Algoritmi di approssimazione

A.A. 2017-2018

Limiti dell'prrossimazione: TSP

Algoritmi di approssimazione

Abbiamo detto che per tutti i problemi di ottimizzazione si dispone di un algoritmo di approssimazione.

Come consseguenza del teprema precedentemente visto per alcuni problemi famosi come "**Cricca**", "**Colorabilità**" e "**Commesso viaggiatore**" trovare una soluzione approssimata con un rapporto di approssimazione indipendente dall'input (come per esempio nel caso della copertura di vertici) è difficile, in termini di complessità, tanto quanto trovarne una ottima.

Dimostriamo che questo si verifica, per esempio, nel caso del Commesso Viaggiatore.

Commesso viaggiatore

*Il problema è, nella versione decisionale, **NP**-completo.*

Vedremo che esiste un algoritmo approssimato solo se la funzione costo soddisfa la proprietà della “disuguaglianza triangolare”.

Nel caso generale invece, vale il seguente:

Teorema:

Se **P** ≠ **NP**, non esiste alcun algoritmo approssimato in tempo polinomiale che risolva il problema del commesso viaggiatore con rapporto limite costante ρ ($\rho \geq 1$).

Dimostrazione

Supponiamo, per assurdo, che esista un algoritmo approssimato A con tempo polinomiale e rapporto limite $\rho \geq 1$.

Commesso viaggiatore

Dimostriamo che l'esistenza dell'algoritmo A implicherebbe l'esistenza di un algoritmo polinomiale per il problema "circuito hamiltoniano", cioè la sua appartenenza a **P**.

Usiamo la seguente trasformazione polinomiale tra "circuito hamiltoniano" e "commesso viaggiatore".

Per ogni istanza G = (V,E) di circuito hamiltoniano si costruisce un'istanza <G' = (V,E'), w> di commesso viaggiatore:

E' = $\{(u, v)$ per ogni $u, v \in V$ e $u \neq v\}$

- $w(u,v) = 1$ se $(u,v) \in E$
- $w(u,v) = \rho \times |V|+1$ altrimenti

Commesso viaggiatore

Se il grafo G ha un ciclo hamiltoniano H, allora la funzione peso w assegna a ciascun arco di H costo 1

<G′, w> contiene un giro di costo |V|

Se G non contiene alcun ciclo hamiltoniano, allora qualsiasi ciclo g di G′, anche quello minimo per il commesso viaggiatore, deve usare almeno un arco che non è in E, e il suo costo C(g) è perciò:

$$C(g) \geq (\rho \times |V|+1) + (|V|-1) = (\rho+1) \times |V| > \rho \times |V|$$

Commesso viaggiatore

Poiché A garantisce di trovare una soluzione al più ρ volte peggiore di quella ottima se G contiene un circuito Hamiltoniano A lo deve trovare.

Se l'algoritmo A esistesse avremmo un algoritmo polinomiale per "ciclo hamiltoniano", ma nell'ipotesi **P** ≠ **NP** ciò è assurdo.

Commesso viaggiatore

Il problema è, nella versione decisionale, ***NP****-completo.*

Per il problema generale non esistono buoni algoritmi approssimati, che si possono però ottenere se la **funzione costo** soddisfa la seguente proprietà, detta **disuguaglianza triangolare**:

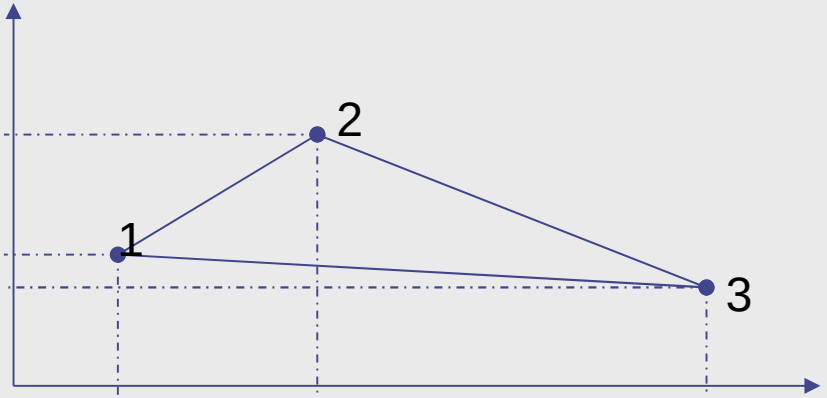
$$c(u,w) \leq c(u,v) + c(v,w) \qquad \forall\, u, v, w \in V[E]$$

In particolare, per il problema del commesso viaggiatore con funzione costo che soddisfa la disuguaglianza triangolare, esiste un algoritmo approssimato avente rapporto limite uguale a 2.

Commesso viaggiatore

(disuguaglianza triangolare)

Considerando i vertici di un grafo come punti del piano cartesiano e il costo degli archi come la distanza tra essi, la disuguaglianza triangolare è automaticamente soddisfatta.



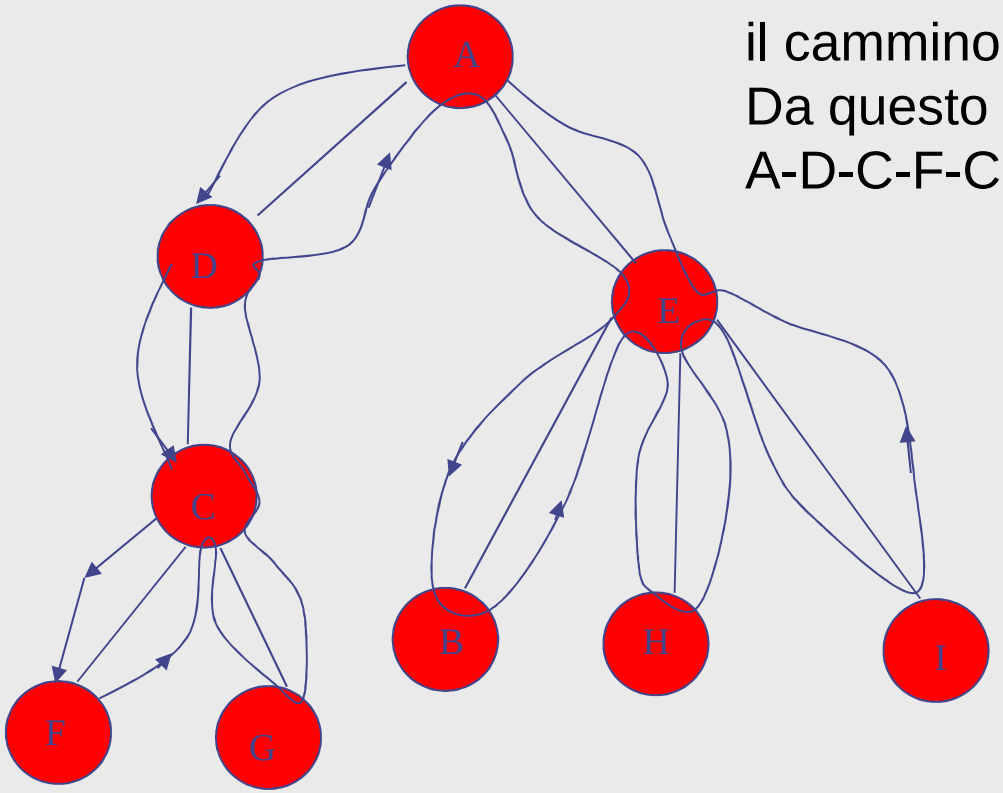
Osservazioni:

- Un albero di copertura del grafo contiene tutti i vertici del grafo e fornisce un cammino dal vertice di partenza al vertice stesso.
- Tale cammino non possiede la proprietà di essere semplice.
- Per ottenere un giro di costo non troppo elevato bisognerà scegliere archi “poco costosi”.
- Un albero di copertura minimo è formato dagli archi del grafo “meno costosi”.

Commesso viaggiatore

(con disuguaglianza triangolare)

Se il seguente è un albero di copertura,



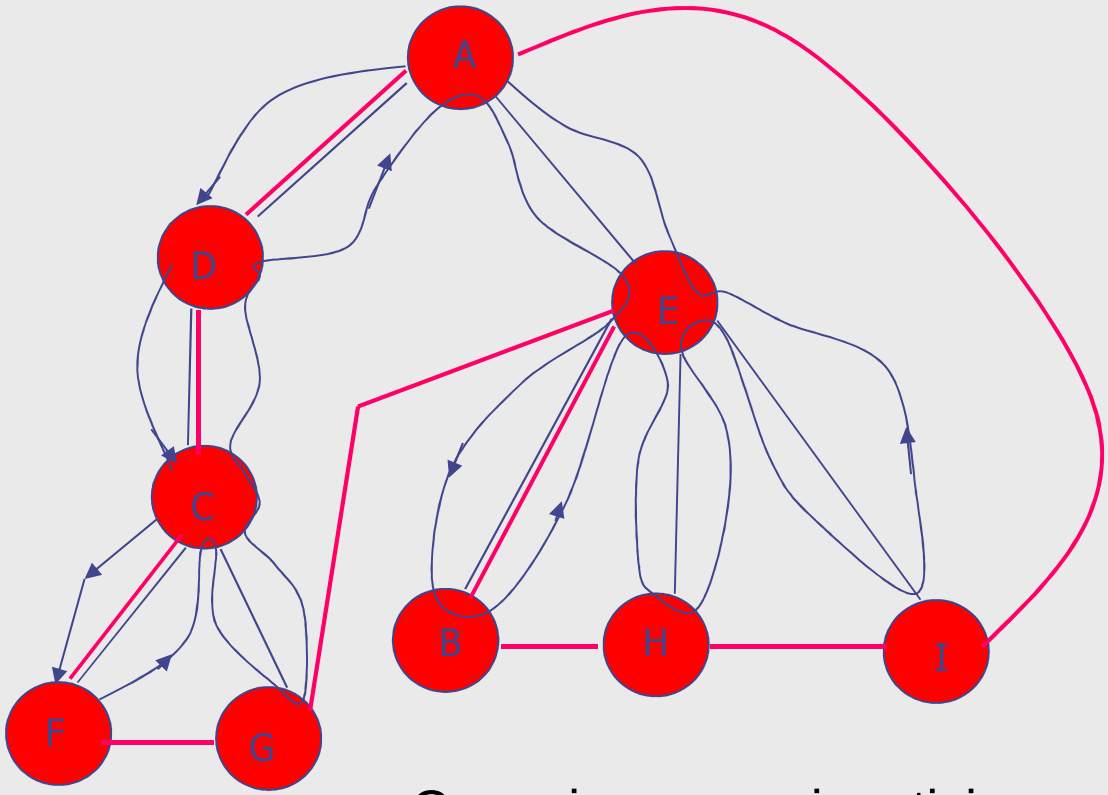
il cammino i(visita in profondità) individuato
Da questo albero è:
A-D-C-F-C-G-C-D-A-E-B-E-H-E-I-E-A

Cancellando dal cammino i vertici ripetuti si ottiene:
A-D-C-F-G-E-B-H-I.

Commesso viaggiatore

(con disuguaglianza triangolare)

Poiché il grafo è completo, gli archi tra due vertici esistono sempre; pertanto cancellando i vertici ripetuti introduciamo nuovi archi che sappiamo presenti nel grafo e, contemporaneamente, sostituiamo due o più attraversamenti di archi con uno soltanto.



Cammino senza i vertici ripetuti:
A-D-C-F-G-E-B-H-I.

Commesso viaggiatore

(con disuguaglianza triangolare)

APPROX_C_V (G,c)

1. Seleziona un vertice r $\in$ V e costruisci un minimo albero di copertura T per G utilizzando, per esempio, l'algoritmo di Prim.
2. Sia L la lista dei vertici ottenuta visitando T in ordine anticipato.
3. Restituisci il ciclo H dei vertici nell'ordine di L.

Il tempo di esecuzione di *APPROX_C_V* è dato dal tempo di esecuzione dell'algoritmo di Prim: $\mathbf{O}(E + V \lg V)$ sommato al tempo di visita dell'albero T: $\mathbf{O}(V)$

Poiché il grafo è completo, $E = V(V - 1)/2$, quindi la complessità dell'algoritmo di Prim, e dell'algoritmo approssimato, è $\mathbf{O}(V^2)$.

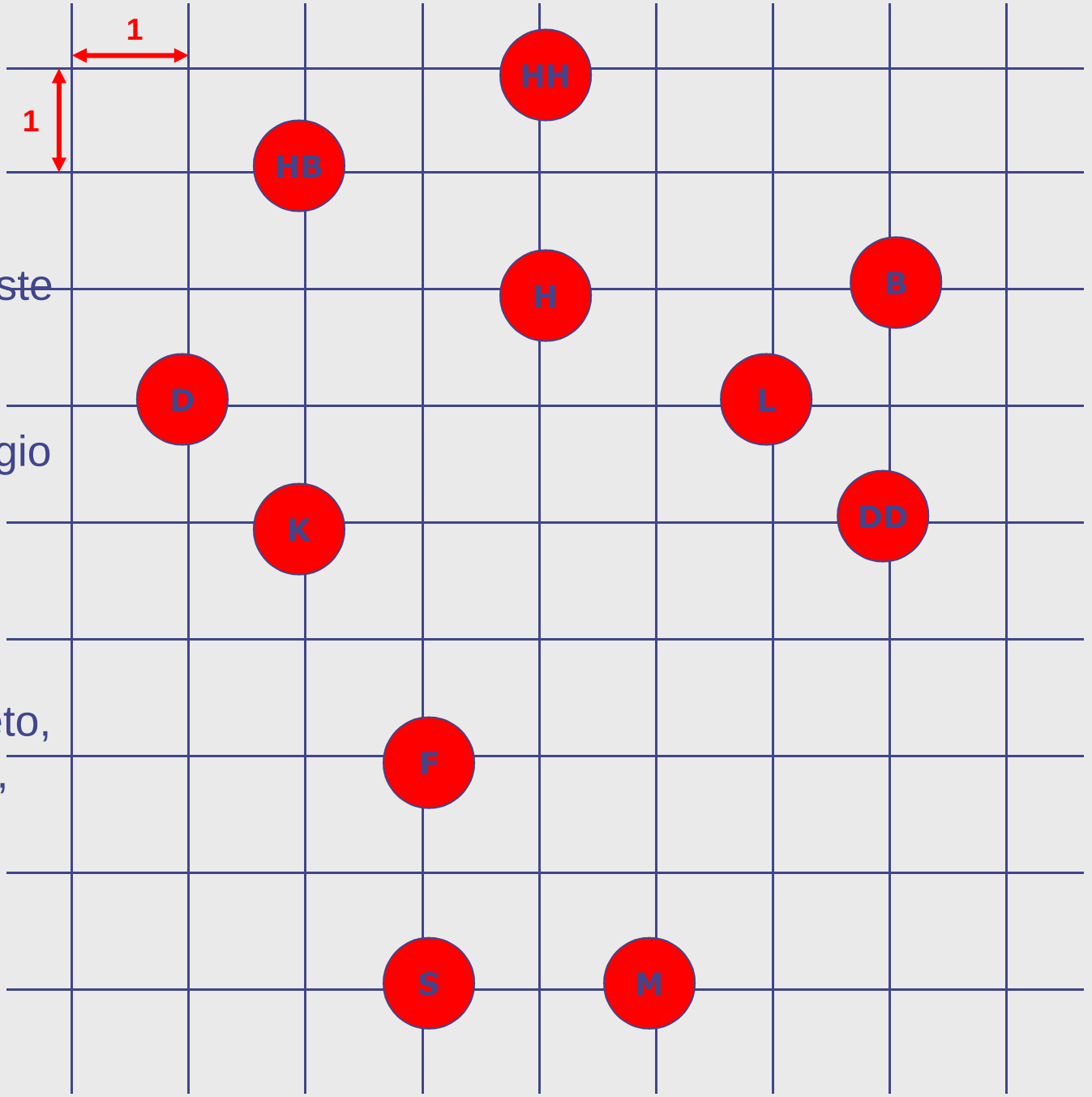
Algoritmo di PRIM

(minimum spanning tree)

```
procedure PRIM(grafo non orientato pesato G = (N, A));
    begin
        considera T formato da un nodo qualsiasi e nessun arco;
        while esistono nodi non in T adiacenti a un nodo in T do begin
            seleziona l'arco di peso minimo tra quelli che connettono un
                nodo in T e un nodo non in T;
            aggiungi a T sia l'arco selezionato che il nodo che non era in T;
        end;
        restituisci T come il minimo albero di copertura di G;
    end;
```

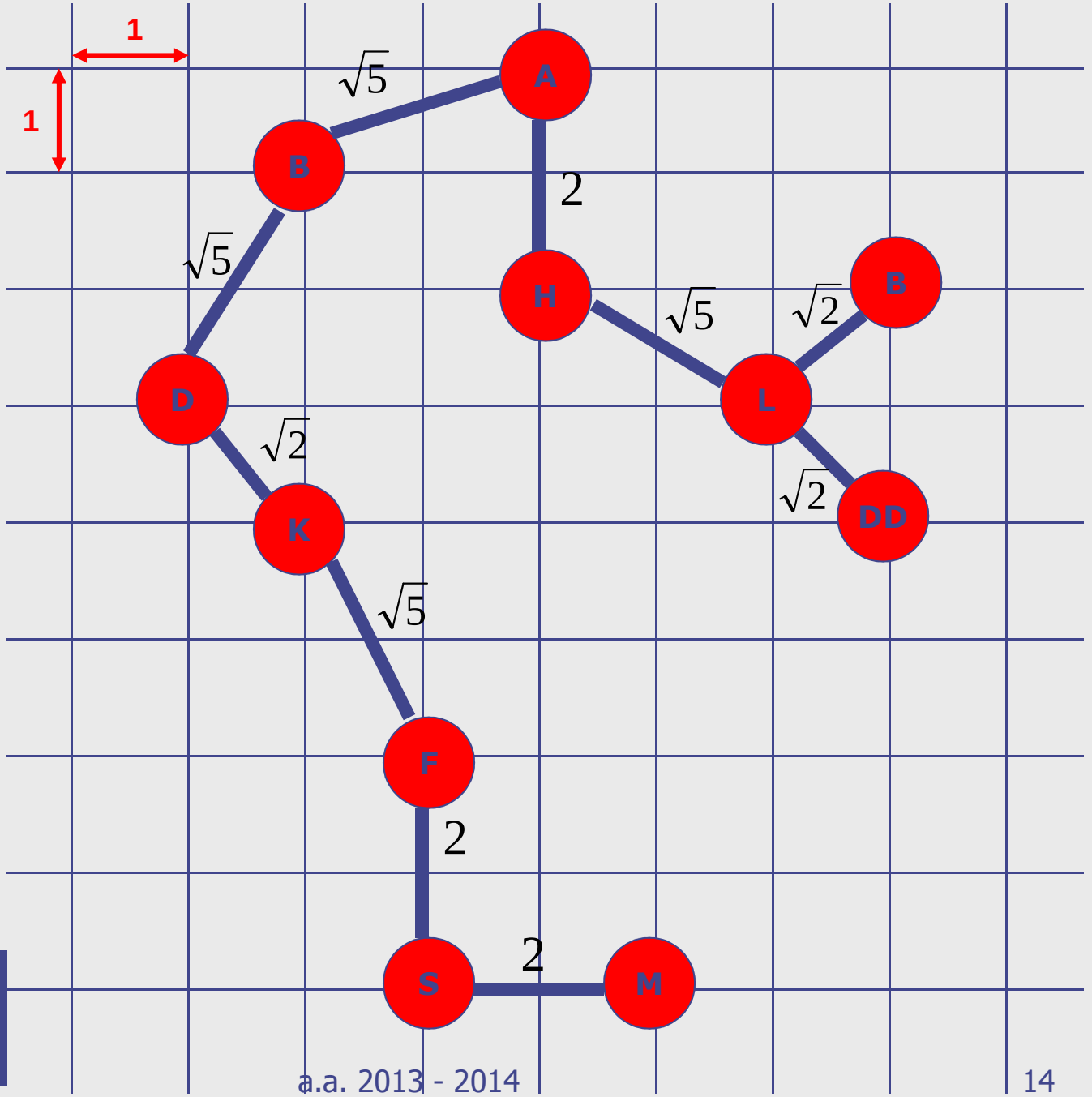
esempio

- 11 città disposte su griglia unitaria.
- Costo di viaggio tra le città: distanza euclidea tra i punti.
- Grafo completo, non orientato, pesato.



Passo 1:

Costruzione di un M.S.T. con l'algoritmo di Prim



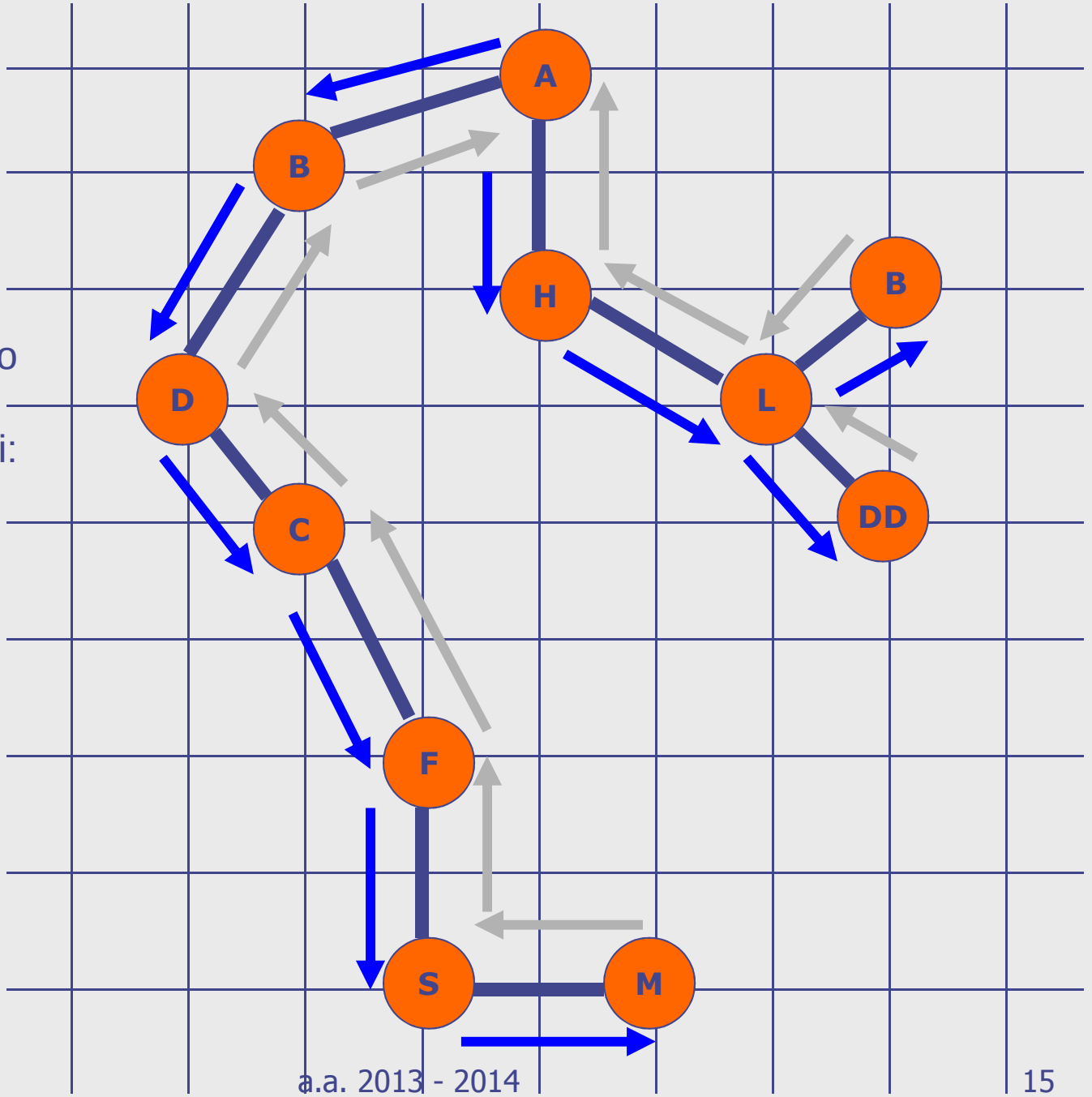
MST-Prim

Peso = 19,187

Passo 2:

visita in ordine anticipato con i vertici elencati solo la prima volta che vengono incontrati:

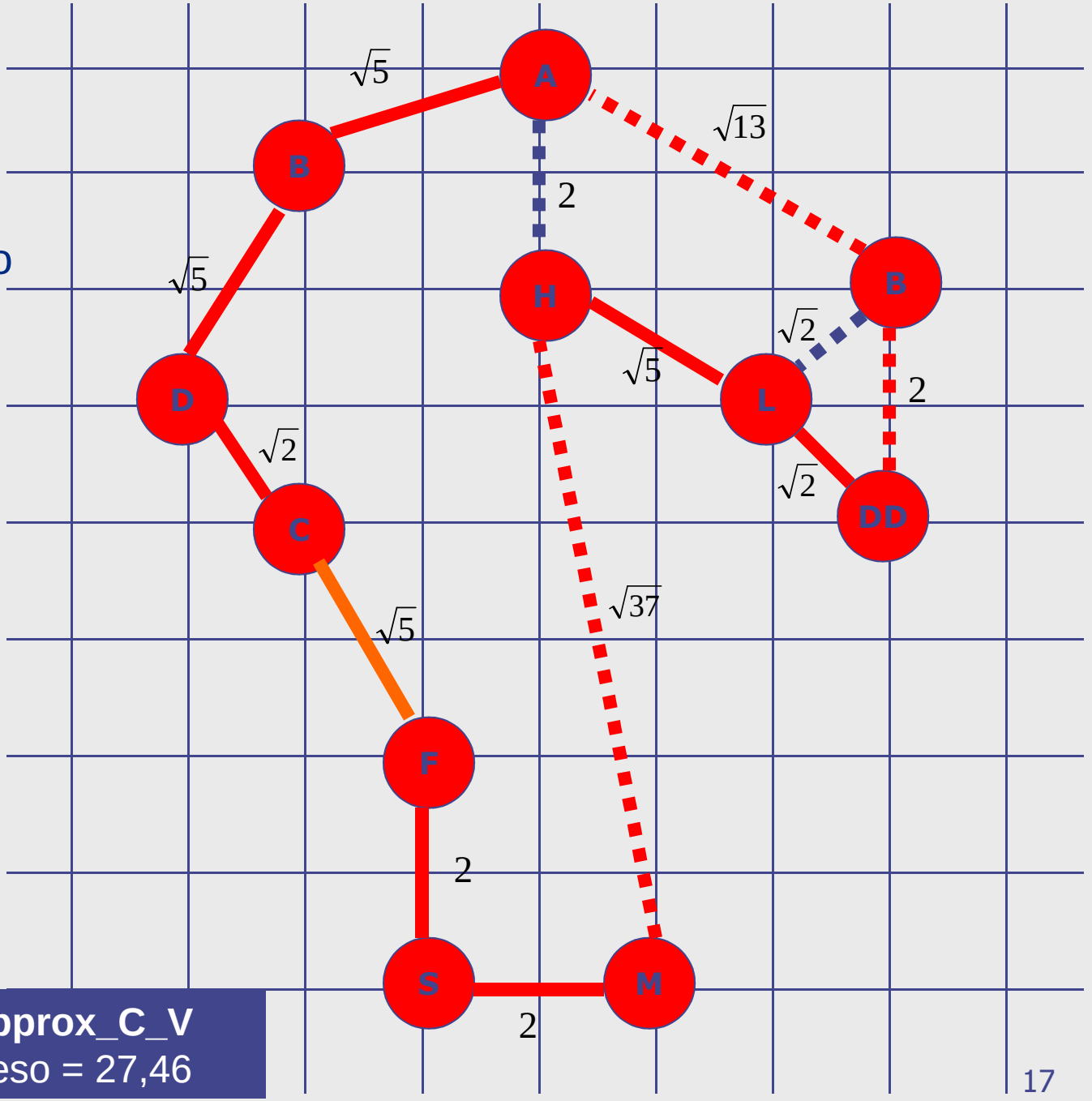
Amburgo
Brema
Düsseldorf
Colonia
Francoforte
Stoccarda
Monaco
Hannover
Lipsia
Dresda
Berlino



Passo 3:

Restituisci il ciclo H dei vertici nell'ordine di L.

Amburgo
Brema
Düsseldorf
Colonia
Francoforte
Stoccarda
Monaco
Hannover
Lipsia
Dresda
Berlino

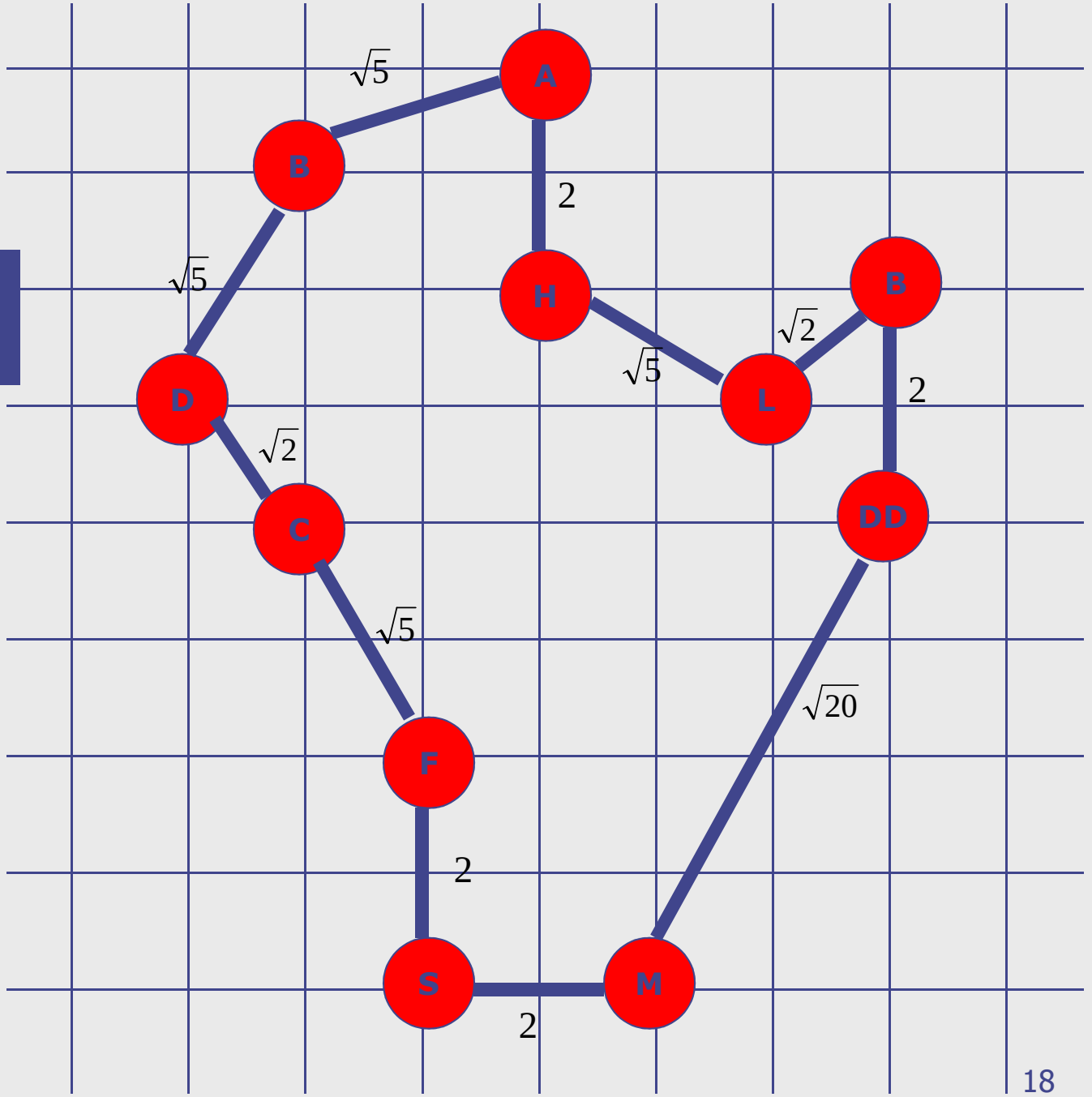


Approx_C_V
Peso = 27,46

Giro ottimo

Peso = 24,24

Amburgo
Brema
Düsseldorf
Colonia
Francoforte
Stoccarda
Monaco
Dresda
Berlino
Lipsia
Hannover



Commesso viaggiatore

(con disuguaglianza triangolare)

Teorema:

1) La lista dei vertici restituita da *Approx_C_V* è un ciclo hamiltoniano.
2) *Approx_C_V* ha rapporto limite uguale a 2.

Dimostrazione

1) Ovvio.
2) Siano H^* una soluzione ottima e H la soluzione calcolata da *Approx_C_V*. Dobbiamo dimostrare:

$$c(H) \leq 2c(H^*)$$

cioè che il costo del giro calcolato dall'algoritmo approssimato non è mai superiore al doppio del costo minimo di un giro.

Commesso viaggiatore

(con disuguaglianza triangolare)

- Cancellando un arco qualunque da *H** si ottiene un albero di copertura; se *T* è un MST, allora:

$$c(T) \leq c(H^*)$$

- Una visita completa di *T* in preordine elenca i vertici quando sono visitati. Indichiamo con α il cammino ottenuto dalla visita.

- Poiché la visita completa attraversa ogni arco di *T* esattamente 2 volte, si ha: $c(\alpha) = 2c(T)$

- Da cui segue: $c(\alpha) = 2c(T) \leq 2c(H^*)$
- α non è un giro perché alcuni vertici sono visitati più di una volta.

Commesso viaggiatore

(con disuguaglianza triangolare)

- Per la proprietà della disuguaglianza triangolare si può cancellare la visita di qualsiasi vertice di α senza aumentare il costo: se un vertice v viene cancellato sul percorso α tra i vertici u e t, poiché il grafo è completo, esiste l'arco (u, t), che può essere sostituito ai due archi (u, v) e (v, t) con la sicurezza che:

$$c(\alpha) - \big(c(u, v) + c(v, t) \big) + c(u, t) \leq c(\alpha)$$

- La soluzione H trovata dall'algoritmo ha perciò un costo

$$c(H) \leq c(\alpha) \leq 2\, c(H^*)$$

N.B. *Approx_C_V* fornisce una buona approssimazione. Vi sono però, per questo problema, algoritmi con prestazioni migliori.