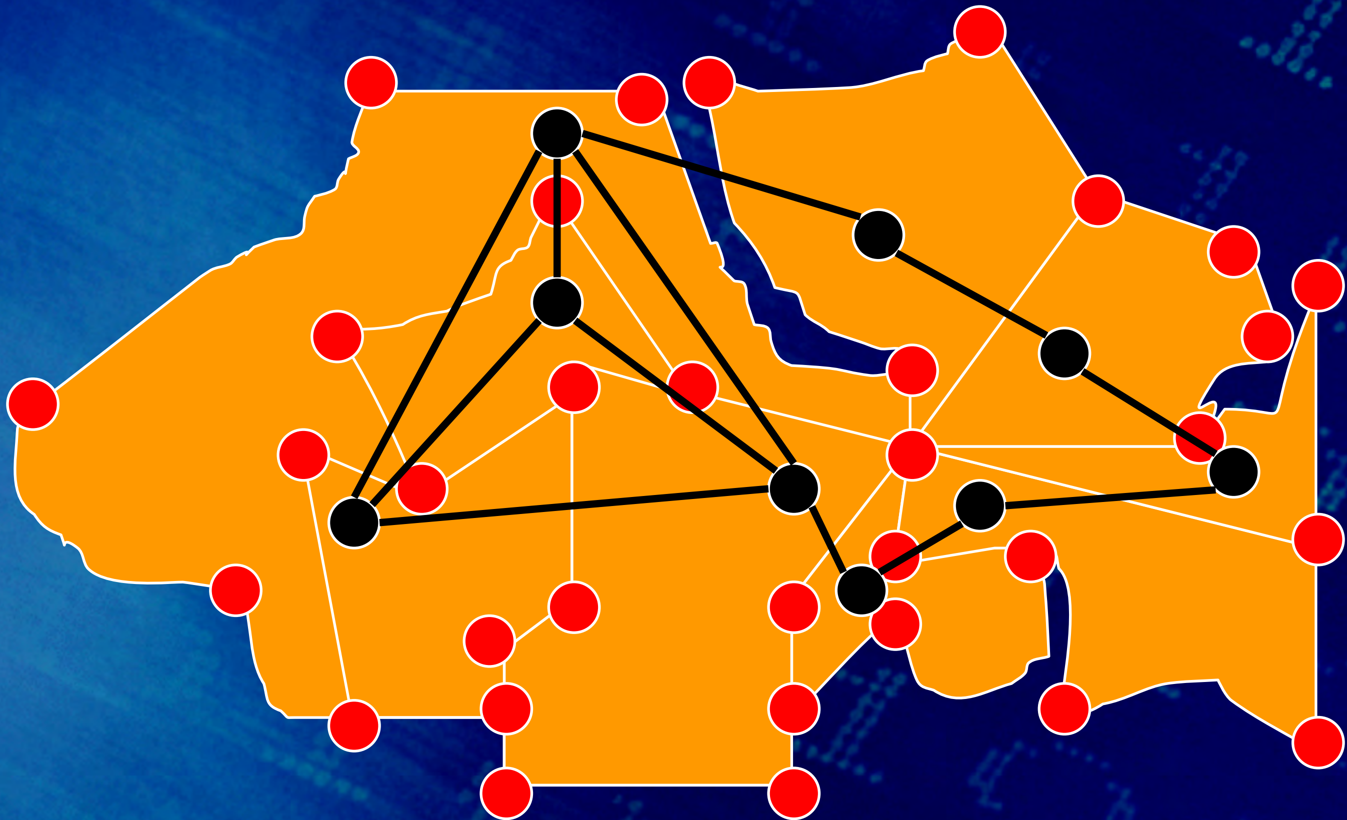
- **1) Put vertex inside each region:**



From Maps to Graphs to Dual Graphs

Dual Graphs :

- **2) Connect vertices across common edges:**



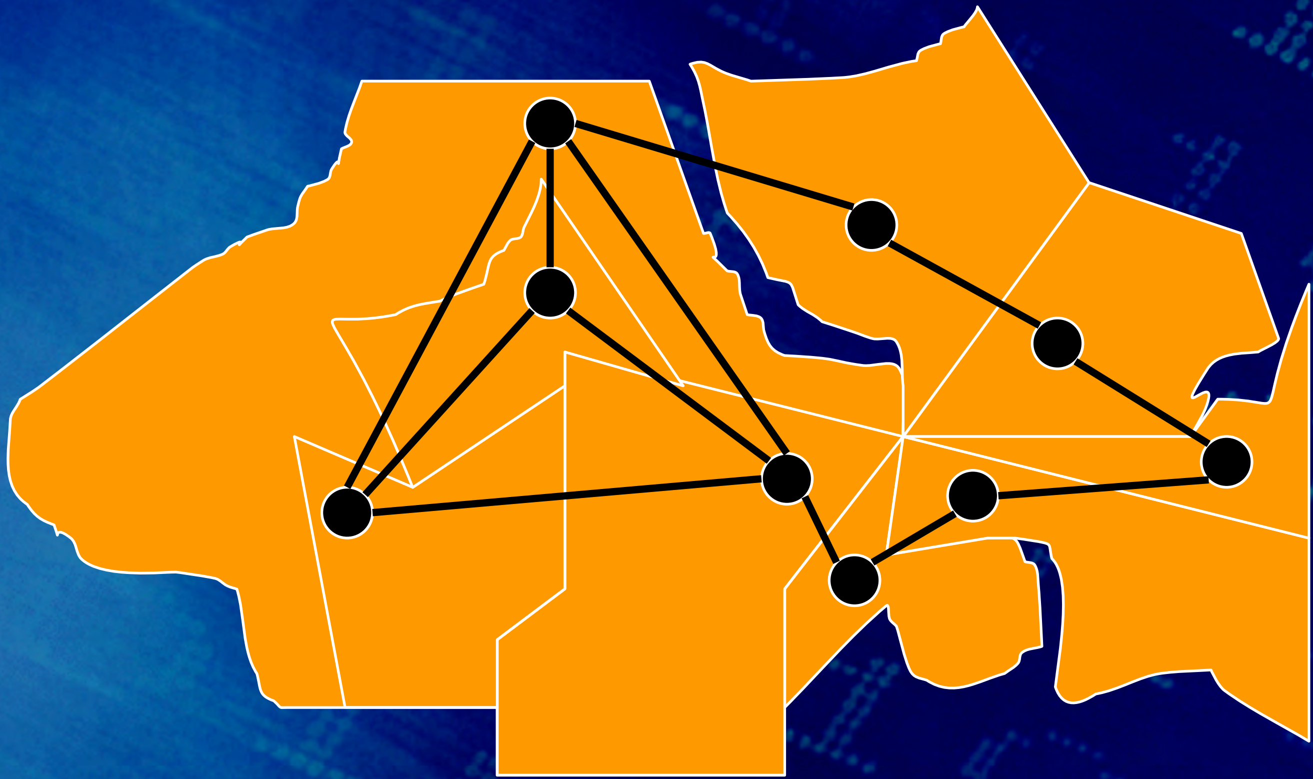
Definition of Dual Graph

The dual graph G^ of a planar graph G = (V, E, R) [Vertices, Edges, Regions] is the graph obtained by setting

- Vertices of G^: V (G^) = R
- Edges of G^: E (G^) = set of edges of the form {F1,F2} where F1 and F2 share a common edge.

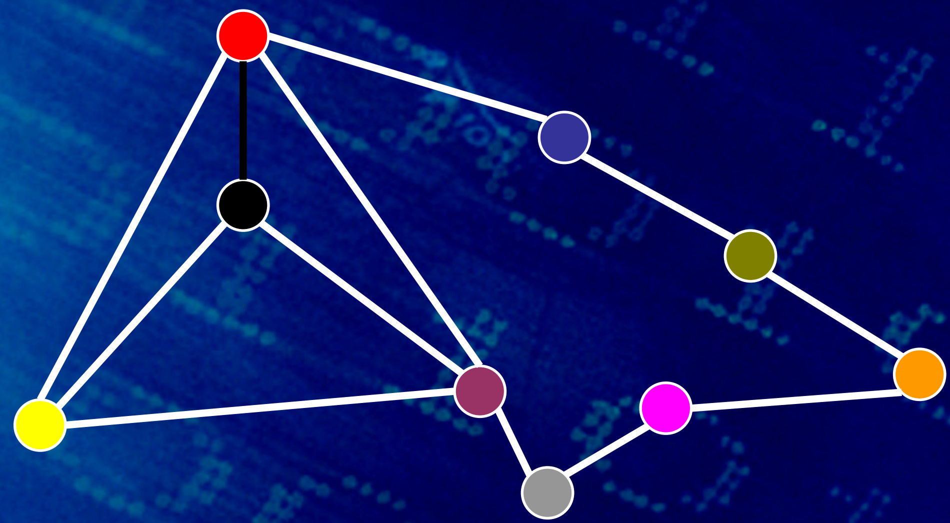
From Maps to Graphs to Dual Graphs

- So take dual graph:



From Map Coloring to Graph Coloring

- Coloring regions is equivalent to coloring vertices of dual graph.

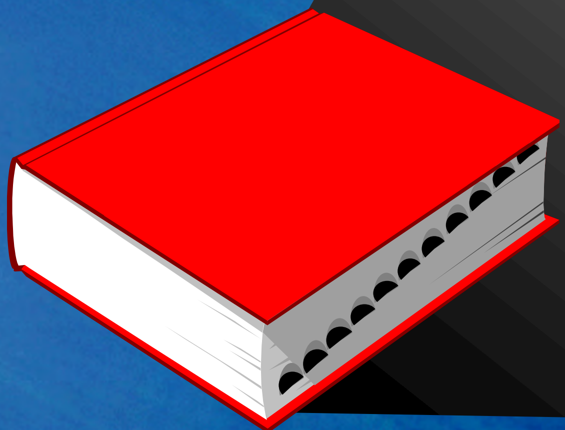


Colorabilità

- **Colorare** un grafo significa colorarne i vertici con uno o più colori diversi.
- Colorare correttamente un grafo significa colorarne i vertici in modo tale che due qualsiasi vertici adiacenti vengano colorati con colori diversi.
- Un grafo semplice non orientato è *k*-colorabile se è colorabile correttamente con *k* colori diversi.

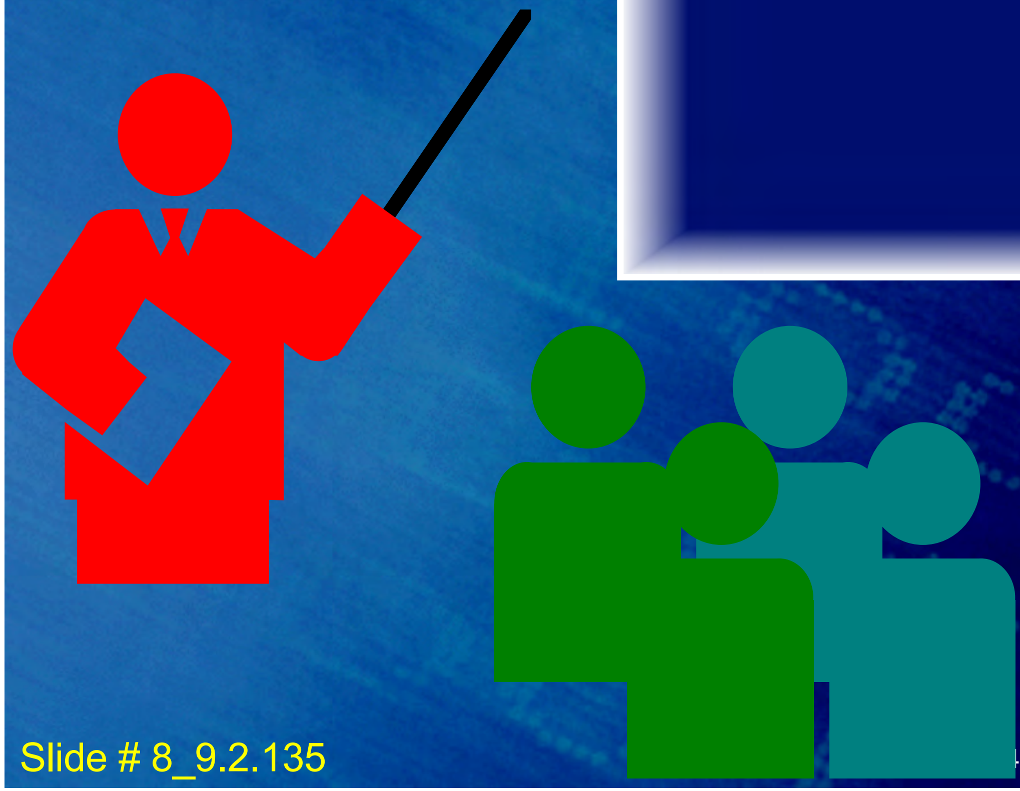
Numero cromatico

Il numero cromatico γ(G) di un grafo semplice non orientato G è il minimo numero di colori diversi necessari per colorare correttamente G.



4-Color Theorem

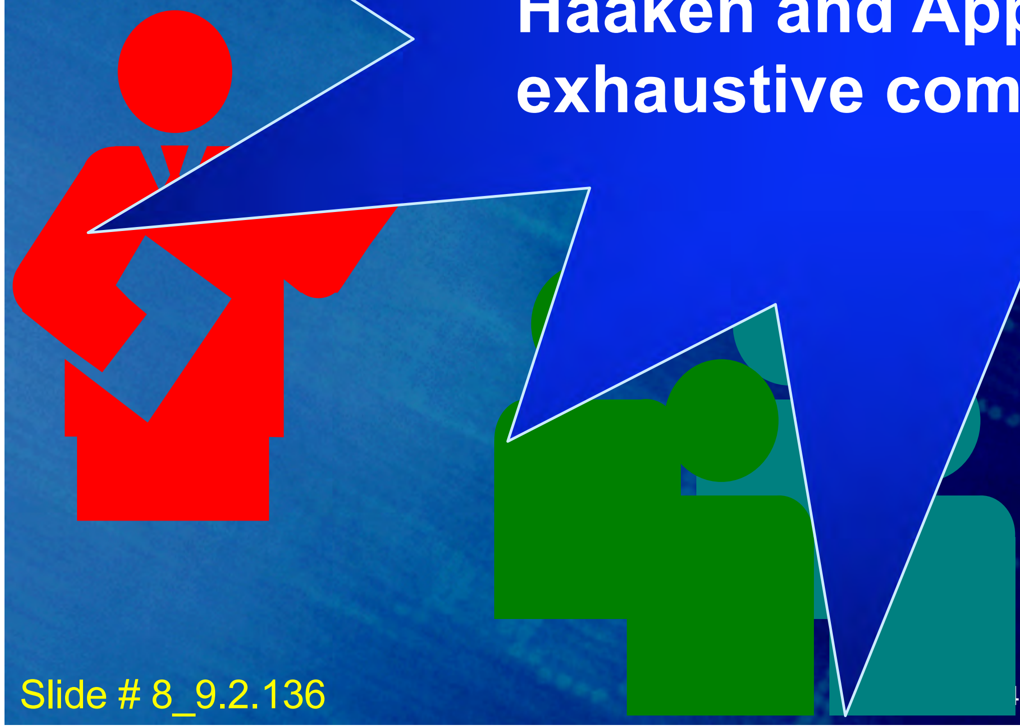
Any planar map of regions can be colored using 4 colors so that no two regions that share a positive-length border have the same color



4-Color eorem

A nar r egions can be
c t no
sitive-

First posed as a conjecture in 1852 by Francis Guthrie. Finally proved in 1976 by Haaken and Appel using exhaustive computer search



References

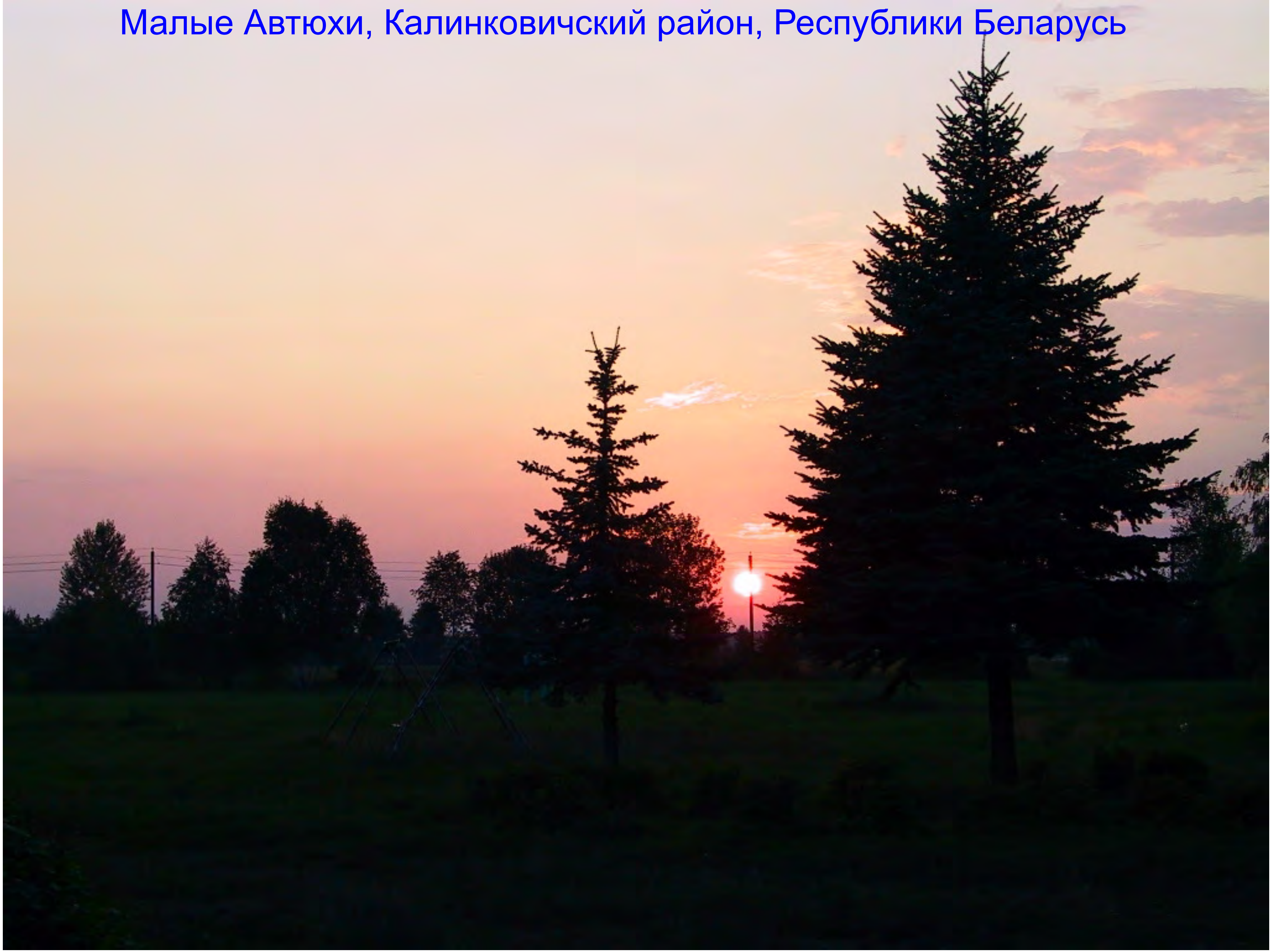
- A.V. Aho, J.E. Hopcroft, J.D. Ullman:
  "Data Structures and Algorithms,"
  Addison Wesley, Reading MA (USA), 1983
  pp. 198-252
- G. Ausiello, A. Marchetti-Spaccamela, M. Protasi:
  "Teoria e Progetto di Algoritmi Fondamentali,"
  Ed. Franco Angeli, Milano, 1985, pp. 265-364
- E. Horowitz, S. Sahni:
  "Fundamentals of Computer Algorithms,"
  Pittman, London (UK), 1978, pp. 272-325

References

- C.L. Liu:
  "Introduction to Combinatorial Mathematics,"
  McGraw-Hill Book Company, New York (USA),
  pp. 167-297
- R. Sedgewick:
  "Algorithms in C,"
  Addison Wesley, Reading MA (USA), 1990
  pp. 415-508
- C.J. Van Wyk:
  "Data Structures and C Programs," Addison
  Wesley, Reading MA (USA), 1988
  pp. 313-341

References

- **R.J. Wilson:**
  "Introduzione alla teoria dei grafi,"
  Cremonese, Roma 1978, pp. 1-155



Малые Автюхи, Калинковичский район, Республики Беларусь