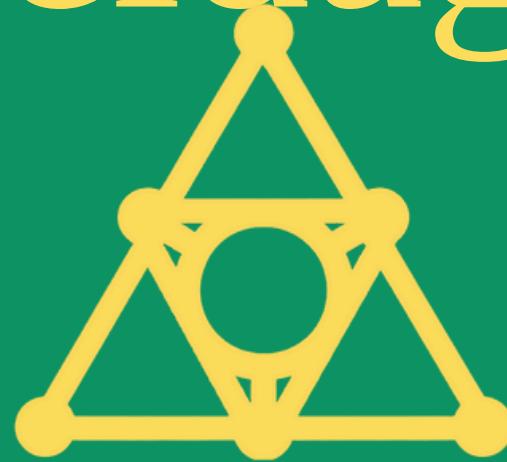


# Zeldager



Progetto di FIA

**Daniele De Martino**

0512116151

a.a. 2024/2025



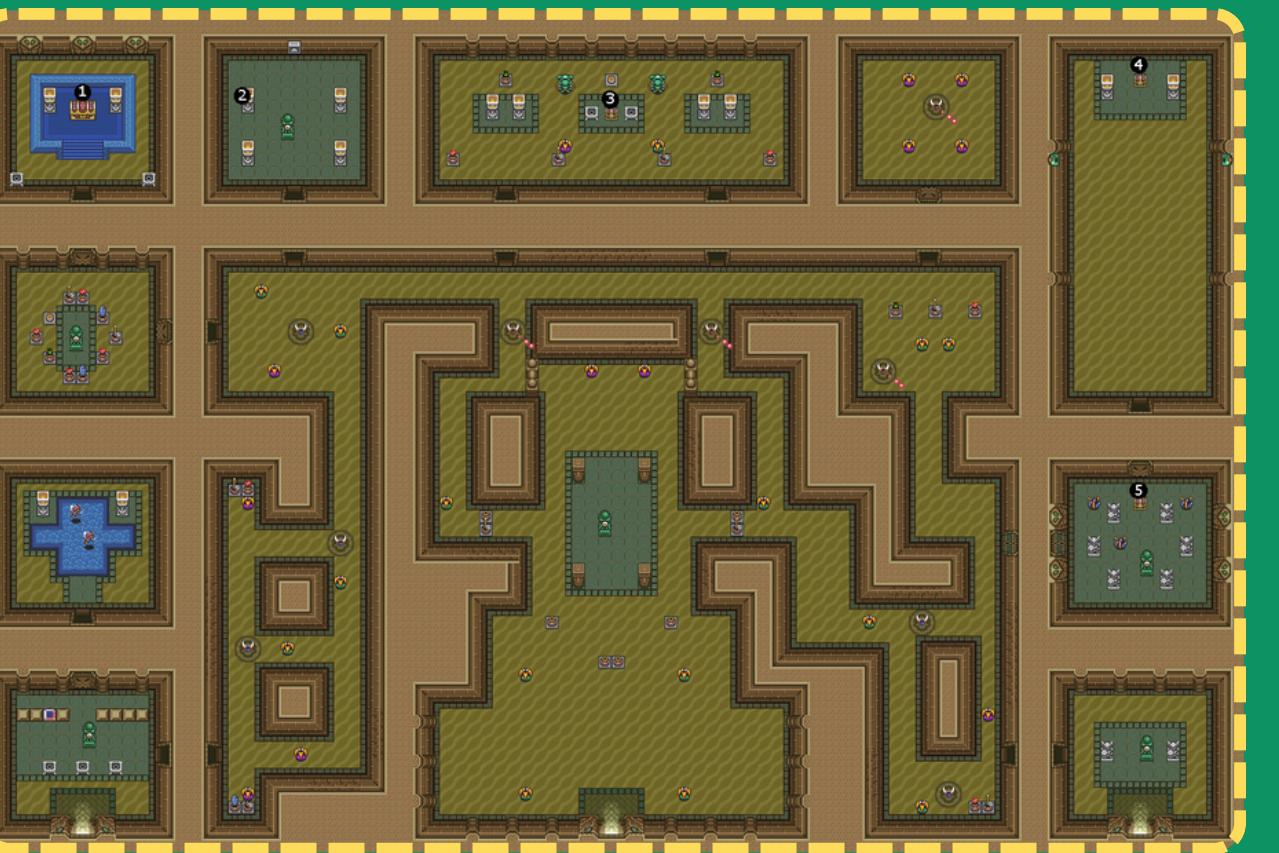
# Cosa?!

- **Zeldagen**, da “gen”-etico e da “gen”-erativo
- Dungeon in stile Zelda
- Game design di qualità
  - anche nella generazione procedurale!
  - compromesso **varietà vs qualità**



o, se parlasse una lingua antica,  
lo chiamereste...

לְבָנָה בְּנֵי נְגַדְּלָה





# Come:

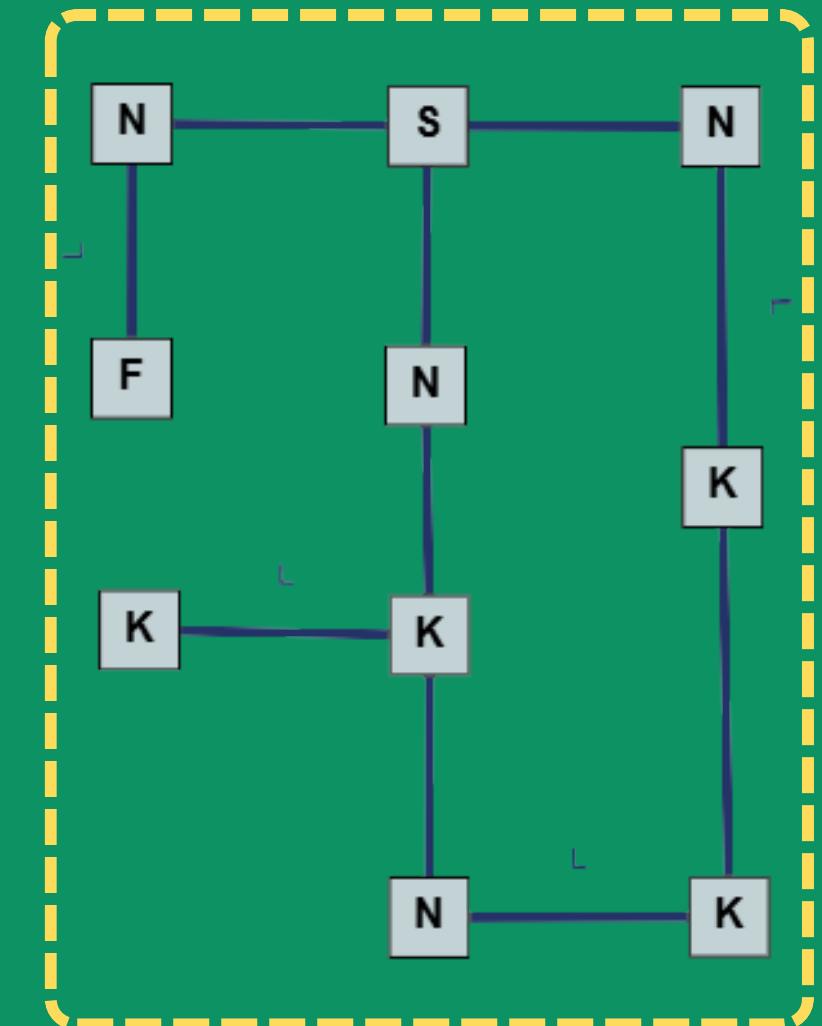
- Idea: un dungeon è un grafo!
- I grafi sono facili da generare...
  - ma non è facile ottenerne uno buono!

Suona familiare? Ottimizzazione  
incrementale? Ricerca locale!  
Magari algoritmi genetici...



PERCHÉ?  
Ci permettono di evolvere una popolazione  
di individui, così è più semplice non restare  
intrappolati in massimi locali!

## TOPOLOGIA DI DUNGEON



# Dungeon'

- Grafo non orientato
- I nodi sono le **stanze**, e possono essere di quattro tipi:
  - **iniziale** (S)
  - **finale** (F)
  - **normale** (N)
  - **con chiave** (K)
- Gli archi sono le **porte**, e possono essere aperte o chiuse a chiave! (L)



# Dungeon

- Durante una partita, si ha un certo numero di chiavi
  - e se ne può usare una per sbloccare una porta!
- Le stanze “con chiave” forniscono una chiave quando attraversate! Semplice, no?
- Esiste quindi una soluzione efficiente al dungeon...
  - ma a noi non interessava generarlo?





# Specifica P.E.A.S

## ENVIRONMENT

- SINGOLO AGENTE



- COMPLETAMENTE OSSERVABILE



- DETERMINISTICO



- STATICO



- EPISODICO



## PERFORMANCE

### METRICHE

- VALIDITÀ
- QUALITÀ

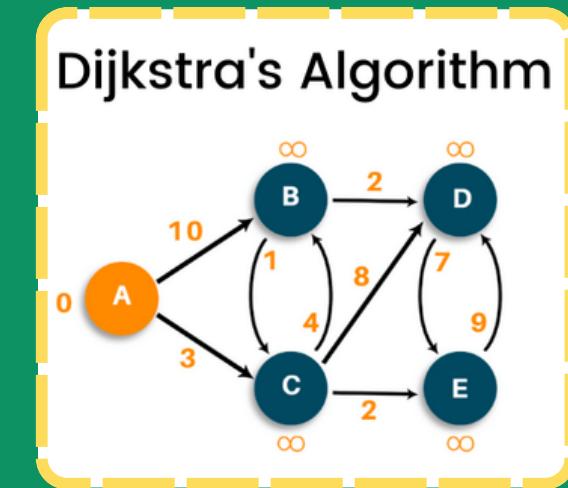
## ACTUATORS

### MODIFICA SUL GRAFO AZIONI POSSIBILI:

- AGGIUNTA DI ARCHI
- RIMOZIONE DI ARCHI
- CAMBIO DEL TIPO DI UNA STANZA
- ATTIVARE O DISATTIVARE UNA SERRATURA

## SENSORS

INFORMAZIONI SUL GRAFO E ALGORITMI EFFETTUABILI



# Fitness

- È la **chiave** del problema!
- Ho formalizzato delle **metriche di validità** e **di qualità**, matematicamente
- Nel codice, diventano:
  - **criteri di validità**
  - **criteri di penalità**
  - **criteri di premialità**

## COSA RENDE UN DUNGEON BUONO?

- Identificati 14 criteri
- Esempi:**
  - una sola stanza iniziale e finale
  - la stanza finale ha sempre una sola entrata chiusa a chiave
  - numero ragionevole di porte e di chiavi
  - la soluzione “**shortest path least keys**” esiste ed è bilanciata...





# Shortest che?

- “**Shortest path least keys**” è il nome del mio algoritmo risolutivo!
  - DFS modificata, con backtracking solo quando necessario
- **Idea chiave:** esplorare il grafo non orientato come un giocatore
  - attraversa una porta chiusa solo se ha una chiave
  - tiene traccia delle **chiavi**
    - e delle **porte già sbloccate**
  - torna indietro **solo** quando trova una chiave: potrebbe sbloccare porte precedentemente bloccate!
  - tenta diversi cammini, e seleziona il migliore: criterio **greedy**, percorso più breve!\*

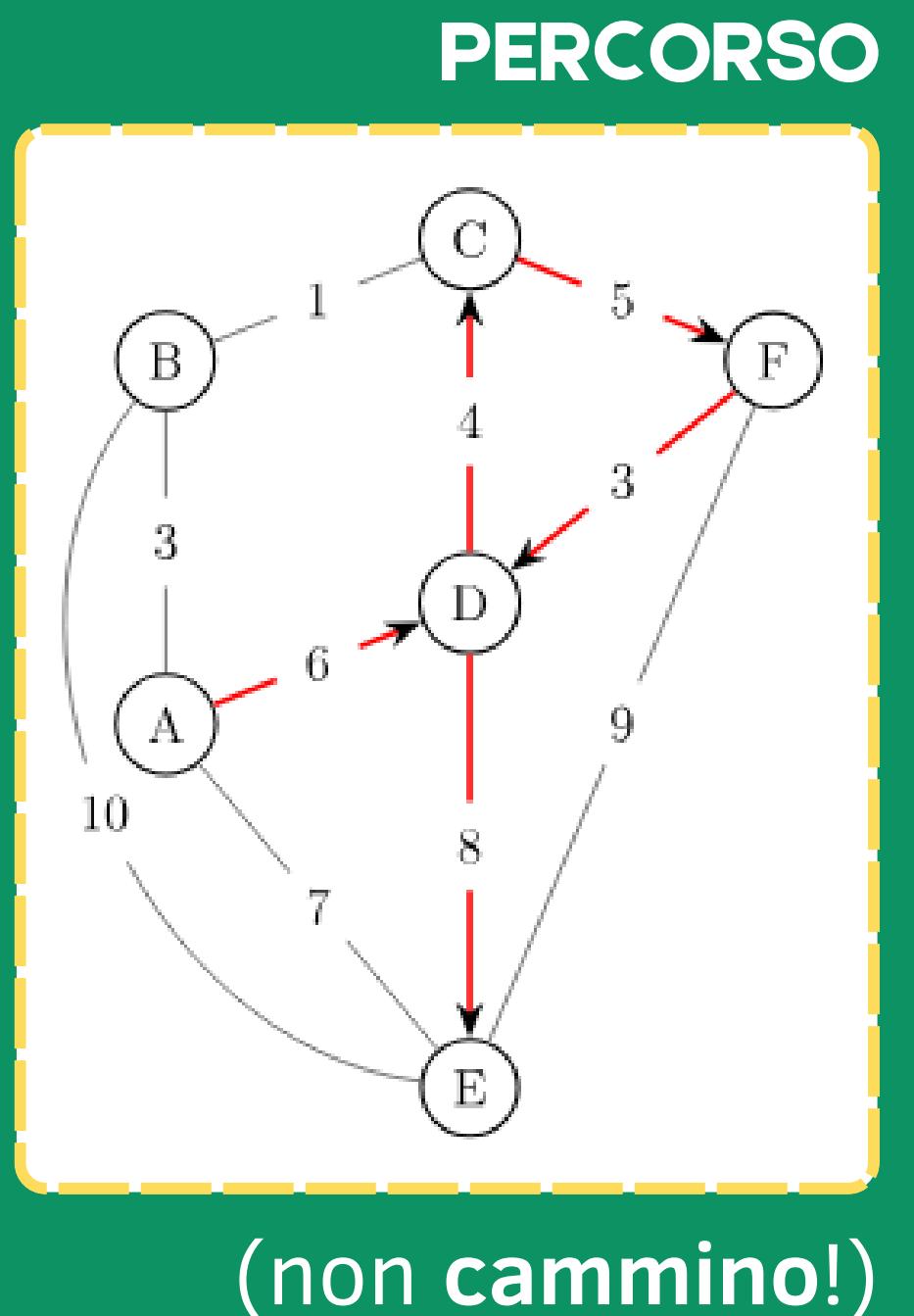
\* Nessuna dimostrazione! Ma assumiamo sperimentalmente...





# E la fitness?

- “Shortest path least keys” trova un **percorso** soluzione al dungeon. Ci assicuriamo che:
  - la soluzione **esista**
  - la difficoltà (lunghezza del percorso) sia **bilanciata**
  - il percorso sia **non-lineare** il giusto
    - secondo alcune fonti, la non-linearità è indesiderabile nel game design di un dungeon, ma una parvenza di non-linearità è ottima
    - => è meglio un percorso con poche ripetizioni!



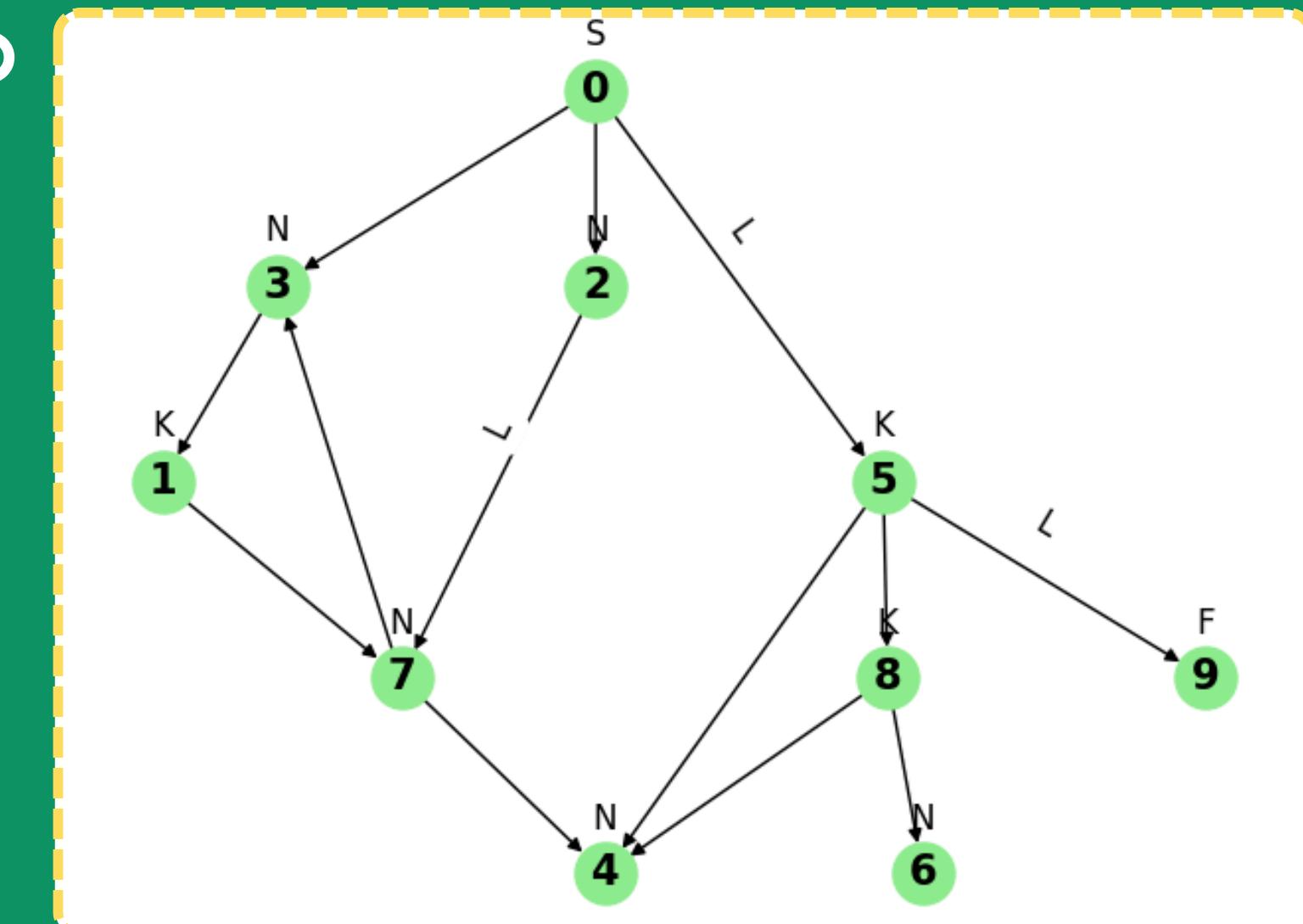
# Soluzione

## INDIVIDUO

- L'agente proposto genera questo:

## IPERPARAMETRI

- **popolazione**: fissa, di 15 individui (oppure k?)
- **sostituzione**: elitarismo, di 3 individui
- **criterio di arresto**: numero di generazioni



orientato?

## SELEZIONE

- provata la roulette wheel selection...
  - si è scelto il **3-way tournament!**

(in realtà,  $\left\lfloor \frac{k}{4} \right\rfloor$ -way tournament)

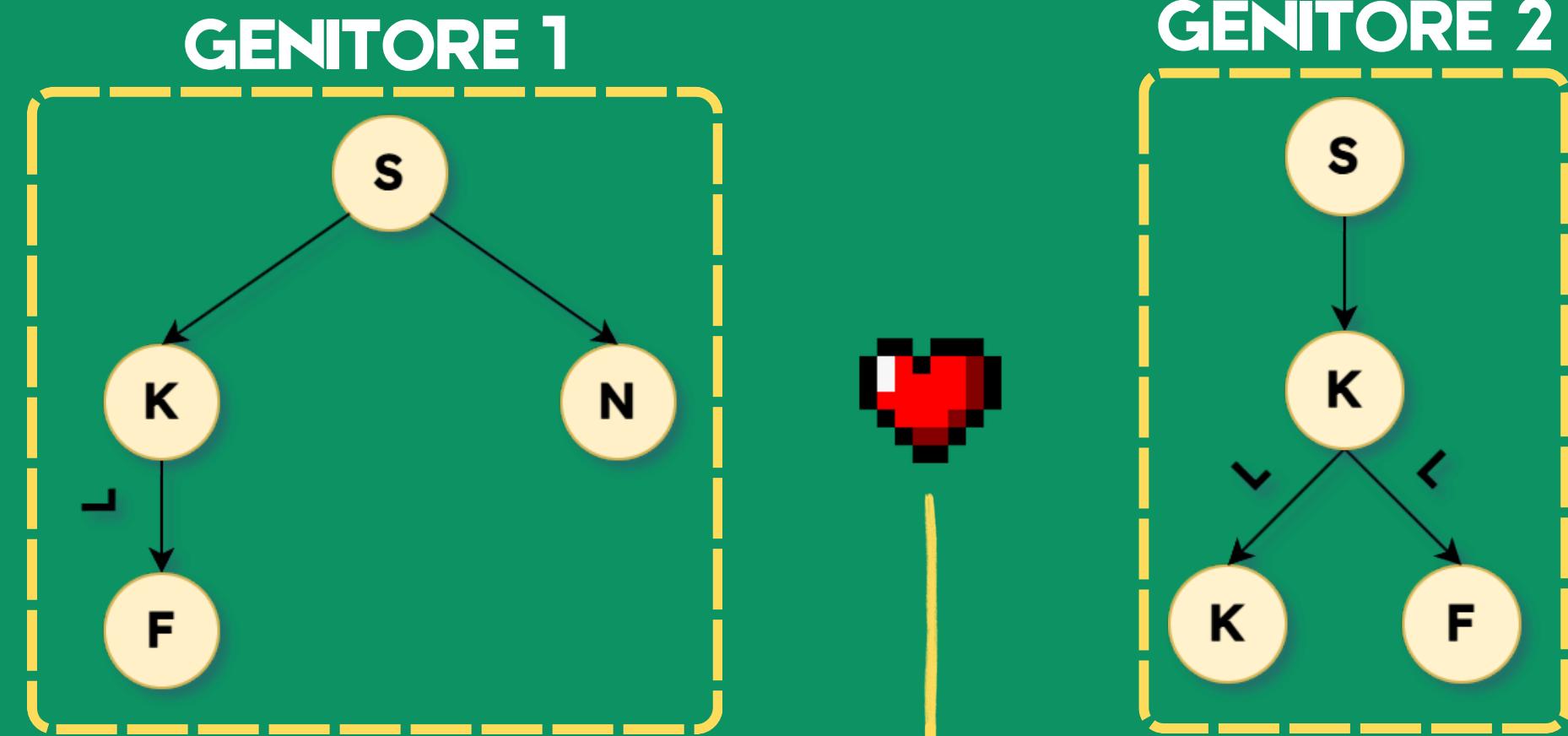
```
...  
Selection algorithm  
  
def selection(population, population_size):  
    """ Selects a single individual through k-way tournament. """  
    tournament = random.sample(population, population_size // 4)  
    return sorted(tournament, key=lambda ind: ind.fitness(), reverse=True)[0]
```



# Soluzione

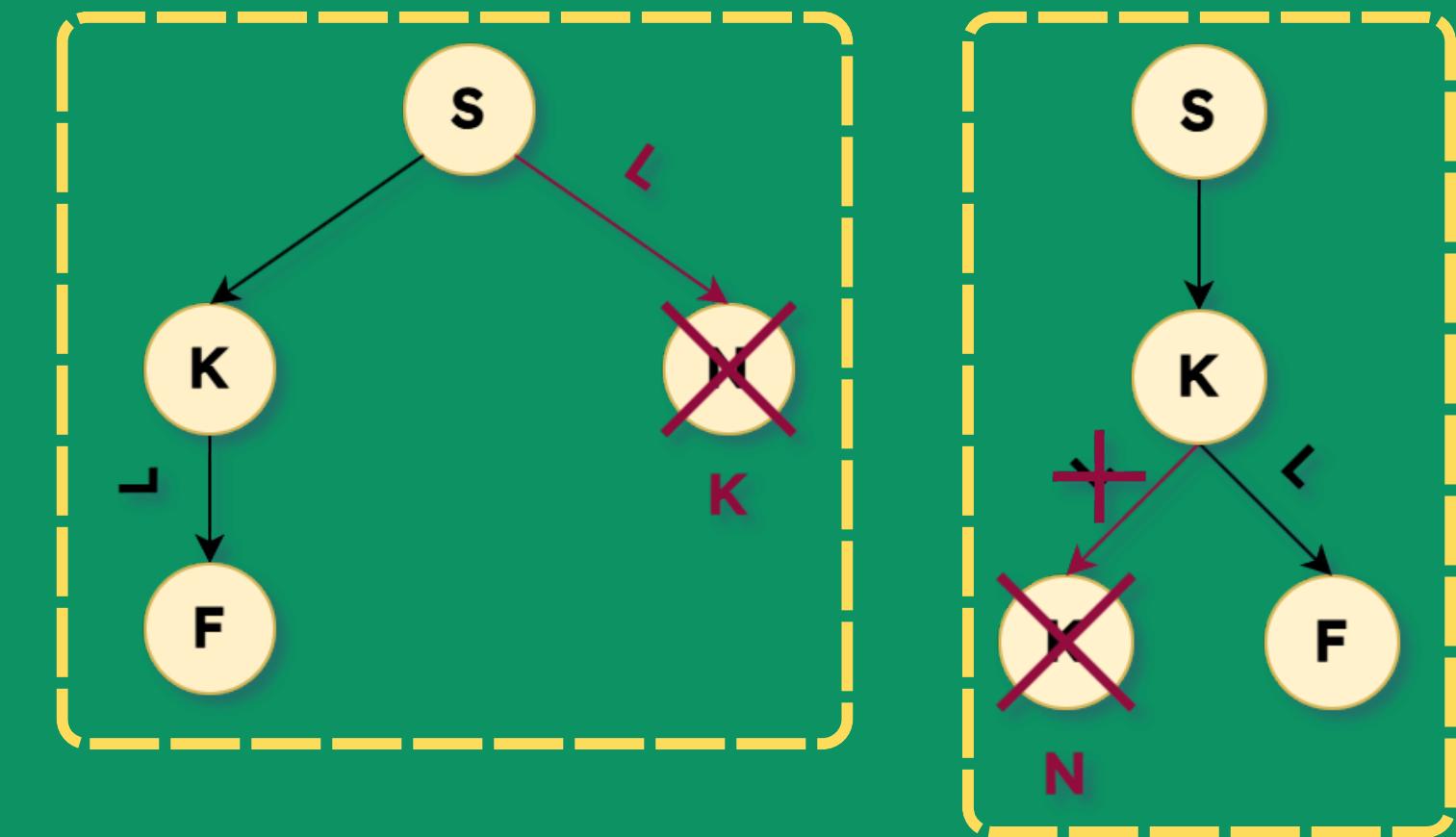
## CROSS-OVER

- difficile
- scambio di due “foglie”!
- poco **estensivo e aleatorio**



## VINCOLI

- **foglie?** ma non è un albero!
- non modifichiamo la posizione delle stanze iniziali e finali
- a volte potrebbero non esserci “foglie”





# Soluzione

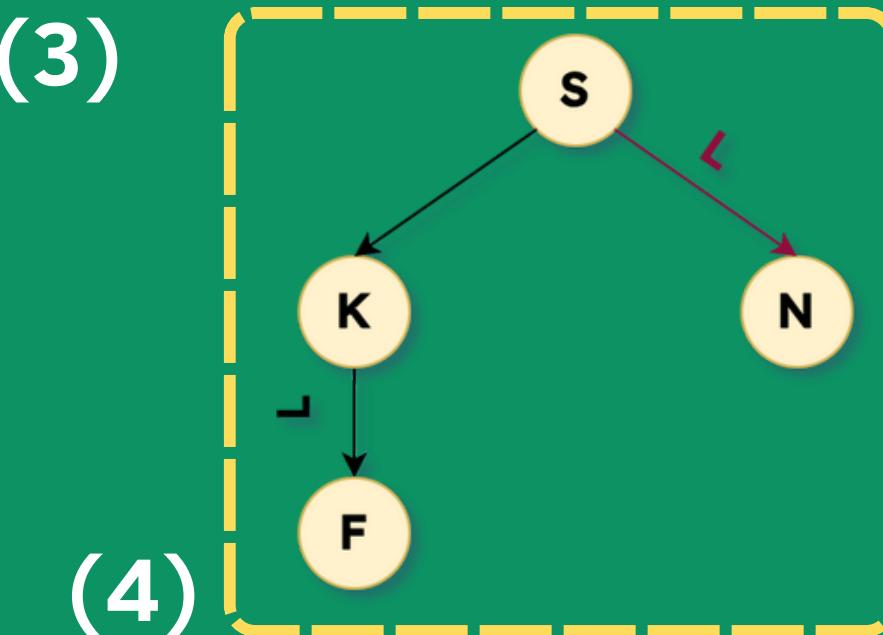
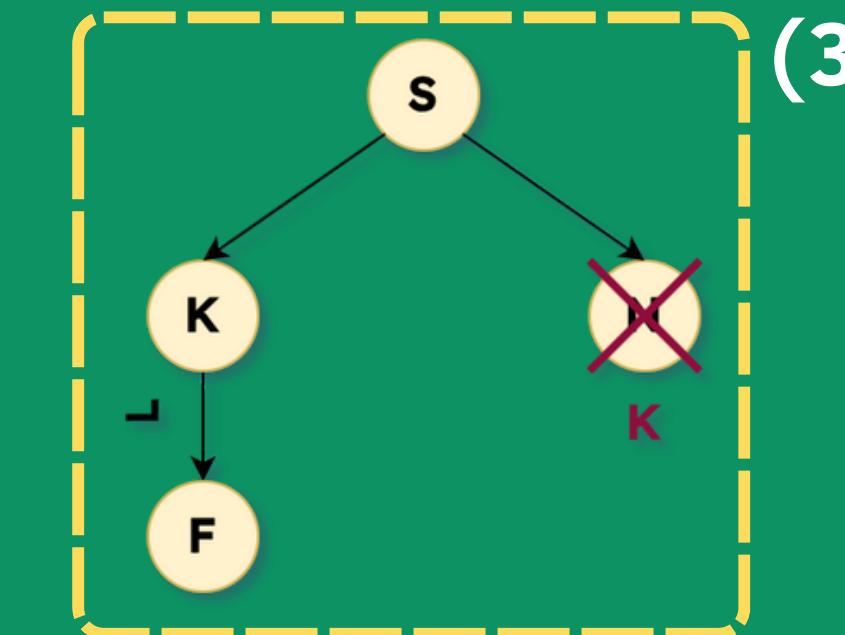
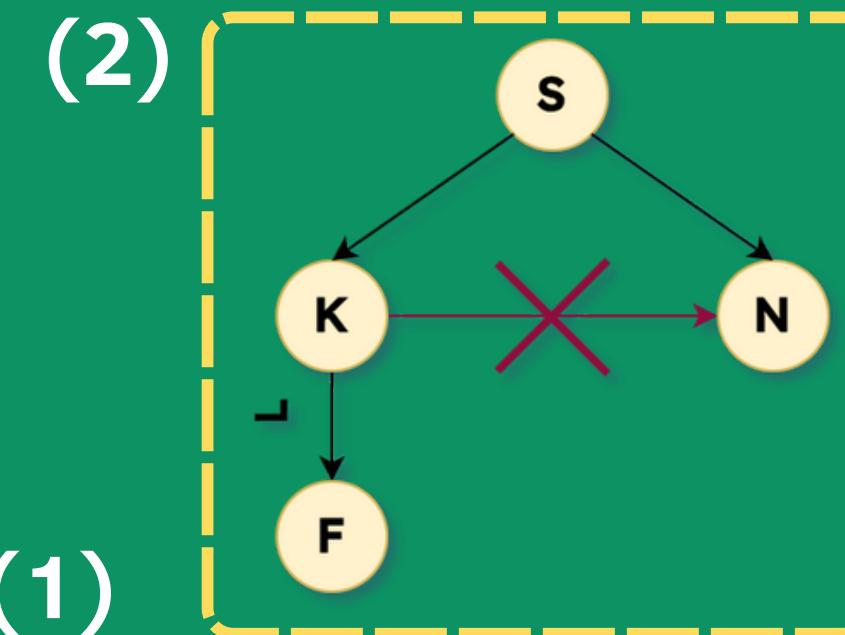
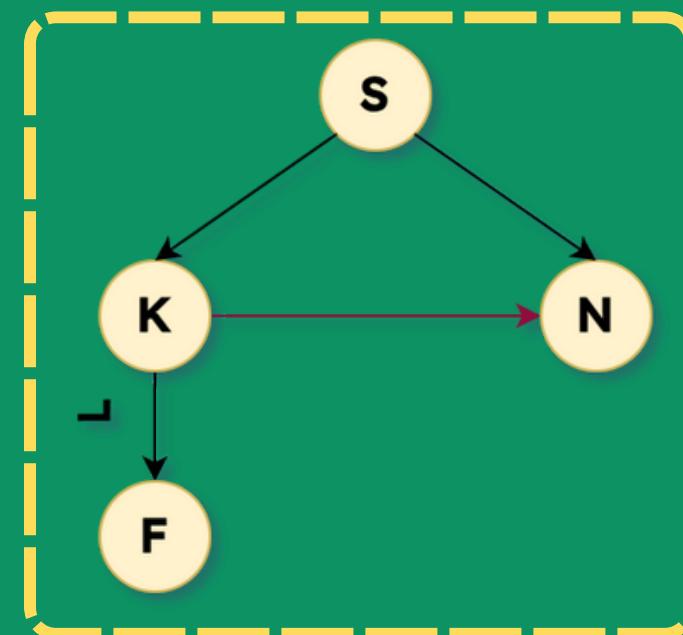
## MUTAZIONE

- semplice
- le azioni possibili, scelte a **caso**:

- (1) aggiunta di un arco
- (2) rimozione di un arco
- (3) cambiamento pseudo-casuale del tipo di una stanza
- (4) cambiamento pseudo-casuale di una serratura

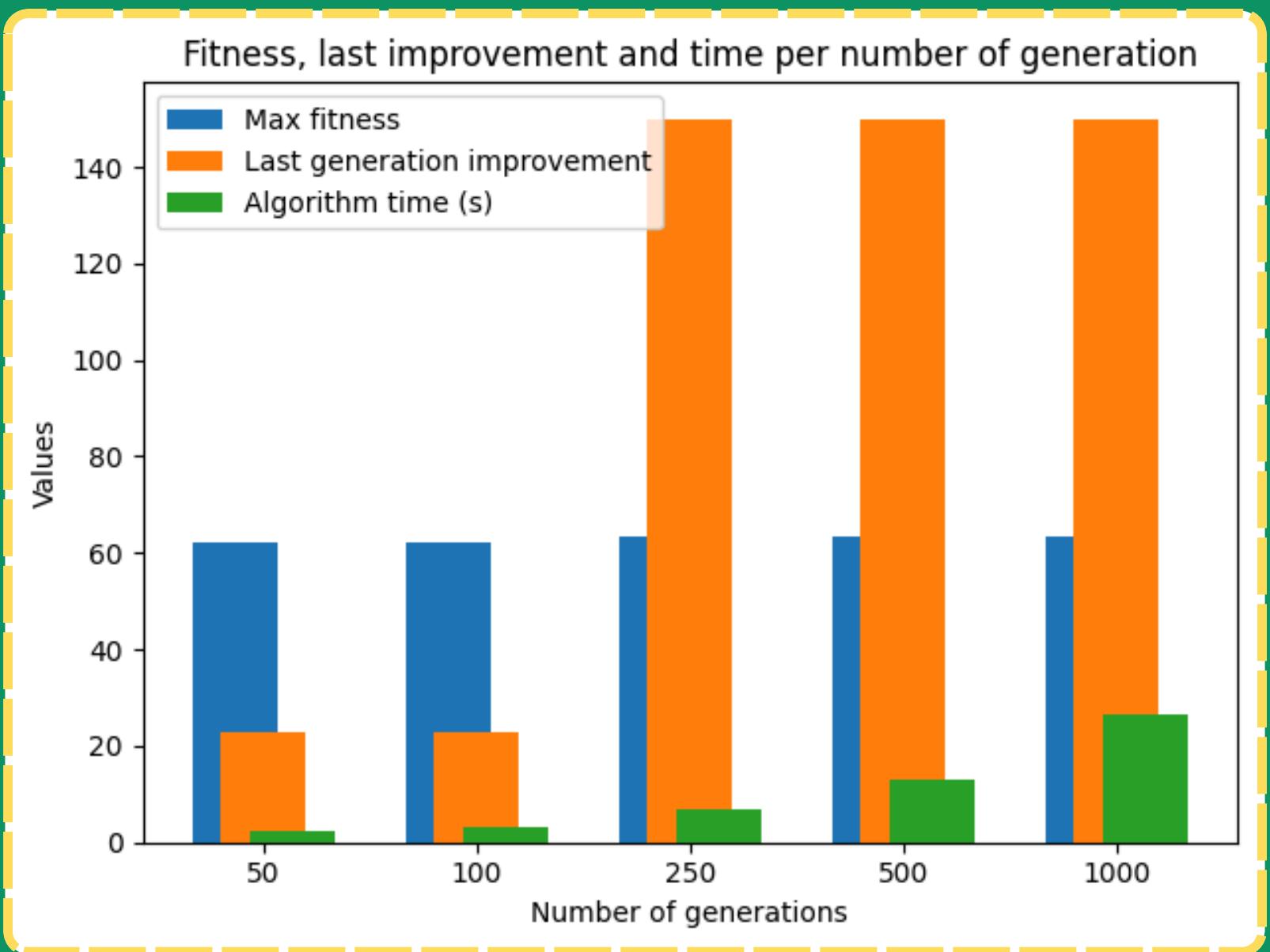
## VINCOLI

- e se disconnettiamo il grafo?
- a volte alcune di queste cose ha più senso non farle... **euristiche!**



# Benchmar]

- I risultati ottenuti sono **fantastici!**
  - Ed estremamente **veloci**...
- **Troppo facile?** Strano!
- Convergenza molto veloce
  - indipendentemente dalla pressione selettiva
  - non necessariamente un male
- Il problema è semplice, aleatorio e con troppi vincoli
  - algoritmo genetico: forse non la scelta migliore!



50 generazioni: 3.3s, ultimo miglioramento 25°

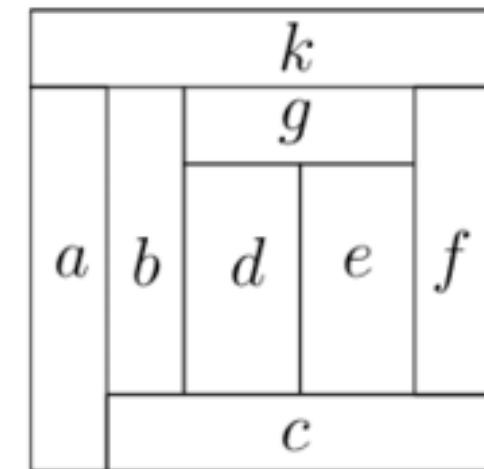
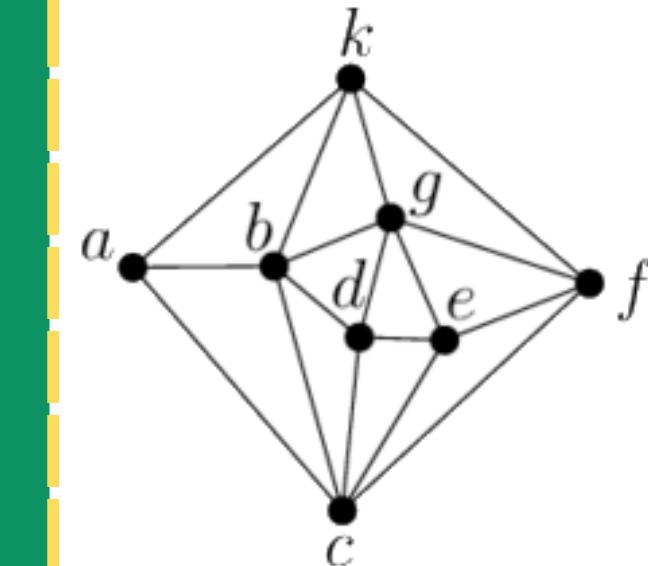




# Dem

- L'idea originale era fare una mappa
  - ma, matematica difficile eccetera eccetera...
  - è dimostrabilmente **impossibile!**\*
- **Idea:** un gioco! Ma è complesso...
  - magari un gioco testuale con un LLM che narra?
- **Soluzione:** un applicativo grafico da terminale che permette di “giocare” un dungeon, esplorandolo
  - la narrazione la fa **OLLaMa!**
  - lo stato del gioco lo gestisce il programma, con qualche eccezione  
(OLLaMa non è bravo a ricordare: allucina spesso, d'altronde è un lama!)

MATEMATICA DIFFICILE!



MATEMATICA DIFFICILE?



\* per i grafi generati dal mio algoritmo

O DemoZeldagen

**Statistics**

- 🔑 1
- ❤️ 19
- 💎 15
- 🗺️ no

[Show map >](#)

**Actions**

What path do you want to take?

- Room 1 (locked)
- Room 5

[Pick >](#)

**Narrative**

You find yourself in a small chamber filled with dusty shelves and forgotten treasures. The air is thick with the scent of old parchment and the faint hint of metal. In the center of the room, you notice a pedestal bearing a rusty key. The key lies on a small wooden box adorned with intricate carvings of locks and keys.

As you approach the pedestal, you hear a soft click, and the box opens, revealing the key inside. You pick up the key, feeling its weight in your hand. It

**History**

You are in Room 0. Adventure begins!  
You are in Room 3.  
You are in Room 5.  
You are in Room 7. You found a key.

 **d** Toggle dark mode

|^p palette

Grazie!

