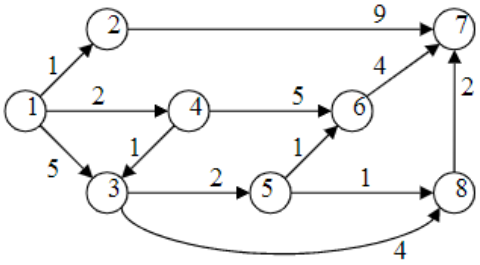
Simulazione Esame 2018/2019 – Esercizio 3 Grafi Raccolta "Esercizi vari"

3. Si vogliono determinare i cammini minimi composti da al più 4 archi sul seguente grafo:



- si scelga un algoritmo appropriato e si motivi la scelta;
- si calcolino i cammini minimi con al più quattro archi dal nodo 1 verso tutti gli altri nodi (i passi dell'algoritmo vanno riportati in una tabella e giustificati);
- si ricavi un cammino minimo di al più quattro archi da 1 a 7, descrivendo il procedimento adottato.

a) Nella scelta dell'algoritmo, notiamo che ci viene dato un massimo numero di archi da dover rispettare, pertanto si può applicare solo l'algoritmo di Bellman – Ford, l'unico che dà la possibilità di calcolo dei cammini minimi sulla base di un massimo numero di archi. Applicheremo Bellman – Ford fermandoci alla quarta iterazioni, con un numero di archi e iterazioni pari a $k \leq 4$

b) Ora, il calcolo dei cammini minimi, riportati in tabella (controllando ad ogni passo miglioramenti rispetto alla riga precedente e segnando in colonna apposita gli aggiornamenti effettuati; i predecessori sono segnati come semplici pedici senza parentesi):

Iterazione	Nodo 1	Nodo 2	Nodo 3	Nodo 4	Nodo 5	Nodo 6	Nodo 7	Nodo 8	Aggiornamenti
$Inizio$	$0_\wedge$	$+\infty_\wedge$	$+\infty_\wedge$	$+\infty_\wedge$	$+\infty_\wedge$	$+\infty_\wedge$	$+\infty_\wedge$	$+\infty_\wedge$	1
$h=1$	$0_\wedge$	1_1	5_1	2_1	$+\infty_\wedge$	$+\infty_\wedge$	$+\infty_\wedge$	$+\infty_\wedge$	2,3,4
$h=2$	$0_\wedge$	1_1	3_4	2_1	7_3	7_4	10_2	~~9_3~~	3,5,6,7,8
$h=3$	$0_\wedge$	1_1	3_4	2_1	5_3	7_4	~~10_2~~	7_3	5,8
$h=4$	$0_\wedge$	1_1	3_4	2_1	5_3	6_5	9_8	6_5	6,7,8

Le etichette di una riga sono ottenute controllando i vincoli duali su tutti gli archi uscenti dai nodi "aggiornati" della riga (iterazione) precedente secondo la $if\ \pi_j > \pi'_i + c_{ij}\ then\ \pi_j = \pi'_i + c_{ij}\ and\ p(j) = i$ dove (i,j) è uno degli archi uscenti da un nodo i aggiornato all'iterazione precedente, π_j è l'etichetta corrente (sulla riga corrente) del nodo j, π'_i è l'etichetta del nodo i all'iterazione (riga) precedente e c_{ij} è il costo dell'arco (i,j).

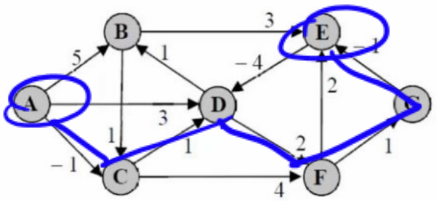
Grazie al fatto di utilizzare l'etichetta del nodo precedente, viene assicurata la scelta del cammino minimo con 4 archi (qua si può fare un qualsiasi esempio numerico dove si va a scegliere, prendendo ad esempio l'ultima iterazione, qui $h = 4$ un cammino che ha costo migliore considerando l'etichetta precedente e non quella corrente, qui $h = 5$, dimostrando la validità di quanto fatto).

c) Un cammino minimo con al massimo 4 archi da 1 a 7 viene fatta seguendo la catena dei predecessori, quindi, partendo dal nodo 7 (ultimo nodo in generale) con $h = 4$, si considera quindi:

$$7 \Leftarrow 8 \Leftarrow 3 \Leftarrow 4 \Leftarrow 1$$

Questo equivale al percorso $1 \Rightarrow 4 \Rightarrow 3 \Rightarrow 8 \Rightarrow 7$, con costo 9.

3. Nel seguente grafo, calcolare i cammini minimi dal nodo A verso **tutti** gli altri nodi.



CAMMINO

a. si scelga l'algoritmo da utilizzare e si motivi la scelta;
b. si applichi l'algoritmo scelto (riportare e **giustificare** i passi dell'algoritmo in una tabella);
c. si utilizzi la tabella del punto b per riportare, se possibile, l'albero e il grafo dei cammini minimi oppure, se esiste, un ciclo di costo negativo (descrivere il procedimento);
d. è possibile, con l'algoritmo scelto, ottenere un cammino minimo da A a E con al più 5 archi? Se sì, qual è? come si ottiene?

Punto a) Si sceglie l'agoritmo di Bellman-Ford in quanto esistono archi con costo negativo. Bellman-Ford è l'unico algoritmo visto in grado di garantire convergenza alla soluzione ottima del problema dei cammini minimi in presenza di archi di costo negativo, sebbene mediamente meno efficiente dell'algoritmo di Dijkstra, che non garantisce di trovare la soluzione al problema dei cammini minimi se esistono archi di costo negativo.

Punto b)

Iter.	A	B	C	D	E	F	G	*Aggiornati*
h=0	0(-)	+∞(-)	+∞(-)	+∞(-)	+∞(-)	+∞(-)	+∞(-)	A
h=1	0(-)	5(A)	−1(A)	3(A)	+∞(-)	+∞ (-)	+∞(-)	B, C, D
h=2	0(-)	4(D)	−1(A)	0(C)	8(B)	3(C)	+∞(-)	B, D, E, F
h=3	0(-)	1(D)	−1(A)	0(C)	~~7(B)~~ 5(F)	2(D)	4(F)	B, E, F, G
h=4	0(-)	1(D)	−1(A)	0(C)	~~4(B)~~ 3(G)	2(D)	3(F)	E, G
h=5	0(-)	1(D)	−1(A)	−1(E)	2(G)	2(D)	3(F)	D, E
h=6	0(-)	0(D)	−1(A)	−2(E)	2(G)	1(D)	3(F)	B, D, F
h=7	0(-)	−1(D)	−1(A)	−2(E)	2(G)	0(D)	2(F)	B, F, G

La tabella riporta al riga 0 di inizializzazione e una riga per ogni iterazione. All'iterazione *h* si controllano gli archi (*i*,*j*) uscenti da ciascun nodo *i* nella colonna *Aggiornati* alla riga *h*–1, e si aggiornano i costi e i predecessori del nodo *j* all'iterazione *h* qualora l'etichetta del nodo *i* all'iterazione *h*–1 più il costo dell'arco (*i*,*j*) sia strettamente minore dell'etichetta corrente del nodo *j*.

L'algoritmo si ferma qualora la lista dei nodi aggiornati sia vuota (convergenza delle etichette ai costi dei cammini minimi da A verso gli altri nodi) o, come in questo caso, venga completata l'iterazione con *h* uguale al numero di nodi avendo dei nodi aggiornati (presenza di un ciclo negativo).

Punto c)

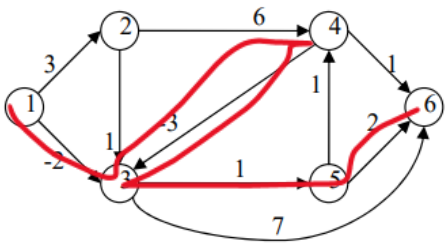
Siccome all'iterazione 7 (numero dei nodi) la lista dei nodi aggiornati non è vuota, le etichette non sono stabili ed esiste un ciclo negativo, pertanto non è possibile determinare albero o grafo dei cammini minimi. Per individuare un ciclo negativo, si considera l'iterazione *h*=7 e si parte da un nodo che è stato aggiornato, seguendo a ritroso la catena dei predecessori fino a incontrare due volte uno stesso nodo. Partendo ad esempio da B otteniamo B←D←E←G←F←D, che individua il ciclo D→F→G→E→D di costo -2.

Punto d)

Sì, l'algoritmo utilizzato permette di ottenere un cammimo minimo da A a E con al più 5 archi, trattandosi dell'algoritmo di Bellman-Ford che fornisce, al completamento dell'interazione *h* i cammini minimi dall'origine verso un qualsiasi nodo con al più *h* archi. Per individuare il cammino, si parte dall'etichetta del nodo E all'iterazione 5 e si considera il predecessore G; si procede quindi con il predecessore di G all'iterazione 4 (F), con il predecessore di F all'iterazione 3 e così via, considerando di volta in volta i predecessori all'iterazione precedente (vedi elementi evidenziati) e ottenendo A→C→D→F→G→E il cui costo è 2, come si può verificare.

Esercizi sui grafi presenti in Raccolta “Esercizi vari”

- Si consideri il seguente grafo:



CICLO NEGATIVO = RITORNO AD UN NODO GIÀ ESPLORATO!

a. si scelga un algoritmo per determinare i cammini minimi dal nodo 1 verso tutti gli altri nodi e si motivi la scelta;
b. si applichi l’algoritmo scelto (riportare e **giustificare** i passi dell’algoritmo in una tabella);
c. L’algoritmo ha individuato un ciclo negativo? Giustificare la risposta.
d. Riportare l’albero e il grafo dei cammini minimi, oppure il ciclo negativo (in ogni caso, si descriva il procedimento utilizzato).

a) In questo grafo, utilizziamo Bellman-Ford, in quanto esistono archi con costi ridotti negativi.

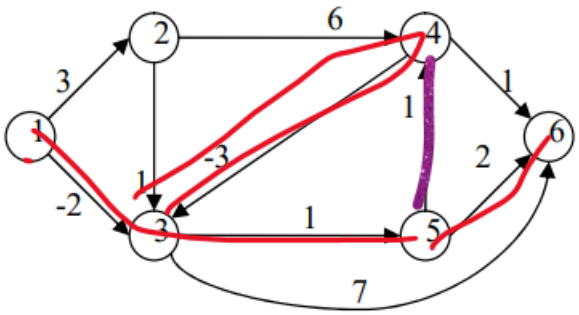
b) Ora, il calcolo dei cammini minimi, riportati in tabella (controllando ad ogni passo miglioramenti rispetto alla riga precedente e segnando in colonna apposita gli aggiornamenti effettuati; i predecessori sono segnati come semplici pedici senza parentesi). Dato che non siamo limitati dai max-hop, andremo a creare la tabella iterando un numero pari di volte al numero di nodi (quindi, facendoli tutti):

Iterazione	Nodo 1	Nodo 2	Nodo 3	Nodo 4	Nodo 5	Nodo 6	Aggiornamenti
$Inizio$	$0_\wedge$	$+\infty_\wedge$	$+\infty_\wedge$	$+\infty_\wedge$	$+\infty_\wedge$	$+\infty_\wedge$	1
$h=1$	$0_\wedge$	3_1	-2_1	$+\infty_\wedge$	$+\infty_\wedge$	$+\infty_\wedge$	2, 3
$h=2$	$0_\wedge$	3_1	-2_1	9_2	-1_3	5_3	4,5,6
$h=3$	$0_\wedge$	3_1	-2_1	0_5	-1_3	1_5	4, 6
$h=4$	$0_\wedge$	3_1	-3_4	0_5	-1_3	1_5	3
$h=5$	$0_\wedge$	3_1	-3_4	0_5	-1_3	1_5	//
$h=6$	$0_\wedge$	3_1	-3_4	0_5	-1_3	1_5	//

c) L’algoritmo ha individuato un ciclo negativo, dato che seguire la catena dei predecessori (l’ultima volta in cui i singoli nodi sono stati aggiornati guardando il loro predecessore) ritorna in un arco precedente.

d) Per individuare l’albero dei cammini minimi, si percorre a ritroso la catena dei predecessori partendo dall’ultima iterazione come detto,

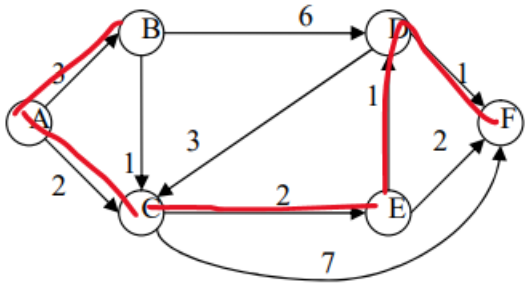
Per individuare il grafo dei cammini minimi, composto da *tutti* i cammini minimi, esamino il grafo e scrivendo le etichette ottime, verifico se esistono altri cammini minimi. In questo caso, albero e grafo non coincidono, dato che esiste altra etichetta di costo minimo (costo <= a quella attualmente presente).



Legenda:

- Rosso – Albero
- Arcobaleno - Grafo

- Si consideri il seguente grafo:



a. si scelga il miglior algoritmo tra quelli presentati per determinare i cammini minimi dal nodo A verso tutti gli altri nodi e si motivi la scelta;
b. si applichi l'algoritmo scelto (riportare e **giustificare** i passi dell'algoritmo in una tabella);
c. si disegni l'albero e il grafo dei cammini minimi, descrivendo il procedimento usato.

a) Il miglior algoritmo in questo contesto è Dijkstra, in quanto più efficiente e usabile in quanto tutti i costi ridotti sono positivi.

b) Ora, il calcolo dei cammini minimi, riportati in tabella. Si ricorda che:

- $\hat{v}$ rappresenta l'etichetta minima di ogni iterazione
- $\bar{S}$ rappresentano le etichette ancora da fissare
- Il segno * rappresenta l'etichetta fissata
- Il segno – rappresenta l'etichetta controllata ma non aggiornata
- Il segno x rappresenta l'etichetta non controllata perché il nodo è già fissato
- Gli spazi vuoti servono per indicare che non considero più l'etichetta nelle varie iterazioni in quanto fissata
- L'algoritmo termina quando non ci sono più nodi in $\bar{S}$

Iterazione	Nodo A	Nodo B	Nodo C	Nodo D	Nodo E	Nodo F	$\bar{S}$	$\hat{v}$
$Inizio$	$0_{\wedge}$	$+\infty_{\wedge}$	$+\infty_{\wedge}$	$+\infty_{\wedge}$	$+\infty_{\wedge}$	$+\infty_{\wedge}$	A,B,C,D,E,F	
$h=1$	*	3_A	2_A	$+\infty_{\wedge}$	$+\infty_{\wedge}$	$+\infty_{\wedge}$	B,C,D,E,F	A
$h=2$		-	*	9_B	4_C	9_C	B,D,E,F	C
$h=3$		*		5_E	-	6_E	D,E,F	B
$h=4$				-	*	-	D,F	E
$h=5$				*		-	F	D
$h=6$						*	$\emptyset$	F

Ad ogni iterazione, percorriamo il grafo e scegliamo il percorso con costo minore. Ad ogni iterazioni, scegliamo e fissiamo un'etichetta che ha costo minore, scremando ad ogni iterazione quelle da controllare e avere sempre in mano il costo minimo. Anche in questo caso, come per gli algoritmi

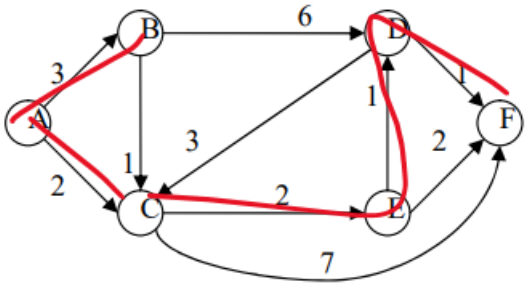
label correcting in caso di convergenza alla soluzione ottima, le etichette calcolate e i relativi puntatori rappresentano, rispettivamente, una soluzione duale ammissibile e i predecessori su dei cammini dall'origine ai diversi nodi. Questo funziona non avendo costi negativi.

c)

Pertanto, come abbiamo visto per gli algoritmi label correcting in caso di convergenza, è possibile derivare l'albero (risp. il grafo) dei cammini minimi attraverso i puntatori ai predecessori (risp. la verifica della saturazione dei vincoli duali sugli archi).

Concretamente, avremo che:

- In rosso riportiamo l'albero dei cammini minimi
- In arcobaleno riportiamo il grafo (composto come sempre da *tutti* i cammini minimi, quindi fissando le etichette ottime e scegliendo come cammino sia l'albero che tutte le altre etichette con costo $\leq$). In questo caso non viene disegnato, essendo che albero e grafo coincidono



ALBERO = 1 CAMMINO MINIMO (TUTTI I NODI)

GRAFO = TUTTI I CAMMINI MINIMI