

Scenario

- Archi di un grafo controllati da **agenti egoistici**
- Solo l'agente conosce il peso associato al proprio arco
- Obiettivo: calcolare una "buona" soluzione di un certo **problema di ottimizzazione** rispetto a pesi reali
- Strumento: progettazione di un **meccanismo truthful** (**pagamento** opportuno degli agenti per convincerli a dire la verità!)

Il problema del cammino minimo
tra 2 nodi in un grafo con archi
privati

Comprare un cammino minimo in una rete

F : insieme dei cammini fra s e t

Meccanismo

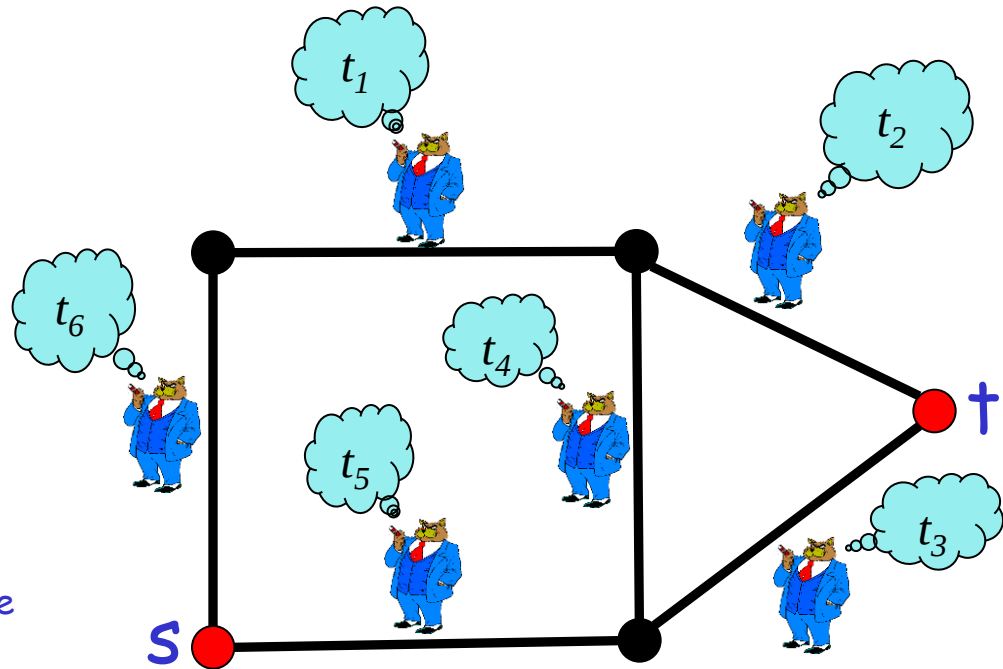
decide il cammino e i pagamenti

$f(t)$:
un cammino minimo
rispetto ai tipi
degli agenti

t_e : costo utilizzo arco e

se arco e è selezionato
e riceve un pagamento di p_e
utilità di e :

$$p_e - t_e$$



Il problema dello *shortest path egoistico*

- **Input:** un grafo $G=(V,E)$, ogni arco è un agente egoistico, un nodo **sorgente** s e un nodo **destinazione** t ; il **tipo** di un agente è il costo di utilizzo dell'arco (quindi **tipo** > 0); la sua **valutazione** è uguale al suo **tipo**;
- **SCF:** un vero cammino minimo in $G=(V,E, \text{tipi})$ tra s e t .

Più Formalmente

- Soluzioni ammissibili:
 - F : insieme dei cammini in G da s a t
- Tipo dell'agente e :
 - τ_e : peso dell'arco
 - intuitivamente: τ_e è il costo che l'agente sostiene per utilizzare e
- Valutazione agente e di un cammino $P \in F$:
 - $v_e(\tau_e, P) = \tau_e$ se $e \in P$, 0 altrimenti
- SCF: shortest path in $G=(V, E, \tau)$ fra s e t .

Come progettare un meccanismo truthful per il problema?

Osservazione cruciale:

la (vera) lunghezza di un cammino P è:

$$\sum_{e \in P} \tau_e = \sum_{e \in E} v_e(\tau_e, P)$$

problema **utilitario**!

...usiamo i meccanismi VCG

Meccanismo VCG

- $M = \langle g(r), p(r) \rangle$:

- $g(r) := x^* = \arg \min_{x \in F} \sum_j v_j(r_j, x)$

- $p_e(r)$: Per ogni arco $e \in E$:

$$p_e(r) = \sum_{j \neq e} v_j(r_j, g(r_{-e})) - \sum_{j \neq e} v_j(r_j, x^*) \text{ cioè}$$

Meccanismo VCG

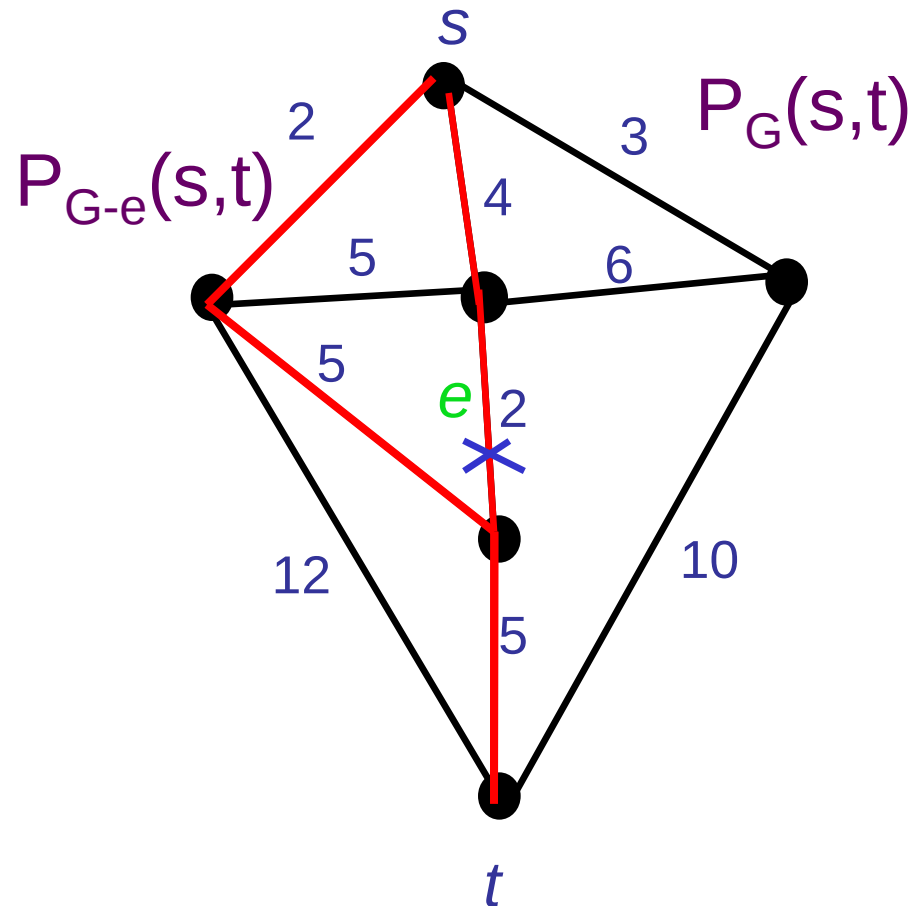
- $M_{SP} = \langle g(r), p(r) \rangle$:
 - $g(r)$: calcola un cammino minimo $P_G(s,t)$ in $G=(V,E,r)$
 - p_e : Per ogni arco $e \in E$:

$$p_e(r) = \sum_{j \neq e} v_j(r_j, g(r_{-e})) - \sum_{j \neq e} v_j(r_j, P_G(s,t)) \text{ cioè}$$

$$p_e(r) = \begin{cases} d_{G-e}(s,t) - (d_G(s,t) - r_e) & \text{se } e \in P_G(s,t) \\ 0 & \text{altrimenti} \end{cases}$$

⇒ Per ogni $e \in P_G(s,t)$, dobbiamo calcolare $P_{G-e}(s,t)$, ovvero il **cammino minimo di rimpiazzo** in $G-e = (V, E \setminus \{e\}, r_{-e})$ tra s e t

Cammino di rimpiazzo per e



quanto viene
pagato e ?

$$p_e = 12 - (11 - 2) = 3$$

Quel è la complessità temporale del meccanismo?

...dobbiamo calcolare con uno SP tra s e t

...e il pagamento per gli archi selezionati

...è solo una questione algoritmica!

Ipotesi di lavoro

- $n=|V|$, $m=|E|$
- $d_G(s,t)$: **distanza** in G da s a t (somma dei pesi degli archi di $P_G(s,t)$)
- I nodi s,t sono **2-edge connessi**: cioè, esistono in G almeno 2 cammini tra s e t che sono disgiunti sugli archi $\Rightarrow$ per ogni arco e del cammino $P_G(s,t)$ che viene rimosso esiste almeno un cammino alternativo in $G-e$

...infatti, in caso contrario...

- Se s, t non sono 2-edge connessi, c'è almeno un arco in $P_G(s, t)$ che è un **ponte** (arco che rimosso spezza G in due componenti C_1 e C_2 , $s \in C_1$ e $t \in C_2$)
 - Se e è un ponte $\rightarrow d_{G-e}(s, t) = \infty$
- $\Rightarrow$ Il possessore di quell'arco "tiene in pugno" il sistema: può chiedere qualsiasi cifra!

Una soluzione banale

$\forall e \in P_G(s,t)$ applichiamo l'algoritmo di Dijkstra al grafo $G-e$

Complessità: $k=O(n)$ archi per $O(m + n \log n)$:
tempo $O(mn + n^2 \log n)$

Si può dimostrare il seguente:

Teorema

M_{SP} è calcolabile in tempo $O(m + n \log n)$.

Malik, Mitta, Gutpa, the k most vital arcs in the shortest path problem, 1989

Esercizio

Progettare un meccanismo veritiero per il problema del calcolo del cammino minimo tra due nodi nello scenario in cui un agente controlla **più di un arco** del grafo. Come nel caso del singolo arco, i pesi degli archi controllati dall'**agente** i sono privati e la valutazione di un agente rispetto a una soluzione (cammino) P è uguale alla somma dei (veri) pesi degli archi selezionati nel cammino.

Meccanismi one-parameter

Archer & Tardos, Truthful mechanisms for one-parameter agents, FOCS'01

Tecniche note

Un problema è **utilitario** quando:

$$f(t) = \arg \min_{o \in F} \sum_i v_i(t_i, o)$$

Meccanismi VCG

Validi per problemi utilitari (es., MST e SP)

$$g(r) = \arg \min_{o \in F} \sum_i v_i(r_i, o)$$

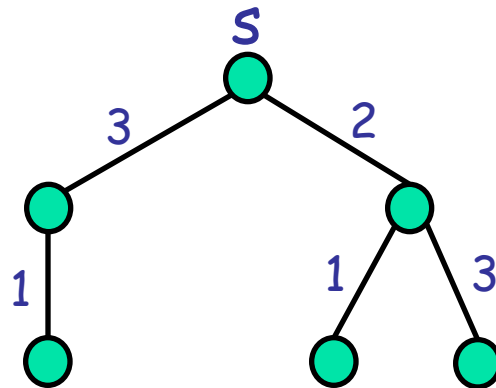
$$p_i(r) = \sum_{j \neq i} v_j(r_j, g(r_{-i})) - \sum_{j \neq i} v_j(r_j, g(r))$$

Shortest Path Tree (SPT) non cooperativo

- **Problema:** broadcasting
 - una sorgente s vuole spedire un messaggio ai nodi $V \setminus \{s\}$
- **Informazione posseduta dagli agenti:**
tempo di attraversamento dei link
- **Obiettivo:** minimizzare il tempo di consegna di ogni messaggio

Formulazione

- F : insieme alberi ricoprenti V (radicati in s)
- Per ogni $T \in F$
 - $f(t) = \arg \min_{T \in F} \sum_{v \in V} d_T(s, v) = \arg \min_{T \in F} \sum_{e \in E(T)} t_e ||e||$
 - $||e||$ è la molteplicità dell'arco e , intesa come numero di cammini ai quali appartiene (in T)



Formulazione

- F : insieme alberi ricoprenti V (radicati in s)
- Per ogni $T \in F$
 - $f(t) = \arg \min_{T \in F} \sum_{v \in V} d_T(s, v) = \arg \min_{T \in F} \sum_{e \in E(T)} t_e ||e||$
 - $||e||$ è la molteplicità dell'arco e , intesa come numero di cammini ai quali appartiene (in T)

il problema è utilitario?

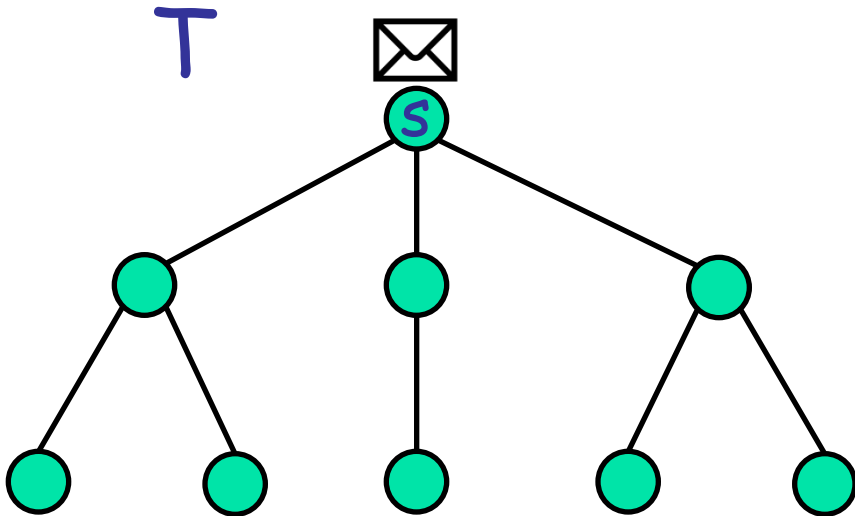
...dipende dalla funzione di valutazione dei giocatori

come è la funzione di valutazione?

...dipende da come è "usato" l'albero T

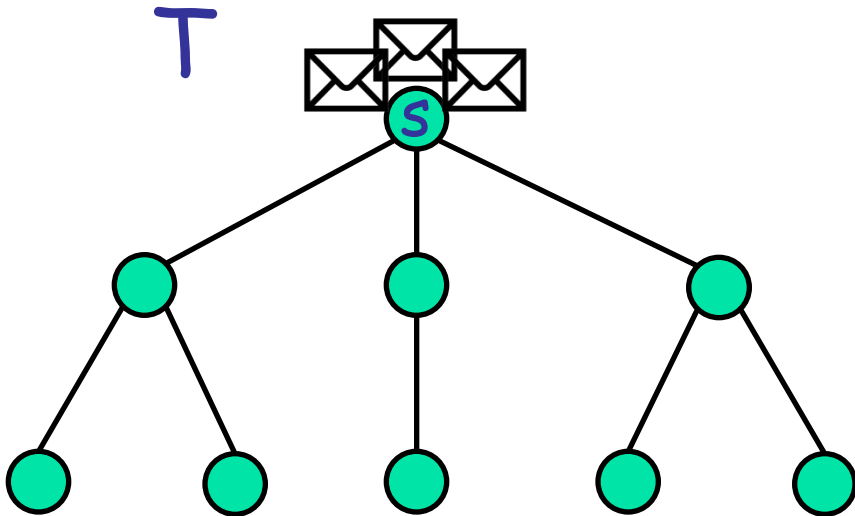
Protocollo multicast:

una sola copia del
messaggio viene
spedita
(eventualmente
duplicata nei nodi)



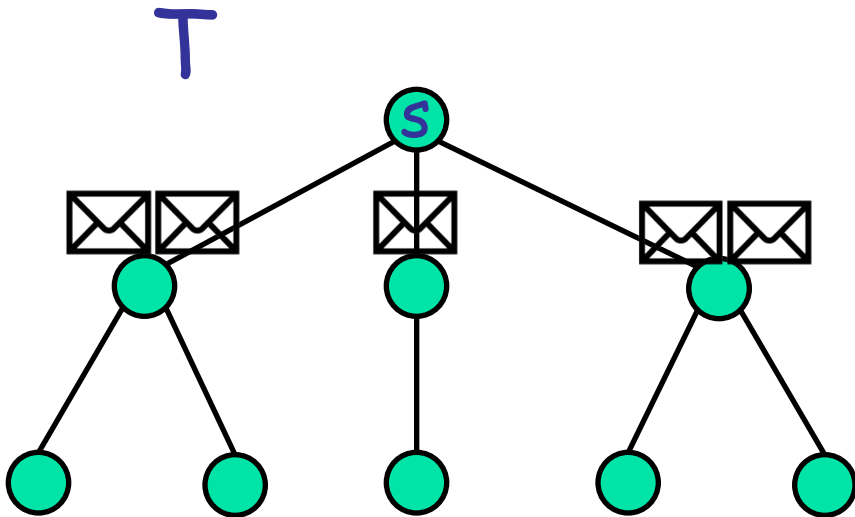
Protocollo multicast:

una sola copia del
messaggio viene
spedita
(eventualmente
duplicata nei nodi)



Protocollo multicast:

una sola copia del
messaggio viene
spedita
(eventualmente
duplicata nei nodi)



valutazione del giocatore rispetto a **T**:

$$v_e(t_e, \mathbf{T}) = t_e \text{ se } e \in E(\mathbf{T}), 0 \text{ altrimenti}$$

$$\Rightarrow f(\mathbf{t}) \neq \arg \min_{T \in F} \sum_{e \in E(T)} v_e(t_e, \mathbf{T}) \quad \text{problema non utilitario!}$$

Come tratto i problemi non utilitari?

...per problemi one-parameter uso i meccanismi
one-parameter (OP)

Un problema è **one-parameter** se

1. L'informazione posseduta da ogni agente a_i è un **singolo parametro**
 $t_i \in \mathbb{R}$

2. La valutazione di a_i ha la forma
$$v_i(t_i, o) = t_i w_i(o),$$

$w_i(o)$: **carico di lavoro** per a_i in o

SPT non cooperativo (ogni agente controlla un arco)

- F : insieme alberi ricoprenti V (radicati in s)
- Per ogni $T \in F$

- $$f(t) = \arg \min_{T \in F} \sum_{v \in V} d_T(s, v) = \sum_{e \in E(T)} t_e ||e||$$

- $$v_e(t_e, T) = \begin{cases} t_e & \text{se } e \in E(T) \\ 0 & \text{altrimenti} \end{cases}$$

Multicast: caso non utilitario

- $$v_e(t_e, T) = t_e w_e(T) \quad w_e(T) = \begin{cases} 1 & \text{se } e \in E(T) \\ 0 & \text{altrimenti} \end{cases}$$

VCG vs OP

- Meccanismi VCG: valutazioni (costi) e tipi arbitrari ma problemi utilitari
- Meccanismi OP: funzione di scelta sociale arbitraria ma tipi a singolo-parametro e valutazioni vincolate
- Se un problema è utilitario e one-parameter → meccanismo (esatto) VCG e OP coincidono

Una proprietà interessante

Definizione

Un algoritmo $g()$ per un problema OP di minimizzazione è **monotono** se

$\forall$ agente a_i , $w_i(g(r_{-i}, r_i))$ è **non crescente** rispetto a r_i , per tutti gli $r_{-i} = (r_1, \dots, r_{i-1}, r_{i+1}, \dots, r_N)$

Notazione

Scriveremo $w_i(r)$ al posto di $w_i(g(r))$

Teorema 1 (Mayerson '81)

Condizione necessaria affinché un meccanismo $M=\langle g(r), p(r) \rangle$ per un problema OP sia veritiero è che $g(r)$ sia monotono.

Dim (per assurdo)

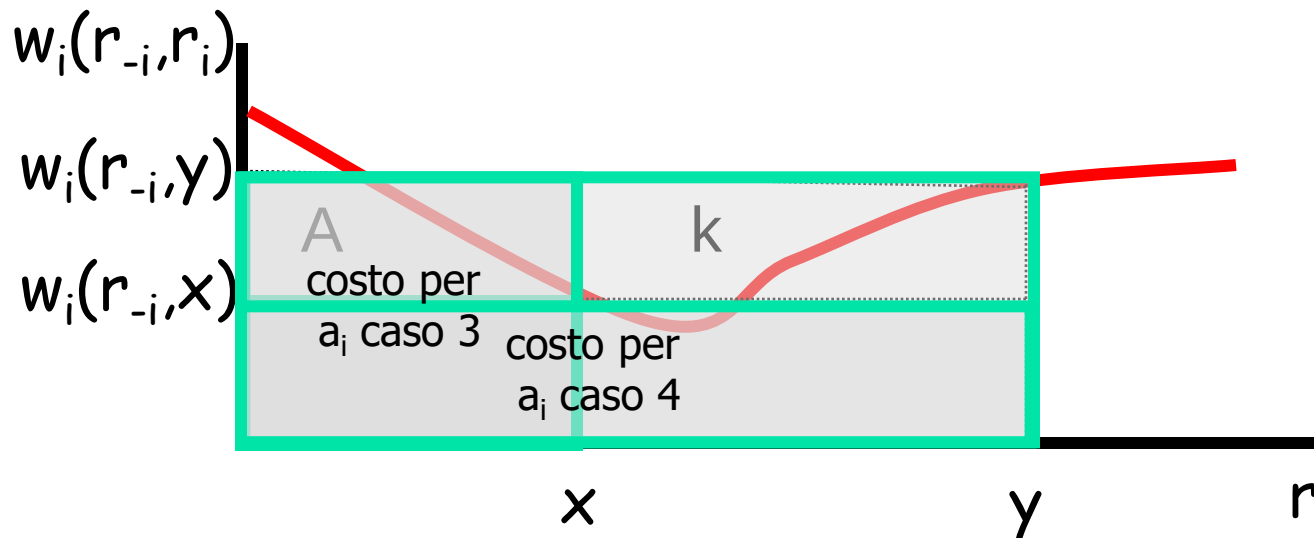
Supponiamo $g()$ non monotono, e...

...facciamo vedere che nessuno schema di pagamento può rendere M veritiero

Se $g(\cdot)$ è non monotono esiste un agente a_i e un vettore r_{-i} tale che $w_i(r_{-i}, r_i)$ è non "non crescente"...

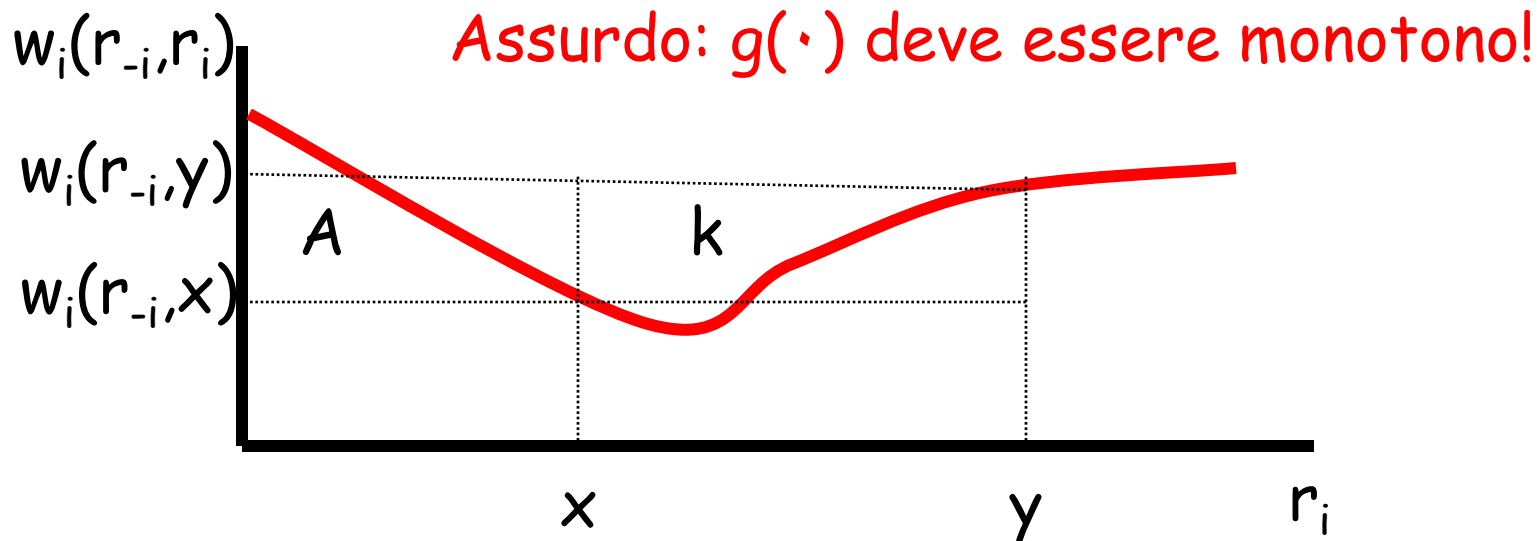
Dim (continua)

1. Se $t_i = x$ e $r_i = t_i \rightarrow v_i(t_i, 0) = x w_i(r_{-i}, x)$
2. Se $t_i = y$ e $r_i = t_i \rightarrow v_i(t_i, 0) = y w_i(r_{-i}, y)$
3. Se $t_i = x$ e $r_i = y \rightarrow a_i$ aumenta il suo costo di A
4. Se $t_i = y$ e $r_i = x \rightarrow a_i$ ha un risparmio di $A+k$



Dim (continua)

- Sia $\Delta p = p_i(r_{-i}, y) - p_i(r_{-i}, x)$
 - Se M è truthful deve essere:
 - $\Delta p \leq A$ (altrimenti quando $t_i = x$, a_i dichiara y , in quanto in tal caso il suo costo aumenta di A , e quindi se $\Delta p > A$, la sua utilità aumenta!)
 - $\Delta p \geq A + k$ (altrimenti quando $t_i = y$, a_i dichiara x , in quanto in tal caso il suo costo diminuisce di $A + k$, e quindi se $\Delta p < A + k$, ciò significa che il decremento nel pagamento è minore del decremento del costo, ovvero la sua utilità aumenta!)
- ... ma k è strettamente positivo!



Meccanismi one-parameter (OP)

- $g(r)$: qualsiasi algoritmo monotono
- $p_i(r) = h_i(r_{-i}) + r_i w_i(r) - \int_0^{r_i} w_i(r_{-i}, z) dz$

$h_i(r_{-i})$: funzione arbitraria indipendente da r_i

Teorema 2 (Mayerson '81) Un meccanismo OP
(per un problema OP) è veritiero.

Dim: Facciamo vedere che l'utilità di un
agente a_i può solo decrescere se a_i mente

Siano r_{-i} le dichiarazioni degli altri agenti

Il pagamento fornito ad a_i (quando dichiara r_i) è:

$$p_i(r) = \underbrace{h_i(r_{-i})}_{\text{Ininfluyente perché independiente da } r_i} + r_i w_i(r) - \int_0^{r_i} w_i(r_{-i}, z) dz$$

Ininfluyente perché
independente da r_i

pongo
 $h_i(r_{-i})=0$

Dim (continua)

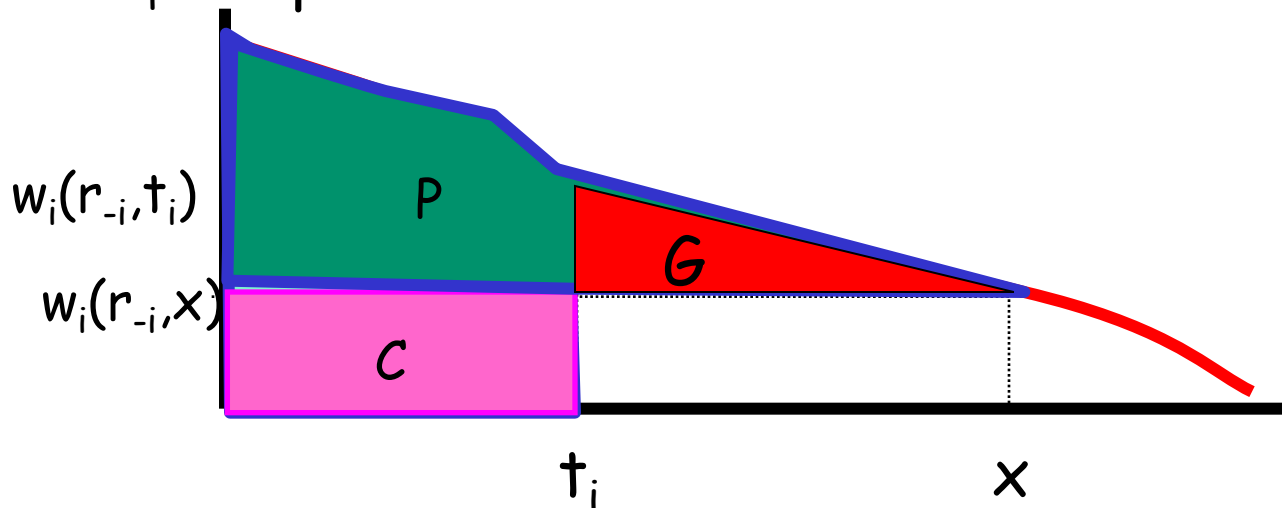
- $u_i(t_i, (r_{-i}, t_i)) = p_i(g(r_{-i}, t_i)) - v_i(t_i, g(r_{-i}, t_i)) =$
 $t_i w_i(g(r_{-i}, t_i)) - \int_0^{t_i} w_i(r_{-i}, z) dz - t_i w_i(g(r_{-i}, t_i)) = - \int_0^{t_i} w_i(r_{-i}, z) dz$

- Se a_i dichiara $x > t_i$:

- La valutazione diventa: $C = t_i w_i(r_{-i}, x)$

- il pagamento diventa: $P = x w_i(r_{-i}, x) - \int_0^x w_i(r_{-i}, z) dz$

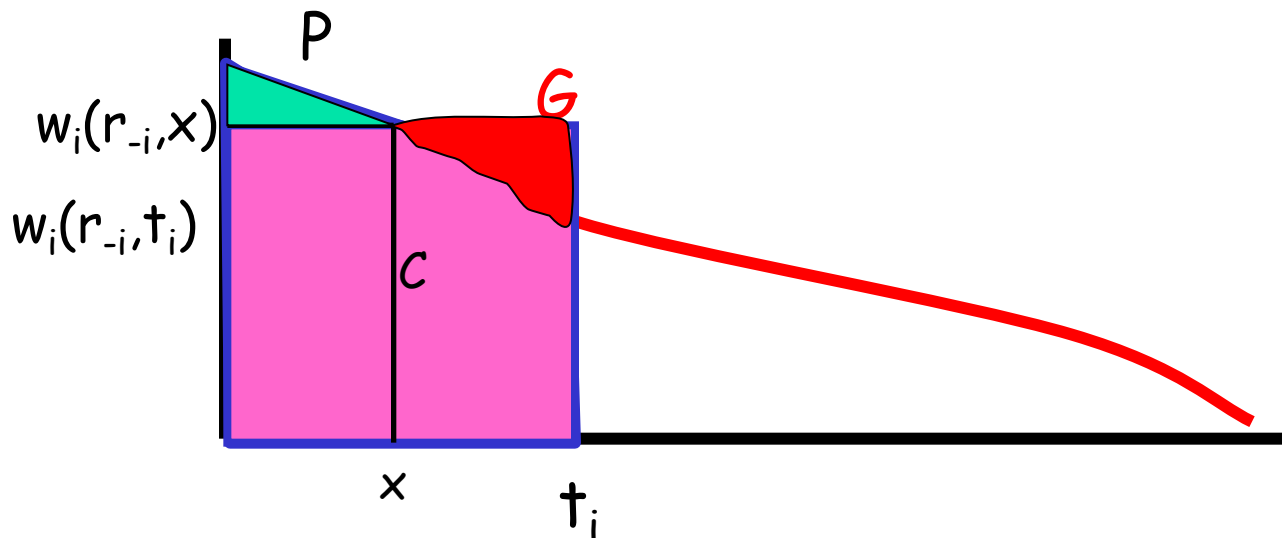
$\Rightarrow a_i$ sta perdendo G



Dim (continua)

- $u_i(t_i, (r_{-i}, t_i)) = - \int_0^{t_i} w_i(r_{-i}, z) dz$
- Se a_i dichiara $x < t_i$
 - La valutazione diventa C
 - il pagamento diventa P $\Rightarrow a_i$ sta perdendo G

a_i non ha convenienza a mentire!



Sulla funzione $h_i(r_{-i})$

Un meccanismo garantisce la **volontaria partecipazione (VP)** se l'utilità di un qualsiasi agente (che dichiara il vero) ha sempre un utile non negativo

Ma il pagamento di a_i quando dichiara r_i è:

$$p_i(r) = h_i(r_{-i}) + r_i w_i(r) - \int_0^{r_i} w_i(r_{-i}, z) dz$$

Se scegliamo la costante $h_i(r_{-i}) = \int_0^{\infty} w_i(r_{-i}, z) dz$,

il pagamento diventa: $p_i(r) = r_i w_i(r) + \int_{r_i}^{\infty} w_i(r_{-i}, z) dz$

$\Rightarrow$ L'utilità di un agente che dichiara il vero diventa:

$$u_i(t_i, r) = \int_{t_i}^{\infty} w_i(r_{-i}, z) dz \geq 0.$$

Meccanismi

one-parameter: il problema
dell'albero dei cammini minimi a
sorgente singola

SPT non cooperativo (ogni agente controlla un arco)

- **Input:** un grafo $G=(V,E)$ biconnesso sugli archi, in cui ogni arco corrisponde in modo biunivoco ad un insieme di agenti egoisti, ed un nodo sorgente $s \in V$; il **tipo** di un agente è il costo di utilizzo dell'arco (quindi **tipo** > 0);
- **SCF:** un albero dei cammini minimi radicato in s in $G=(V,E,t)$.

SPT non utilitario

- Per ogni albero ricoprente T di G radicato in s , la funzione obiettivo minimizzata dalla SCF è:

$$f(t) = \arg \min_{T \in \mathcal{F}} \left\{ \sum_{v \in V} d_T(s, v) = \sum_{e \in E(T)} t_e ||e|| \right\}$$

- Con protocollo *multicast* : $v_e(t_e, T) = \begin{cases} t_e & \text{se } e \in E(T) \\ 0 & \text{altrimenti} \end{cases}$

e quindi $f(t) \neq \arg \min_{T \in \mathcal{F}} \sum_{e \in E} v_e(t_e, T)$ (problema **non utilitario**)

- Ma il problema è **one-parameter**, in quanto

$$v_e(t_e, T) = t_e w_e(T), \text{ ove } w_e(T) = \begin{cases} 1 & \text{se } e \in E(T) \\ 0 & \text{altrimenti} \end{cases}$$

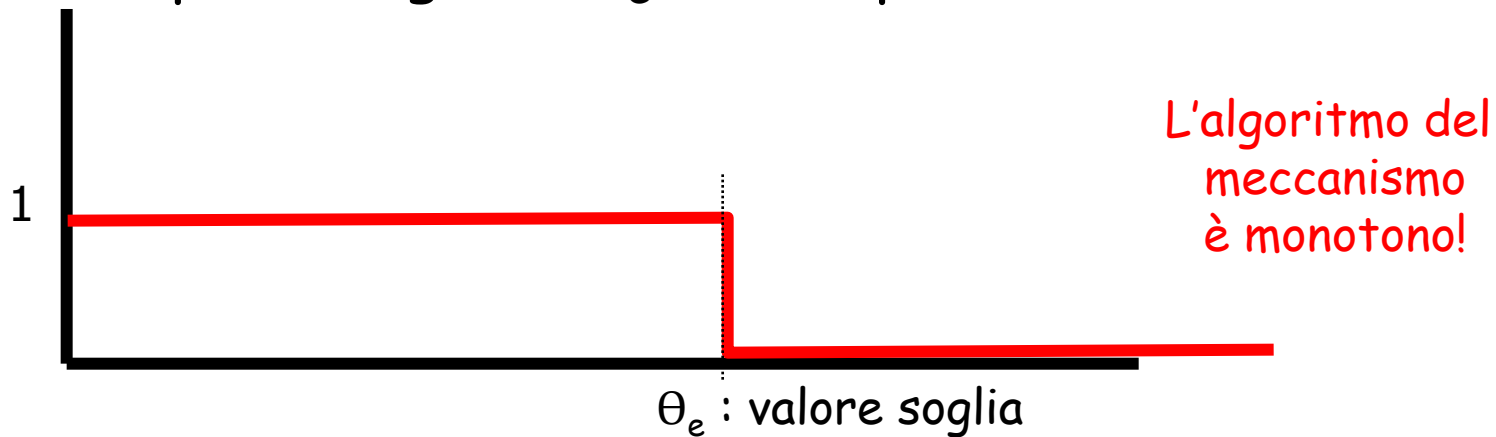
Meccanismo one-parameter per l'SPT non utilitario

- $M_{SPT} = \langle g(r), p(r) \rangle$
 - $g(r)$: dato il grafo e le dichiarazioni, calcola un SPT $S_G(s)$ di $G=(V,E,r)$ utilizzando l'algoritmo di Dijkstra.
 - $p(r)$: per ogni arco $e \in E$
$$p_e(r) = r_e w_e(r) + \int_{r_e}^{\infty} w_e(r_{-e}, z) dz$$

così da garantire la **partecipazione volontaria**.

Truthfulness

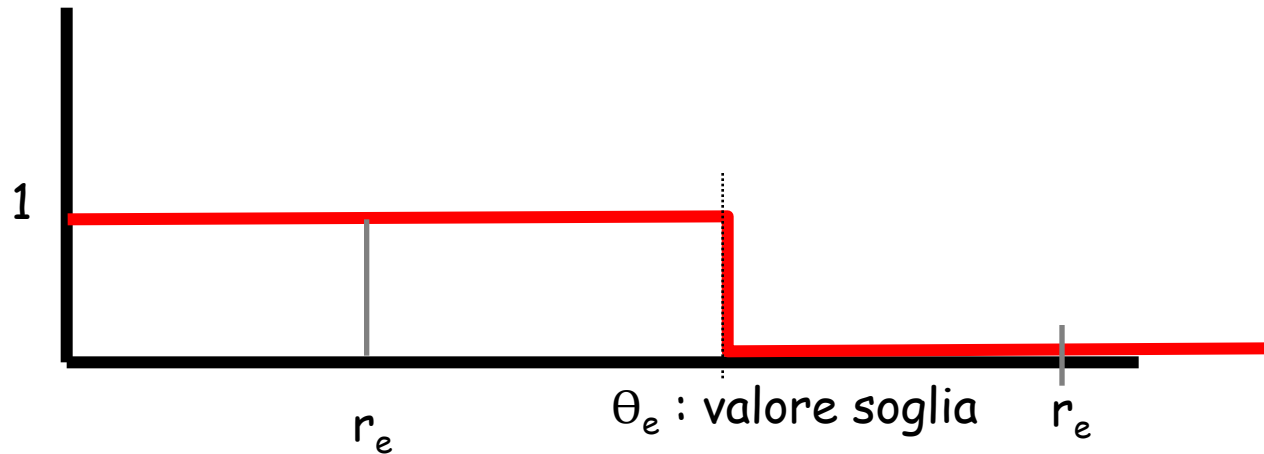
Osservazione: M_{SPT} è truthful. La truthfulness segue dal fatto che M_{SPT} è un meccanismo OP. Infatti, l'algoritmo di Dijkstra per il calcolo dell'SPT è **monotono**, in quanto il carico di lavoro per un agente a_e ha sempre la forma:



θ_e è il valore tale che, fissato r_{-e} :
se a_e dichiara al più θ_e , allora **e** è selezionato
se a_e dichiara più di θ_e , allora **e** non è selezionato



Sui pagamenti



- $p_e = 0$, se e non è un arco selezionato

$$p_e = r_e w_e(r) + \int_{r_e}^{\infty} w_e(r_{-e}, z) dz = 0 + 0 = 0$$

- $p_e = \theta_e$, se e è nella soluzione

$$p_e = r_e w_e(r) + \int_{r_e}^{\infty} w_e(r_{-e}, z) dz = r_e + \theta_e - r_e = \theta_e$$

Un caso speciale dei problemi OP

Un problema è **binary demand (BD)** se

1. L'informazione posseduta da ogni agente a_i è un **singolo parametro** $t_i \in \mathbb{R}$
2. La valutazione di a_i ha la forma
$$v_i(t_i, o) = t_i w_i(o),$$

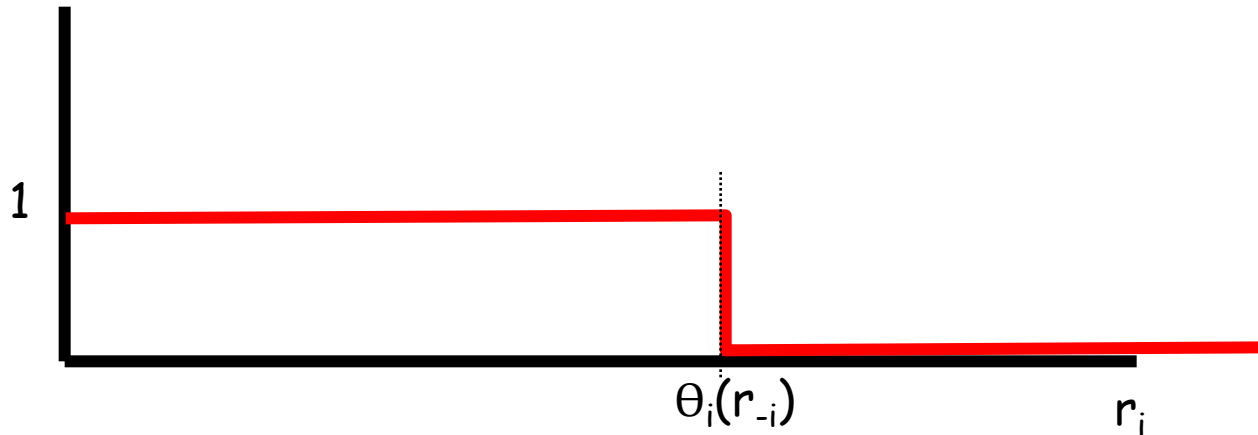
$w_i(o) \in \{0, 1\}$ **carico di lavoro** per a_i in o

quando $w_i(o) = 1$ diremo che a_i è
selezionato in o

Definizione

Un algoritmo $g()$ per un problema BD di minimizzazione è **monotono** se

$\forall$ agente a_i , e per tutti gli $r_{-i}=(r_1,\dots,r_{i-1},r_{i+1},\dots,r_N)$, $w_i(g(r_{-i},r_i))$ è della forma:



$\theta_i(r_{-i}) \in \mathbb{R} \cup \{+\infty\}$: **valore soglia**

il **pagamento** per a_i diventa:

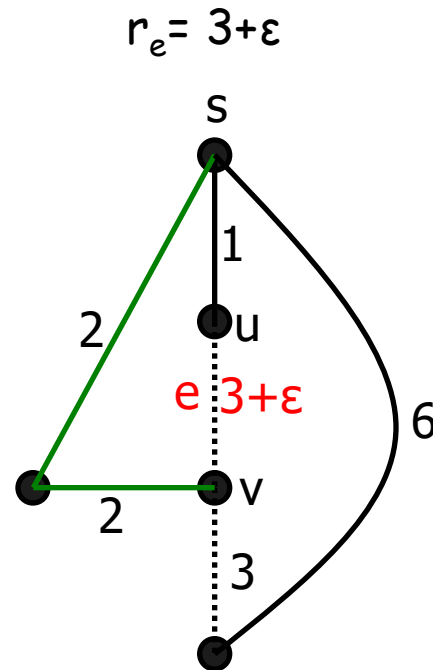
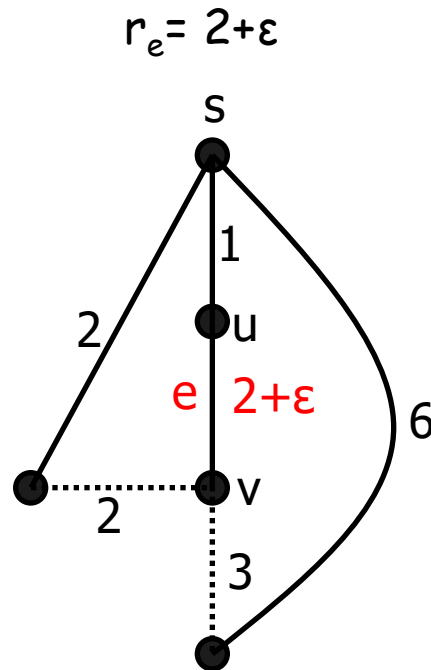
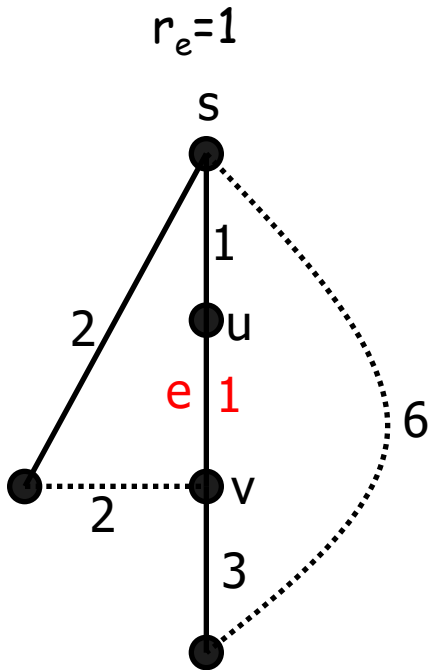
$$p_i(r) = \theta_i(r_{-i})$$

Sulle soglie

Sia $e=(u,v)$ un arco in $S_G(s)$ (u più vicino a s che v)

- e resta in $S_G(s)$ finché uso e per raggiungere v
- Allora, $\theta_e = d_{G-e}(s, v) - d_G(s, u)$

Esempio



$$\theta_e = 3$$

Una soluzione banale

$\forall e=(u,v) \in S_G(s)$ applichiamo l'algoritmo di Dijkstra al grafo $G-e$ e troviamo $d_{G-e}(s,v)$

Complessità: $k=n-1$ archi per $O(m + n \log n)$: $O(mn + n^2 \log n)$ time

Si può dimostrare il seguente teorema:

Teorema

M_{SPT} è calcolabile in tempo $O(m + n \log n)$.

Esercizio

Si consideri lo scenario della single-item auction (versione di minimizzazione) in cui un singolo job deve essere allocato a una delle possibili macchine, e dove ogni macchina è un giocatore che conosce privatamente il proprio costo a cui è possibile eseguire il job.

Se l'obiettivo del sistema è quello di allocare il job alla macchina che è in grado di eseguire il job al costo più basso, allora il meccanismo VCG (second-price mechanism) è truthful.

E se il sistema volesse allocare il job alla seconda migliore macchina? È possibile progettare un meccanismo truthful per questo problema?

Esercizio

Pensare allo scenario in cui un agente controlla **più di un arco** del grafo e si vuole progettare un meccanismo veritiero per calcolare un SPT.

Come nel caso del singolo arco, i pesi degli archi controllati dall'**agente i** sono privati e la valutazione di un agente rispetto a una soluzione (albero) **T** è uguale alla somma dei (veri) pesi degli archi selezionati nel cammino.

Possiamo usare i meccanismi **VCG**?

E quelli **OP**?