Sprzedaż online

Piotr Sankowski
Uniwersytet Warszawski
Warszawa 18.04.2013

Plan wykładu

- Problem skojarzeń online
  - Algorytm zachłanny
  - Algorytm losowo rankujący
  - Dolne ograniczenie
- Problem aukcji sponsorowanych online
  - Sformułowanie liniowe
  - Algorytm online

Skojarzenia online

Skojarzenie to niezależny zbiór krawędzi grafu.

Doskonałe skojarzenie to skojarzenie zawierające wszystkie wierzchołki grafu.

Niech $G(U, V, E)$ będzie dwudzielnym grafem na $2n$ wierzchołkach zawierającym doskonałe skojarzenie.

Będziemy rozważać problem konstruowania jak największego skojarzenia online.

Skojarzenia online

Załóżmy, że wierzchołki U pojawiają się w zadanej kolejności, a krawędzie sąsiednie do nich ujawniane są w raz z ich pojawieniem się.

Po pojawieniu się wierzchołka u z U naszym zadaniem jest zdecydowanie z jakim wierzchołkiem v z V go połączymy.

Raz przydzielonego wierzchołka v nie możemy zmienić.

Naszym celem jest połączenie wierzchołków tak aby rozmiar skojarzenia był jak największy.

Algorytm zachłanny

Algorytm zachłanny zawsze łączy wierzchołek u z dowolnym wolnym wierzchołkiem z V.

Obserwacja 1 *Algorytm zachłanny konstruuje skojarzenie rozmiaru co najmniej* $\frac{n}{2}$.

Obserwacja 2 *Dla każdego deterministycznego algorytmu istnieje scenariusz, w którym konstruuje on skojarzenie rozmiaru co najwyżej* $\frac{n}{2}$.

Dowody na ćwiczeniach.

Adaptacyjny przeciwnik

Adaptacyjny przeciwnik może wybrać kolejność pojawiania się wierzchołków i krawędzi w zależności od wyborów (i losowań) algorytmu.

Obserwacja 3 *Adaptacyjny przeciwnik, może zmusić, każdy randomizowany algorytm do skojarzenia co najwyżej* $\frac{n}{2} + O(\log n)$ *wierzchołków.*

Dowód na ćwiczeniach.

Algorytm rankujący

Następujący algorytm nazwiemy *algorytmem rankującym*:

Inicjalizacja wybierz losową permutację V, tzn. każdemu wierzchołkowi przypisz losową rangę,

Kojarzenie każdy pojawiający się wierzchołek u połącz z wolnym wierzchołkiem z V o najwyższej randze.

Algorytm rankujący

Pokażemy, że algorytm rankujący kojarzy $(1 - \frac{1}{e})n$ wierzchołków w przypadku nieadaptacyjnego adwersarza.

Wydaje się, że ten algorytm jest tak dobry jak algorytm losowy, ale:

Obserwacja 4 *Algorytm losowy zachłanny kojarzy co najwyżej* $\frac{n}{2} + O(\log n)$ *wierzchołków.*

Dowód na ćwiczeniach.

Algorytm rankujący

Po fazie inicjalizacji algorytmu mamy określone porządki na wierzchołkach:

- w U porządek ich pojawiania się,
- w V losowy porządek wybrany przez algorytm.

Niech Ranking(σ) oznacza uruchomienie algorytmu dla porządku na V zadanego przez σ.

Niech $m^* : V \to U$ oznacza doskonałe skojarzenie.

Algorytm rankujący

Lemat 5 *Ustalmy $u \in U$ oraz niech $v = m^*(u)$. Jeżeli v nie jest skojarzone przez Ranking(σ), to u jest skojarzone przez Ranking(σ) do wierzchołka v', którego ranga $\sigma(v')$ jest mniejsza niż $t = \sigma(v)$.*

Jeżeli v nie jest skojarzone przez Ranking(σ), to w momencie swojego pojawiania się u ma dostępnego sąsiada, bo v jest jednym z nich.

u musi zostać skojarzone do sąsiada o mniejszej randze niż v.

Algorytm rankujący

Lemat 6 *Niech $u \in U$ oraz $v = m^*(u)$. Niech σ' będzie permutacją, oraz niech σ_i będzie permutacją otrzymaną z σ' poprzez usunięcie v i dodanie go na pozycji i. Jeżeli v jest nieskojarzone przez $Ranking(\sigma')$, to dla każdego i, wierzchołek u jest skojarzony przez $Ranking(\sigma_i)$ do wierzchołka v_i, którego ranga $\sigma_i(v_i)$ wynosi co najwyżej $t = \sigma'(v)$.*

Niech $m' = Ranking(\sigma')$ oraz niech $m_i = Ranking(\sigma_i)$.

Algorytm rankujący

Skojarzenia m' oraz m_i, jeżeli nie są identyczne, to różnią się wzdłuż jednej ścieżki alternującej p zaczynającej się w v krawędzią z m_i.

Co więcej, możemy zauważyć, że wierzchołki V na ścieżce p pojawiają się w kolejności rosnących rang.

Dlatego m_i kojarzy u do wierzchołka v_i, którego ranga jest $\sigma_i(v_i) \leq \sigma_i(v')$, gdzie $v' = m'(u)$.

Algorytm rankujący

Z definicji σ_i wiemy, że $|\sigma_i(v') - \sigma'(v')| \leq 1$.

Z Lematu 5 wiemy, że $\sigma'(v') < t$.

Dlatego mamy $\sigma_i(v_i) < 1 + t$.

Z całkowitości otrzymujemy, że $\sigma_i(v_i) \leq t$.

Algorytm rankujący

Lemat 7 *Niech x_t oznacza prawdopodobieństwo, że dla σ wierzchołek V rangi t jest skojarzony przez Ranking(σ), wtedy* $1 - x_t \leq \frac{1}{n} \sum_{1 \leq n \leq t} x_s$.

Zanim udowodnimy ten lemma pokażemy jak z niego wynika:

Twierdzenie 8 *Algorytm rankujący ma współczynnik kompetytywności wynoszący asymptotycznie* $1 - \frac{1}{e} \approx 0.63$.

Algorytm rankujący

Ponieważ graf G zawiera doskonałe skojarzenie, to współczynnik kompetytywności jest równy infimum z

$$\frac{1}{n} \sum_{1 \leq s \leq n} x_s.$$

Zdefiniujmy S_t przez:

$$S_t = \sum_{1 \leq s \leq t} x_s.$$

Algorytm rankujący

Z Lematu 7 otrzymujemy, że:

$$1 + S_{t-1} \leq S_t \left(1 + \frac{1}{n}\right).$$

Możemy teraz zauważyć, że infimum jest nie mniejsze niż gdy wszystkie nierówności zachodzą z równością. Oznacza to, że:

$$S_t \geq \sum_{s=1}^{t} \left(1 - \frac{1}{n+1}\right)^s,$$

dla każdego t.

Algorytm rankujący

Współczynnik kompetytywności wynosi więc co najmniej:

$$\frac{S_n}{n} \geq \frac{1}{n} \sum_{s=1}^{t} \left(1 - \frac{1}{n+1}\right)^s =$$

$$= \frac{1}{n} \cdot \sum_{s=1}^{t} \left(1 - \frac{1}{n+1}\right)^s = \frac{1}{n} \times \frac{1 - \left(1 - \frac{1}{n+1}\right)^n}{1 - \left(1 - \frac{1}{n+1}\right)} =$$

$$= \frac{n+1}{n} \left(1 - \left(1 - \frac{1}{n+1}\right)^n\right) \rightarrow_{n \rightarrow \infty} 1 - \frac{1}{e}.$$

Algorytm rankujący

Wróćmy do dowodu Lematu 7.

Mając daną losową permutację σ, zdefiniujmy nową losową permutację σ' otrzymaną z σ poprzez:

- wybranie wierzchołka v z V z rozkładem jednostajnym,
- wyjecie v z σ i włożenie go tak by jego ranga wynosiła t.

Rozważmy skojarzenie m' otrzymane przez Ranking(σ').

Algorytm rankujący

Niech u będzie takie, że $v = m^*(u)$.

Z Lematu 6 zastosowanego do $\sigma_i = \sigma$ wiemy, że jeżeli v nie jest skojarzone przez Ranking(σ'), to u jest skojarzone przez Ranking(σ_i) do wierzchołka $\bar{v}$ takiego, że $\sigma(\bar{v}) \leq t$.

Prawdopodobieństwo, że v nie jest skojarzone przez Ranking(σ') wynosi $1 - x_t$.

Algorytm rankujący

Niech R_t oznacza zbiór wierzchołków z U, które zostały skojarzone w algorytmie do wierzchołków z V o randze mniejszej równej t.

Innymi słowy pokazaliśmy, że jeżeli v nie jest skojarzone przez Ranking(σ') to $u \in R_t$.

Zauważmy, że u jest niezależne of σ, a więc także od R_t.

Dla danej permutacji σ prawdopodobieństwo, że $u \in R_t$ wynosi $\frac{|R_t|}{n}$.

Algorytm rankujący

Z drugiej strony wiemy, że:

$$E_\sigma[|R_t|] = \sum_{s=1}^{t} x_s.$$

Otrzymujemy więc, że

$$1 - x_t \leq E_\sigma\left[\frac{|R_t|}{n}\right] = \frac{1}{n}\sum_{s=1}^{t} x_s.$$

Co kończy dowód Lematu 7

Dolne ograniczenie

Niech graf T będzie grafem dwudzielnym, w którym u jest połączone z v wttw $u \leq v$.

Załóżmy też, że wierzchołki z T pojawiają się w odwróconej kolejności.

Twierdzenie 9 (Ćwiczenia) *Rozmiar skojarzenia wygenerowany przez dowolny algorytm jest ograniczona z góry przez oczekiwany rozmiar skojarzenie wygenerowanego przez algorytm losowy na grafie T.*

Dolne ograniczenie

Twierdzenie 10 *Oczekiwany rozmiar skojarzenie wygenerowanego przez algorytm losowy na grafie T wynosi $n(1-\frac{1}{e})+o(n)$.*

Załóżmy, że zostało na jeszcze l wierzchołków z V gdy pojawia się k wierzchołek z U.

Wtedy te l wierzchołków V jest losowym podzbiorem pierwszych $n-k+1$ wierzchołków.

Dolne ograniczenie

Niech $y(t)$ będzie zmienną losową oznaczającą liczbę dostępnych wierzchołków w V gdy pojawia się t'ty wierzchołek z U.

Wartość $y(t)$ może zmaleć o:

- 2 jeżeli $t+1$ wierzchołek był dostępny,
- 1 w przeciwnym przypadku.

Ponieważ zbiór dostępnych wierzchołków z V jest losowy, to

$$E[\Delta y(t)] = -1 - \frac{y(t)}{n-t}\frac{y(t)-1}{y(t)} = -1 - \frac{y(t)-1}{n-t}.$$

Dolne ograniczenie

Zgodnie z Twierdzeniem Kurtza przy n dążącym do nieskończoności, wartości oczekiwane są bliskie rozwiązaniu równania różniczkowego:

$$\frac{dy}{dt} = -1 - \frac{y-1}{n-t}$$

Rozwiązując to równanie otrzymujemy:

$$y = 1 + (n-t)\left(\frac{n-1}{n} - \ln\frac{n-t}{n}\right).$$

Dolne ograniczenie

Wstawiając $y = 1$ i rozwiązując na x otrzymujemy, że $x = n\left(1 - \frac{1}{e}\right) + o(n)$.

Algorytm rankujący jest więc $1 - \frac{1}{e}$ kompetytywny.

Zrobimy teraz to samo dla aukcji online.

Aukcje online

Rozważymy model, w którym wyszukiwarka otrzymuje oferty ogłaszających oraz ich maksymalny dzienny budżet.

W ciągu dnia użytkownicy wykonują pewne wyszukiwania, a ogłaszający są przypisywani slotom reklamowym, oraz obciążani ofertą jaką zgłosili.

Zakładamy, że każda strona ma tylko jeden slot.

Naszym celem jest zmaksymalizować zysk bez przekraczania budżetów graczy.

Aukcje online

Niech n będzie liczbą ogłaszających, a m niech będzie liczbą słów kluczowych.

Zakładamy, że gracz j zgłasza ofertę b_{ij} za słowo i oraz całkowity budżet B_j.

Zakładamy, co więcej, że $b_{ij} \ll B_j$.

Zakładamy, że wyszukiwarka zna r_i liczbę zapytań o i'te słowo kluczowe.

Aukcje online

Możemy zapisać następując program liniowy:

$$\max \quad \sum_{i=1}^{m}\sum_{j=1}^{n} b_{ij}x_{ij}$$

$$\forall_{1\le i\le m} \quad \sum_{j=1}^{n} x_{ij} \le r_i$$

$$\forall_{1\le j\le n} \quad \sum_{i=1}^{m} b_{ij}x_{ij} \le B_j$$

$$\forall_{1\le i\le m,1\le j\le n} \quad x_{ij} \ge 0.$$

Aukcje online

Program dualny natomiast jest następujący:

$$\begin{aligned} \min \quad & \sum_{j=1}^{n} B_j \beta_j + \sum_{i=1}^{m} r_i \alpha_i \\ \forall_{1 \le i \le m, 1 \le j \le n} \quad & \alpha_i + b_{ij} \beta_j \ge b_{ij} \\ \forall_{1 \le j \le n} \quad & \beta_j \ge 0 \\ \forall_{1 \le i \le m} \quad & \alpha_i \ge 0. \end{aligned}$$

Gracz j jest przypisany słowu i jeżeli $(1 - \beta_j) b_{ij} = \max_{1 \le k \le n} (1 - \beta_k) b_{ik}$.

Aukcje online

Za każdym razem gdy pojawia się wyszukiwanie o słowo i wyszukiwarka przypisuje miejsce ogłoszeniowe graczowi j o największym $b_{ij}(1-\beta_j)$.

Innymi słowy oferta gracza j jest przeskalowana w dół przez $1-\beta_j$.

W ten sposób powstały przydział ogłoszeń będzie optymalny.

Aukcje online

Co zrobić jednak gdy nie znane nam są wartości r_i.

Możemy spróbować przypisywać zachłannie, tzn. przypisywać ogłoszenia graczom, którzy dają największą ofertę i nie wyczerpali jeszcze swojego budżetu.

Algorytm zachłanny jest co najmniej i co najwyżej 1/2 kompetytywny.

Czy możemy to zrobić lepiej?

Aukcje online

Niech $\phi(x) = 1 - e^{x-1}$.

Niech f_j oznacza ułamek budżetu jaki wydał dotychczas gracz j.

Algorytm *rozmywający* przypisuje ogłoszenie graczowi o najwyższym $b_{ij}\phi(f_j)$.

Twierdzenie 11 *Współczynnik kompetytywności algorytmu rozmywającego wynosi* $1 - \frac{1}{e}$.

Aukcje online

Niech k będzie dostatecznie duża liczbą...

Powiemy, że ogłaszający jest typu j jeżeli wydał budżet w ułamku z przedziału $(\frac{j-1}{k}, \frac{j}{k}]$.

Niech s_j oznacza całkowity budżet graczy typu j.

Niech w_i będzie całkowitym wydatkiem graczy z ułamku budżetu $(\frac{j-1}{k}, \frac{j}{k}]$.

Aukcje online

Zdefiniujmy też dyskretną wersję funkcji ϕ:

$$\Phi(s) = 1 - \left(1 - \frac{1}{k}\right)^{k-s},$$

możemy zauważyć, że jak k dąży do nieskończoności to $\Phi(s) \to \phi(\frac{s}{k})$.

Niech OPT będzie optymalnym rozwiązaniem offline.

Dla prostoty załóżmy, że optymalne rozwiązanie wydaje całe budżety wszystkich graczy.

Aukcje online

Lemat 12 *Na koniec działania algorytmu zachodzi:*

$$\sum_{i=0}^{k} \Phi(i) s_i \leq \sum_{i=0}^{k} \Phi(i) w_i.$$

Rozważmy moment czasu gdy pojawi się zapytanie q.

Załóżmy, że OPT przypisuje q graczowi, który ma typ t, a którego typem na koniec działania algorytmu jest t'.

Aukcje online

Niech b_{opt} oraz b_{alg} będzie opłatą jaką dostaje OPT i algorytm od graczy za q.

Niech i będzie typem gracza, któremu algorytm przypisuje q.

Wtedy mamy:

$$\Phi(t')b_{opt} \leq \Phi(t)b_{opt} \leq \Phi(i)b_{alg}.$$

Sumując po wszystkich zapytaniach

otrzymujemy tezę lematu.

Aukcje online

Korzystając z tego lematu udowodnimy teraz twierdzenie.

Z definicji wiemy, że $w_i \leq \frac{1}{k} \sum_{j=i}^{k} s_j$, a z lematu:

$$\sum_{i=0} \Phi(i) s_i \leq \frac{1}{k} \sum_{i=0}^{k} \Phi(i) \sum_{j=i}^{k} s_j = .$$

zmieniając kolejność sumowania otrzymujemy:

$$= \frac{1}{k} \sum_{i=0}^{k} \left(\sum_{j=0}^{i} \Phi(j) \right) s_i \approx \sum_{i=0}^{k} \left(\frac{i}{k} + \Phi(i) - \Phi(0)) \right) s_i$$

Aukcje online

Otrzymujemy, więc:

$$\left(\Phi(0) - O(\frac{1}{k})\right) \sum_{i=0}^{k} s_i \leq \sum_{i=0}^{k} \frac{i}{k} s_i.$$

Jak k dąży do nieskończoności to
$\Phi(0) = 1 - \left(1 - \frac{1}{k}\right)^k$ dąży do $1 - \frac{1}{e}$.

A więc lewa strona dąży do $(1 - \frac{1}{e})OPT$.

Natomiast, prawa strona jest równa dokładnie zyskowi algorymu.

Aukcje online

Problemy otwarte:

- budżety nigdy nie są ”twarde”,
- budżety używane są przez graczy aby wyrazić ich preferencje względem długości trwania kampanii,
- co z klikaniem przez roboty i przeciwników?
- modele z wieloma wyszukiwarkami,
- nie analizowaliśmy zupełnie strategii graczy,
- co z problemem wyznaczania klikalności?