Index (parte 3)

- **search e advertising**:
  - search combinata ad advertising: mercato lucrativo
  - interessi utenti specificati nella query
  - advertising basato su keywords e query
  - genera mercato da ricerca utente
  - genera quasi tutto profitto Google
  - ads basati su keywords mostrati di fianco a risultati ricerca
  - molti risultati a pagamento per singolo termine query: motore di ricerca ha venduto ad a molti inserzionisti per query
  - slot alti più costosi
  - pay per click: inserzionisti pagano per click su loro ad
  - cliccare su ad è intenzione più forte di effettuare query
  - come settare prezzo per click:
    - soluzione 1: pubblicare prezzi come nei markets
      - inammissibile, troppe keyword e combinazioni di keywords
    - soluzione 2: offerte da parte degli inserzionisti
      - soluzione adottata
      - 2 implementazioni:
        - matching markets
        - aste

- **sponsored search** (advertising come matching market):
  - insieme di slots per posizionare ads per ciascuna query
  - slot numerati $1, 2, \ldots$ partendo dall'alto della pagina
  - r_i: frequenza di click slot i (n. click per ora)
  - v_j: profitto per click inserzionista j (profitto per click su ad)
  - $r_1 \geq r_2 \geq \ldots$: frequenza click sono non-crescenti
  - $r_i \cdot v_j$: guadagno inserzionista j per essere mostrato in slot i
  - assunzioni semplificanti:
    1. inserzionisti conoscono frequenze click
    2. frequenza click dipende solo da slot, non da qualità ad mostrato
    3. frequenza click non dipende da ads in altri slots
    4. profitto per click dipende da inserzionista e non dipende da pagina dove utente ha cliccato sull'ad
    5. numero slot $=$ numero inserzionisti:
       - se slots $>$ inserzionisti: aggiungiamo inserzionisti con $v_j = 0$
       - se inserzionisti $>$ slots: aggiungiamo slots con $r_i = 0$
  - allocazione slot a inserzionisti modellata come **matching market**
  - **matching market**:
    - insieme buyers (inserzionisti) e insieme sellers (slots)
    - ciascun buyer j ha valutazione v_{ij} per item offerto da seller i
    - **goal**: abbinare opportunamente buyers e sellers
    - quindi:
      - sellers $=$ slots
      - buyers $=$ inserzionisti
      - valutazioni $v_{ij} = r_i \cdot v_j$
    - primo esempio di interazione strutturata in forma di rete tra agenti

- principi base:
  1. agenti hanno preferenze diverse per tipi diversi di merce
  2. i prezzi possono decentralizzare allocazione merce ad agenti
  3. i prezzi possono portare ad allocazioni che sono socialmente ottime
- il miglior matching è ottenuto assegnando:
  - il primo slot (quello con frequenza di click maggiore) al primo inserzionista (quello con profitto per click maggiore)
  - il secondo slot al secondo inserzionista
  - ...
- **qualità dell'ad**:
  - abbiamo assunto nelle assunzioni che semplificano che le frequenze di click r_i dipendono solo dallo slot e non dalla qualità dell'ad mostrato
  - ma in realtà l'anteprima dell'ad è importante: se gli utenti non si fidano della compagnia non cliccano
  - il motore è pagato per click non per ad
  - scenario brutto:
    - gli inserzionisti di bassa qualità fanno offerte alte e prendono i primi slots
    - gli utenti non cliccano sui loro ad perchè non si fidano di loro
    - il motore di ricerca perde soldi
  - introduzione del **fattore di qualità**:
    - fattore di qualità q_j per l'inserzionista j
    - se l'inserzionista j appare nello slot i, allora la frequenza di click dello slot i è $q_j \cdot r_i$
    - le valutazioni quindi sono: $v_{ij} = q_j \cdot r_i \cdot v_j$
    - come si computano le qualità?
      - dipendono dalla frequenza di click sull'ad, dalla rilevanza del testo dell'ad e dalla rilevanza della landing page dell'ad
      - i motori di ricerca non condividono questi dettagli agli inserzionisti
  - con le qualità computate dai motori di ricerca senza condividere dettagli, il motore di ricerca ha potere illimitato nei confronti dell'ordinamento degli inserzionisti
  - altri problemi:
    - il market non sorge soltanto da una query ma ci sono market simultanei per keywords diverse sul web:
      - inserzionista dovrebbe dividere budget tra differenti markets, ma non si sa come farlo bene
    - inserzionisti devono comunicare i loro interessi in specifiche keywords ma è possibile che nessun inserzionista crei frasi di keywords complesse:
      - motore di ricerca può mostrare ads solo su frasi esplicitamente specificate, quindi inserzionista e motore di ricerca perdono soldi
    - quali ad mostrare è difficile:
      - cattiva idea considerare la valutazione massima per ciascuna keyword
    - quando devono pagare gli inserzionisti rilevanti per click dato che loro non hanno specificato interesse per una data query:
      - accordi tra motori di ricerca e inserzionisti:
        - i motori di ricerca estrapolano prezzi basandosi sui prezzi che gli inserzionisti vogliono pagare per e query che hanno specificato, ma non si sa come farlo bene
- **matching markets**: informazioni finali
  - possono modellare in maniera adatta la sponsored search
  - sono abbastanza generali da incorporare raffinamenti nella valutazione dei buyers (quality factors)
  - problemi:

        - motori di ricerca non conoscono le valutazioni degli inserzionisti
        - i motori di ricerca possono chiedere ma gli inserzionisti possono barare per avere migliori payoffs
            - dichiarando valutazioni più basse
        - soluzioni: le **aste**

- **scenario 1: assegnamento camere**
    - assegnamento stanze a studenti
    - ciascuna stanza associata a singolo studente
    - studenti hanno preferenze dievrse per le stanze
    - grafo **bipartito**, l'assegnamento è un **matching perfetto**
    - **valutazioni**: numeri che esprimono preferenze di agente verso un oggetto
    - qualità assegnamento $=$ somma delle valutazioni di quello che l'agente prende, per ogni agente
    - **assegnamento ottimo**: un assegnamento che massimizza la felicità totale di tutti
    - **qualità assegnamento**: somma delle valutazioni dell'assegnamento ottimo
    - prezzi **market-clearing**:
        - nello scenario 1 c'è un'amministrazione centrale che determina un matching perfetto (assegnamento ottimo)
        - ma in un mercato ci sono individui che fanno scelte libere basate sui prezzi e sulle valutazioni
        - utilizzo dei prezzi per decentralizzare il market

- **recap**:
    - grafi bipartiti:
        - nodi divisi in 2 categorie
        - gli archi connettono i nodi di una categoria all'altra categoria
    - matching perfetto:
        - scelta di archi nel grafo bipartito tale che:
            - ciascun nodo è endpoint di esattamente 1 arco tra quelli scelti (matching senza nodi isolati)
    - constricted set:
        - insieme di nodi tale che i loro archi restringono la formazione di un perfect matching
        - implica la non esistenza di un perfect matching
        - se grafo bipartito G ha un constricted set, allora non ammette un matching perfetto
        - se G non ha un perfect matching allora G ha un constricted set $\rightarrow$ teorema Hall's Matching
    - teorema Hall's Matching: un grafo bipartito G non ha un matching perfetto $\iff$ ha un constricted set
        - dimostrazione:
            - $\Leftarrow$: ha constricted set, non ha matching perfetto:
                - se G ha constricted set dalla sua definizione non ha un matching perfetto
            - $\Rightarrow$ non ha matching perfetto, ha constricted set:
                - se G non ha matching perfetto come identifichiamo un contricted set
                1. iniziamo da qualsiasi matching non perfetto M
                2. cerchiamo di ingrandire
                3. IF success:
                    - switchiamo al matching ingrandito
                4. ELSE:
                    - identifichiamo un constricted set
                - archi matching: archi usati in M
                - archi non matching: gli altri archi
                - cammino alternante: cammino che alterna archi matching e non matching

    - cammino aumentante: cammino alternante i cui endpoints sono nodi unmatched (non contenuti nel matching): M può essere ingrandito
    - dobbiamo cercare un cammino aumentante partendo da un matching non perfetto M
    - usiamo alternating BFS:
        - inizio da qualsiasi nodo unmatched W a destra (layer 0)
        - esploriamo grafo layer su layer usando archi non-matching a step dispari e archi matching a step pari
        - ad ogni step aggiungiamo nodi al layer se ci sono archi corrispondenti
        - dopo aver eseguito alternating BFS, cerchiamo un cammino alternante
        - torniamo al grafo originale e swappiamo archi matching e non matching al cammino aumentante trovato
    - albero BFS:
        - layer pari nodi lato destro grafo (buyers)
        - layer dispari nodi lato sinistro grafo (sellers)
        - se c'è nodo unmatched Z in un layer dispari, allora il cammino da W a Z in albero BFS è aumentante e quindi matching è ingrandibile
    - se nell'albero BFS non c'è alcun nodo unmatched in un layer dispari (solo W è nodo unmatched):
        - numero nodi in ogni layer dispari $=$ numero nodi in ogni layer pari $+1$
        - non contando W nel layer 0, numero nodi in ogni layer dispari $=$ numero nodi in ogni layer pari
        - ciascun nodo in un layer pari ha tutti i suoi neighbors che occorrono nei layers dispari
            - - i neighbors matched nel layer precedente - gli altri nel layer successivo
        - **constricted set**: W e tutti i nodi nei layers pari

- **computare matching perfetto**:
    1. iniziamo da matching vuoto
    2. cerchiamo unmatched node W sulla destra
    3. usiamo alternating BFS per cercare cammino aumentante che inizia da W
    4. SE trovato: usiamo il cammino aumentante ed iteriamo
    5. Altrimenti: indichiamo il constricted set

- **computare matching massimo**: se non c'è cammino aumentante partendo da uno specifico unmatched node W
    - sulla destra non significa che il matching è massimo:
        - se non si cono cammini aumentanti partendo da ciascun unmatched node W sulla destra, il matching corrente ha grandezza massima
    - per trovarlo, dobbiamo modificare alternating BFS:
        - mettiamo tutti gli unmatched node W sulla destra nel layer 0

- **scenario 2: vendita di case**
    - insieme S di sellers
    - insieme B di buyers
    - decisioni libere di buyers basate sui prezzi e sulle loro valutazioni
    - abbiamo:
        - seller
        - prezzo (associato ad ogni seller)
        - buyer
        - valutazioni (associate ad ogni buyer per ogni seller)

- ciascun seller vende casa a prezzo $p_i \geq 0$
- payoff buyer è: $v_{ij} - p_i$
- sellers preferiti del buyer j:
  - sellers che massimizzano il suo payoff
  - se payoff è negativo per ogni seller, non c'è seller preferito
- grafo preferred-seller: archi tra i buyers e i loro sellers preferiti
- assunzione: $v_{ij} \geq 0$ e $p_i \geq 0$
- scopo: assegnare a ciascun buyer una delle case a lui più conventienti:
  - trovare un matching perfetto nel grafo preferred-seller
- questo pricing è chiamato **market-clearing**: pricing che risolve i conflitti
- in caso di cravatte (più di un seller preferito per qualche buyer), è richiesta coordinazione
- **esistenza dei prezzi market-clearing**:
  - **teorema**: i prezzi market-clearing esistono per ogni possibile insieme di valutazioni
  - **dimostrazione**:
    - dimostriamo teorema fornendo procedura adatta che termina con prezzi market-clearing
    - idea 1: aumento prezzi
      - se un insieme di prezzi P non è market-clearing, allora esiste un constricted set di buyers C
      - sia $N(C)$ l'insieme dei neighbors dei buyers in C nel grafo preferred-seller, allora $|N(C)| < C$
      - incrementiamo di 1 i prezzi di $N(C)$, in questo modo si cerca di dissuadere qualche buyer in C e quindi si cerca di eliminare il constricted set
    - idea 2: argomento di riduzione
      - prima dell'inizio di ciascun round, tutti i prezzi sono decrementati di un quantità fissata δ in maniera che il prezzo più piccolo diventi 0
      - questo aumenta i payoffs ($v_{ij} - p_i$) di ogni buyer a causa dei prezzi P diminuiti di δ. Il nuovo insieme di prezzi P' non modifica l'ordine dei payoffs
      - questa idea garantisce che ciascun buyer ha payoff almeno 0 per almeno 1 seller (quello con prezzo 0) e quindi ciascun buyer ha almeno 1 seller preferito all'inizio di ciascun round e alla fine della procedura
    - procedura (prezzi market-clearing):
      1. sia $p_i = 0, \forall i$
      2. costruisci grafo preferred-seller G
      3. IF **c'è matching perfetto** in G:
         - **STOP**: i prezzi sono market-clearing
      4. ELSE:
         - sia C constricted set di buyers ed $N(C)$ i loro neighbors
      5. incrementa di 1 i prezzi di ciascun seller in $N(C)$
      6. decrementa **tutti** i prezzi di δ in modo che il prezzo più piccolo diventa 0
      7. ripetiamo da **2**
    - alla fine di esecuzione procedura, da idea 2 il prezzo più basso è 0
    - quindi , ogni buyer ha payoff almeno 0 per almeno 1 seller
    - quindi ogni buyer ha almeno 1 seller preferito e non ci sono constricted set: c'è un **matching perfetto** e quindi l'insieme finale dei prezzi e market-clearing
    - ma aumentare i prezzi anche se elimina constricted set può anche crearne di nuovi
    - dobbiamo dimostrare che la procedura termina: usiamo una funzione potenziale ϕ definita sui prezzi P
    - P_0: insieme iniziale dei prezzi (tutti 0)
    - P_k: insieme dei prezzi all'inizio del round k:
    - dimostriamo che:

- $\phi(P_0) \geq 0$
        - $\phi(P_k) \geq 0, \forall k$
        - $\phi(P_{k+1}) < \phi(P_k), \forall k$
      - argomento di ϕ:
        - dato P, sia $i(j)$ un seller preferito del buyer j in accordo a P
        - potenziale del buyer j: massimo payoff del buyer j, quindi $v_{i(j),j} - p_{i(j)}$
        - potenziale del seller i: prezzo del seller i, quindi p_i
        - potenziale di P: somma dei potenziali per buyer e seller, quindi $(\sum_{j \in B} v_{i(j)} - p_{i(j)}) + (\sum_{i \in S} p_i) = \phi(P)$
      - dimostriamo $\phi \geq 0$:
        - consideriamo qualsiasi round k con prezzi P_k
        - dato che, il prezzo minimo in P_k è 0 e quindi il massimo payoff di ciascun buyer è ≥ 0 : $(\sum_{j \in B} v_{i(j)} - p_{i(j)}) + (\sum_{i \in S} p_i) \geq 0$
      - dimostriamo $\phi(P_{k+1}) < \phi(P_k)$:
        - i soli steps che modificano i prezzi sono **5** e **6**
        - per **5**:
          - i prezzi dei sellers in $N(C)$ sono incrementati di 1, quindi in totale ϕ viene incrementato di $|N(C)|$
          - i payoffs dei buyers sono decrementati di 1, quindi in totale ϕ viene decrementato di $|C|$
          - dato che $|N(C)| < |C|$, allora ϕ è decrementato di almeno 1
        - per **6**:
          - tutti i prezzi dei sellers sono decrementati di δ, quindi in totale ϕ viene decrementato di $\delta \cdot |S|$
          - tutti i payoffs dei buyers sono incrementati di δ, quindi in totale ϕ viene incrementato di $\delta \cdot |B| = \delta \cdot |S|$
          - quindi ϕ non viene cambiato dallo step **6**
        - quindi abbiamo dimostrato che ϕ decresce strettamente ad ogni round
  - **ottimalità dei prezzi market-clearing**:
    - valutazione assegnamento
    - assumiamo che i sellers guadagnano i loro prezzi, il payoff dei sellers è il loro prezzo
    - denotiamo con $i(j)$ un seller preferito i del buyer j, cioè il seller i è abbinato al buyer j in M
    - **social welfare**: felicità globale di tutti i partecipanti: somma dei payoffs di buyers e sellers $SW(M) = (\sum_{j \in B} v_{i(j)} - p_{i(j)}) + (\sum_{i \in S} P_i) = \sum_{j \in B} v_{i(j)}$: valutazione totale di M
    - **teorema**: per ogni insieme di prezzi market-clearing, qualsiasi matching perfetto M nel grafo preferred-seller ha massimo social welfare di qualsiasi altro assegnamento, cioè massimizza $SW(M)$
    - **dimostrazione**:
      - dato che M abbina ogni buyer al suo seller preferito, M massimizza il payoff totale dei buyers in M
      - il payoff totale in M è: $\sum_{j \in B} v_{i(j)} - p_{i(j)} = (\sum_{j \in B} v_{i(j)}) - (\sum_{i \in S} p_i) = SW(M) -$ **somma dei prezzi**
      - dato che la **somma dei prezzi** non dipende da M, ed M massimizza il payoff totale, allora M massimizza $SW(M)$
    - **revenue**: revenue totale dei sellers, cioè $REV(M) = \sum_{i \in S} p_i$, terribile, dato che la procedura potrebbe ritornare tutti i prezzi a 0 (caso in cui ogni buyer ha la stessa valutazione per tutti i sellers)

- **le aste**:
  - usata dalle compagnie per vendere prodotti

- portata nella vita di tutti i giorni da internet: eBay, sponsored web search
- ambientazione:
  - 1 seller vende all'asta 1 item ad un insieme di buyers che fanno offerte per prendere l'item
  - seller $=$ banditore
  - buyers $=$ offerenti
- ciascun offerente ha valore v_i per l'item offerto dal banditore
- ciascun offerente è interessato a comprare l'item ad un prezzo fino a v_i
- v_i è il true value dell'offerente
- asta inutile se banditore conosce i true values degli offerenti: il banditore venderebbe l'item all'offerente con piàù alto true value per un prezzo vicino al suo true value
- **tipi di aste**:
  - asta **ascending-bid**:
    - interattivo in tempo reale
    - banditore gradualmente alza il prezzo
    - offerenti abbandonano fino a quando solo 1 rimane
    - il rimanente vince l'item al prezzo finale
  - asta **descending-bid**:
    - interattivo in tempo reale
    - banditore gradualmente abbassa il prezzo
    - quando un offerente accetta, vince al prezzo corrente
  - asta **first-price sealed-bid**:
    - offerenti sottomettono offerte sigillate al banditore
    - l'offerta più alta vince
    - chi vince paga la sua offerta
  - asta **second-price sealed-bid**:
    - offerenti sottomettono offerte sigillate al banditore
    - l'offerta più alta vince
    - chi vince paga la seconda offerta più alta
- **relazioni tra tipi**:
  - **ascending-bid** e **second-price sealed-bid**:
    - ascending-bid: prezzo alzato gradualemente fino a che solo 1 offerente rimane: per ogni offerente i c'è prezzo b_i al quale abbandonerà, offerente più alto paga il prezzo dell'ultimo offerente che ha abbandonato, cioè paga il prezzo del secondo offerente più alto
    - equivalente a second-price: i prezzi b_i giocano il ruolo delle offerte
  - **descending-bid** e **first-price sealed-bid**:
    - descending-bid: prezzo abassato gradualmente fino a che qualcuno accetta: per ogni offerente i c'è prezzo b_i al quale romperà il silenzio e accetterà il prezzo b_i
    - equivalente a first-price: i prezzi b_i giocano il ruolo delle offerte
- aste first-price sono simulazioni sealed-bid di aste descending-bid
- aste second-price sono simulazioni sealed-bid di aste ascending-bid
- analizziamo comportamento offerenti aste second-price usando ascending-bid, quando offerente dovrebbe abbandonare:
  - dopo che il prezzo raggiunge il true value v_i:
    - stando dentro: paga più di v_i oppure perde
    - meglio abbandonare
  - prima che il prezzo raggiunge il true value v_i:
    - stando dentro: potrebbe vincere l'item ad un prezzo più basso di v_i
    - abbandonando: non vince nulla
    - meglio rimanere
  - deve abbandonare quando il prezzo diventa esattamente uguale al suo true value v_i

- tornando alle aste second-price, la soluzione migliore per l'offerente è settare la sua offerta $b_i = v_i$: **truthful bidding**
- per le aste first-price:
  - gli offerenti tenderanno ad offrire meno per fare un affare: $b_i < v_i$
- l'abbassamento delle offerte compensa la differenza tra il primo prezzo e il secondo prezzo: stessa rendita attesa al seller
- **aste second-price**:
  - ampiamente utilizzate: eBay, sponsored web search
  - **truthful**: offrire il true value è la strategia più conveniente
  - ciascun offerente i:
    - ha true value v_i
    - la sua strategia è selezionare b_i:
      - se b_i non è l'offerta vincente: payoff $= 0$
      - se b_i è l'offerta vincente, sia b_j la seconda offerta più alta: payoff $= v_i - b_j$
    - se ci sono cravatte (ties):
      - 2 o più offerenti offrono la stessa offerta più alta
      - l'offerente i con minimo i vince
      - il secondo prezzo più alto sarà quindi uguale al primo: i ha payoff $= 0$
- **aste first-price**:
  - ciascun offerente i:
    - ha true value v_i
    - la sua strategia è selezionare b_i:
      - se b_i non è l'offerta vincente: payoff $= 0$
      - se b_i è l'offerta vincente: payoff $= v_i - b_i$
    - **non truthful**: offrire il true value porta sempre a payoff $= 0$
    - gli offerenti devono offreire meno per prendere payoffs positivi, quanto di meno?
      - offrire vicino al true value porta a payoff piccoli in caso di vittoria
      - offerte più basse riducono le possibiiltà di vittoria
    - compromesso difficile:
      - richiede conoscenza degli altri offerenti (delle loro valutazioni)