Uniwersytet Warszawski
Wydział Matematyki, Informatyki i Mechaniki

Piotr Wygocki

O najbliższych sąsiadach

Autoreferat rozprawy doktorskiej

Luty 2019

Opiekun rozprawy doktorskiej
dr hab. Piotr Sankowski
Instytut Informatyki
Uniwersytet Warszawski

Niniejsza rozprawa poświęcona jest różnym aspektom wyszukiwania najbliższych sąsiadów, czyli obiektów, które są do siebie podobne. Techniki związane z wyszukiwaniem najbliższych sąsiadów są stosowane w różnych obszarach, w tym geometrii obliczeniowej, bazach danych, robotyce, sekwencjonowaniu DNA, automatycznym sprawdzaniu pisowni, klasyfikacji, klastrowaniu, podobieństwie chemicznym, wizji komputerowej, wykrywaniu plagiatów, systemach rekomendacyjnych, marketingu wirusowym, sieciach społecznościowych, kompresji danych, teorii kodowania i rozpoznawaniu wzorców. Podstawowa motywacja jest następująca. Wiedza na temat danego obiektu może zostać znacznie poszerzona poprzez badanie obiektów, które są do niego podobne lub w jakiś sposób z nim powiązane. Wyszukiwanie najbliższych sąsiadów umożliwia zidentyfikowanie takich obiektów. W zależności od tego, co w danej sytuacji rozumiemy przez "podobny" lub "powiązany", można wprowadzić odpowiednią definicję sąsiada.

Wśród licznych kontekstów, w których stosuje się wyszukiwanie najbliższego sąsiada, w tej rozprawie skoncentrujemy się na wyszukiwaniu podobieństw (ang. similarity search) w przestrzeniach metrycznych i rozpowszechnianiu informacji w sieciach społecznościowych.

Przestrzenie metryczne

W tym przypadku interesują nas wydajne algorytmy wyszukiwania podobieństw w przestrzeniach metrycznych. Dane wejściowe składają się ze zbioru punktów w $\mathbb{R}^d$. Zbiór ten może zostać wstępnie przetworzony (ang. pre–processing) w celu zbudowania struktury danych, która będzie używana do zapytań. Zapytanie składa się z pojedynczego punktu w $\mathbb{R}^d$ a odpowiedzią jest najbliższy sąsiad tego punktu, tj. punkt ze zbioru wejściowego, który jest najbliżej punktu zapytania w danej metryce. W tej rozprawie rozważamy równoważny problem[1], w którym dla danego zbioru wejściowego oraz promienia R, tworzymy strukturę danych odpowiadającą na zapytania dotyczące bliskiego sąsiada. Bliski sąsiad to dowolny punkt ze zbioru wejściowego, który jest bliżej niż R do punktu z zapytania.

Istnieje wiele zastosowań tego problemu. Rozważmy na przykład wyszukiwanie dokumentów w Internecie. W fazie zapytania dostajemy dokument i chcemy znaleźć różne wersje tego samego dokumentu w sieci. Innym scenariuszem związanym z wyszukiwaniem w Internecie jest znalezienie najbardziej podobnego zdjęcia do tego udostępnionego przez użytkownika. Ponadto szeroki zakres aplikacji obejmuje wykorzystanie wyszukiwania sąsiadów do celów klasyfikacji. Klasyfikację danego obiektu uzyskujemy poprzez wyszukanie sąsiadów wśród już sklasyfikowanych obiektów. Uzyskaną w ten sposób informację można również wykorzystać do predykcji. Określone cechy obiektu można przewidzieć na podstawie wiedzy o podobnych obiektach wykrytych jako sąsiedzi.

W tej pracy rozważamy przestrzenie wysoko–wymiarowe i metryki ℓ_p. Naiwny algorytm zapytań polega na skanowaniu wszystkich punktów wejściowych i wybraniu bliskiego (najbliższego) sąsiada. Jednakże, dla takiego algorytmu czas zapytania jest liniowy ze względu na liczność zbioru wejściowego, co nie jest akceptowalne w wielu praktycznych zastosowaniach. W tej pracy rozważamy algorytmy, dla których czas zapytania jest pod–liniowy ze względu na rozmiar zestawu wejściowego: $\mathcal{O}(n^\gamma)$ dla $\gamma < 1$. Ponadto zakładamy, że wymiar przestrzeni punktów jest wysoki, więc ani czas wstępnego przetwarzania, ani czas zapytania nie mogą być wykładnicze ze względu na wymiar przestrzeni. Niestety nie ma obecnie struktur danych spełniających powyższe warunki. Co więcej, istnienie takiego algorytmu byłoby sprzeczne z silną

[1]Każdy z problemów można zredukować do drugiego z niewielkim zwiększeniem złożoności [5].

hipotezą czasu wykładniczego (ang. strong exponential time hypothesis) [6]. W związku z powyższym w tej pracy rozważamy problem c–aproksymowanego bliskiego sąsiada, w którym dla danego zapytania zamiast punktu odległego o mniej niż R dopuszczamy zwrócenie punktu odległego o mniej niż cR.

Problem c–aproksymowanego bliskiego sąsiada zyskał wiele uwagi w literaturze badawczej [5, 7, 8, 9, 10, 11, 12]. Znaczna część rozważanych algorytmów spełnia probabilistyczne gwarancje typu Monte Carlo. W tej pracy skupiamy się na algorytmach, które spełniają mocniejsze założenia typu Las Vegas.

Algorytmy prezentowane w tej rozprawie działają dla metryk ℓ_p dla $p \in [1, \infty]$. Jednak najlepsze wyniki uzyskano dla ℓ_2. W szczególności dla $c < 2$ i $p = 2$ przedstawiamy algorytm o niemal optymalnej złożoności czasu przetwarzania zbioru wejściowego równej $\mathcal{O}(d^{\omega-1}n)$ i z czasem zapytania równym $\mathcal{O}(d^{\omega-1} + dn^{\frac{1}{1+\mathcal{O}(\epsilon^2 \log^{-1} \frac{1}{\epsilon})}})$ dla $\epsilon = c - 1$. Są to, według wiedzy autora, najlepsze znane wyniki dla metryk euklidesowych z gwarancjami typu Las Vegas.

Prezentujemy wiele wersji algorytmów, które umożliwiają płynne przejście pomiędzy optymalizowaniem czasu wstępnego przetwarzania i czasu zapytania. W szczególności przedstawiamy algorytm z efektywnym czasem zapytania $\mathcal{O}(d^{\omega-1})$ i z czasem wstępnego przetwarzania równym $n^{1+\mathcal{O}(\frac{1}{\epsilon^2} \log \frac{1}{\epsilon})}$. Te wyniki są zbliżone do najlepszych wyników uzyskanych przez algorytmy Monte Carlo [11].[2] Ponadto prezentujemy algorytmy dla dowolnego $p \in [1, \infty]$. Niestety gwarancje algorytmów stają się gorsze gdy p oddala się od 2.

Zastosowane metody

Prezentujemy dwa podejścia do rozwiązania NN_{wfn}[3]

- **LSH** (ang. Locality Sensitive Hashing). W tym podejściu używamy funkcji haszujących z gwarancjami typu Las Vegas, t.j. takich funkcji, dla których z prawdopodobieństwem 1 dwa "bliskie" punkty pozostają bliskie po zaaplikowaniu funkcji. Dodatkowo, z dużym prawdopodobieństwem, dwa "odległe" punkty pozostaną "odległe" po zaaplikowaniu funkcji. Innymi słowy, lokalne funkcje haszujące zgrubnie zachowują

[2] Dotyczy to zarówno wersji z efektywnym czasem przetwarzania wstępnego jak i wersji z efektywnym czasem zapytania.

[3] $\mathsf{NN}_{\text{wfn}}(c, n)$ oznacza problem wyszukiwania c–aproksymowanego bliskiego sąsiada bez fałszywie ujemnych wyników (ang. false positives) ze zbiorem wejściowym wielkości n.

odległość. Dzięki temu można ich użyć do zmniejszenia wymiaru rozważanej przestrzeni, co pomoże nam zbudować algorytm dla NN_{wfn}.

- **Metoda zliczania**. W tym podejściu kwantyzujemy naszą przestrzeń tak, aby wszystkie punkty miały współczynniki całkowitoliczbowe. Po kwantyzacji istnieje skończona liczba możliwych sąsiadów dla punktów wejściowych. Możemy przechowywać każdy z nich, co daje prosty algorytm dla NN_{wfn}.

W celu poprawienia wyników, obie techniki łączymy z redukcją wymiaru danej przestrzeni do $\mathcal{O}(\log n)$. Ta operacja dodaje czynnik $\mathcal{O}(d/\log n)$ do złożoności, ale pozwala nam korzystać z algorytmów wykładniczych ze względu na wymiar. Pokazujemy, jak zredukować $\mathsf{NN}_{\text{wfn}}(c,d)$ w ℓ_2^d do $d/\log n$ instancji $\mathsf{NN}_{\text{wfn}}(\mathcal{O}(c),\mathcal{O}(\log n))$. Redukcja opiera się na znanym lemacie Johnsona-Lindenstraussa [13]. Wprowadzamy $d/\log n$ odwzorowań liniowych, z których każde zmniejsza wymiar pierwotnej przestrzeni. Każde odwzorowanie zgrubnie zachowuje długość wektora i dodatkowo co najmniej jedno z nich jej nie zwiększa. Właściwość nie zwiększania długości wektora jest tu istotna. Dla dwóch "bliskich" wektorów $x, y \in \mathbb{R}^d$: $\|x-y\|_2 < 1$ i przekształcenia liniowego A, Ax i Ay są "bliskie" wtedy i tylko wtedy, gdy $\|Ax-Ay\|_2 = \|A(x-y)\|_2 < 1$, więc A odwzorowuje "mały" wektor $x-y$, na "mały" wektor $A(x-y)$.

Następujący wniosek zawiera kluczowe wyniki dla redukcji wymiarów. Pokazujemy, że każdy przypadek problemu bliskich sąsiadów może być zredukowany do małej liczby instancji o wymiarze logarytmicznym ze względu na liczbę punktów wejściowych.

Wniosek. *Dla każdego* $1 \leqslant \alpha < c$ *i* $\gamma \log n < d$, $\mathsf{NN}_{\text{wfn}}(c,d)$ *może być zredukowane do* $\mathcal{O}(d/(\gamma \log n))$ *instancji* $\mathsf{NN}_{\text{wfn}}(\alpha, \gamma \log n)$, *gdzie* $\gamma < \frac{2(1-\nu)}{(\frac{\alpha}{c})^2 - 1 - 2\log\frac{\alpha}{c}}$ *dla zadanego parametru* $\nu \in [0,1)$ *i:*

$$\operatorname{query}(c,d) = \mathcal{O}(d^2 + n^\nu + d/(\gamma \log n)\ \operatorname{query}(\alpha, \gamma \log n)),$$
$$\operatorname{preproc}(c,d) = \mathcal{O}(d^{\omega-1}n + d/(\gamma \log n)\ \operatorname{preproc}(\alpha, \gamma \log n)).$$

Jeżeli zapytania są dostarczane w paczkach o wielkości d, *to złożoność obliczeniowa zapytania wynosi:*

$$\operatorname{query}(c,d) = \mathcal{O}(d^{\omega-1} + n^\nu + d/(\gamma \log n)\ \operatorname{query}(\alpha, \gamma \log n)).$$

Parametr ν pozwala nam osiągnąć różne kompromisy pomiędzy złożonością czasu zapytania a wymiarem przestrzeni wynikowej. Jeśli $\nu = 0$, złożoność

obliczeniowa zapytania (część zależna od algorytmu redukcji wymiaru) nie zależy od n. Jeśli ν jest zbliżony do 1, czas zapytania jest prawie liniowy ze względu na n, a wymiar przestrzeń wynikowej jest stały. Powyższy wniosek będzie przez nas używany do budowania wydajnych algorytmów dla NN_{wfn}.

W pracy pokazujemy kolejne redukcje, które pozwalają nam rozluźnić ograniczenie dla $c = \omega(\sqrt{\log \log n})$. Rozszerzamy redukcję, korzystając z wielu rodzin funkcji haszujących. Prowadzi to do rozwiązania interesującego podproblemu c-najbliższych sąsiadów w $(\mathbb{R}^k)^L$, dla iloczynu l_∞–product$(x) \coloneqq \max_{1 \leqslant i \leqslant L} \|x_i\|_2$ i indukowanej metryki. Ta norma jest znana w literaturze i została oznaczona jako $max - produkt$ lub l_∞–product. Pokazujemy, że szukanie najbliższych sąsiadów l_∞–product może być rozwiązane za pomocą odpowiedniej rodziny funkcji LSH. Ponadto, aby pokazać właściwości funkcji haszujących, badamy szacowania miar anty-koncentracji (ang. anti–concentration bounds), co prowadzi do ciekawych problemów geometrycznych. W szczególności pokazujemy optymalne, z dokładnością do małej stałej, oszacowanie dla pola powierzchni czaszy kuli. Dowodzimy, że dla x, y będących losowymi punktami z $\mathbb{S}^{(d-1)}$, $\mathbb{P}\left[|\langle x, y\rangle| < \alpha\right]$ można interpretować jako pole powierzchni czaszy kuli. Pokazujemy, że $\mathbb{P}\left[|\langle x, y\rangle| < \alpha\right] \leqslant \alpha\sqrt{d}$ i twierdzimy, że ta nierówność jest optymalna z dokładnością do stałego czynnika.

W *podejściu zliczającym* przeglądamy wszystkie punkty i rozważamy NN_{wfn} w domenie całkowitoliczbowej. W domenie całkowitoliczbowej istnieje skończona liczba punktów, które mogą być sąsiadami punktów ze zbioru wejściowego. Naiwny algorytm będzie przechowywał wszystkich sąsiadów w słowniku. Zapytanie sprowadza się wtedy do pobrania właściwej odpowiedzi ze struktury danych. Mimo że przechowywanie wszystkich możliwych rozwiązań jest wykładnicze ze względu na wymiar przestrzeni, to dzięki redukcji wymiaru możemy uzyskać algorytm wielomianowy, co umożliwia nam konstruowanie algorytmu dla dowolnego $c > 1$ dla ℓ_2. Używając standardowych rzutowań pomiędzy normami ℓ_p, możemy uzyskać wyniki dla dowolnego $p \in [1, \infty]$. Zastosowanie podejścia zliczającego daje najlepsze wyniki dla NN_{wfn}, które są przedstawione w tej rozprawie.

	Wstępne przetwarzanie	**Zapytania**
szybkie przetw. wstęp. $c < 2$ (*)	$\mathcal{O}(d^{\omega-1}n)$	$\mathcal{O}(d^{\omega-1} + dn^{\frac{1}{1+\mathcal{O}(\epsilon^2 \log^{-1}\frac{1}{\epsilon})}})$
szybkie zapytanie $c < 2$ (**)	$n^{1+\mathcal{O}(\frac{1-\nu}{\epsilon^2}\log\frac{1}{\epsilon})}$	$\mathcal{O}(d^{\omega-1} + n^{\nu})$
szybkie przetw. wstęp. $c \geqslant 2$ (*)	$\mathcal{O}(d^{\omega-1}n)$	$\mathcal{O}(d^{\omega-1} + dn^{\mathcal{O}(\frac{1}{\log c})})$
szybkie zapytanie $c \geqslant 2$ (**)	$\mathcal{O}(d^{\omega-1}n + dn^{1+\mathcal{O}(\frac{1-\nu}{\log c})}/\log n)$	$\mathcal{O}(d^{\omega-1} + n^{\nu})$

Tabela 1: Złożoności algorytmu dla NN_{wfn} dla metody zliczającej dla $p = 2$. Mamy $\epsilon = c - 1$ i ν parametr wejściowy.

Podsumowanie wyników

Tabela 1 podsumowuje wyniki dla metody zliczającej dla $p = 2$.[4] Rozróżniamy przypadki dla $c \geqslant 2$ i $c < 2$ oraz szybkiego zapytania i szybkiego wstępnego przetwarzania. W wersji z szybkim wstępnym przetwarzaniem złożoność obliczeniowa wstępnego przetwarzania wynosi $\mathcal{O}(nd^{\omega-1})$ i jest bardzo zbliżona do optymalnego czasu równego $\mathcal{O}(nd)$. W wersji z szybkim zapytaniem przedstawiamy złożoność w zależności od parametru ν. Gdy $\nu = 0$, złożoność zapytania jest równa $\mathcal{O}(d^{\omega})$, i jest zbliżona do optymalnej wartości $\mathcal{O}(d)$.

Wyniki przedstawione w Tabeli 1 dla $c \geqslant 2$ można uogólnić na dowolne $p \in [1, \infty]$, co jest przedstawione w Tabeli 2. Jeśli $p = 2$, otrzymamy wyniki tożsame z tymi przedstawionymi w Tabeli 1. Wyniki te stopniowo się pogarszają kiedy p oddala się od 2. Jednakże, dla $p \neq 2$, dla wersji szybkiego przetwarzania wstępnego, złożoność przetwarzania wstępnego jest nadal bliska optymalnej. Analogiczne twierdzenie jest prawdziwe również dla wersji z szybkim zapytaniem, złożoność obliczeniowa zapytania jest bliska optymalnej.

Nasze algorytmy są porównywalne z najlepszymi algorytmami z gwarancjami Monte Carlo. W szczególności, w wersji z szybkim czasem przetwarzania wstępnego, czas przetwarzania wstępnego jest równy $\mathcal{O}(nd^{\omega-1})$ a czas zapytania jest równy $\mathcal{O}(d^{\omega-1} + dn^{1-\mathcal{O}(\epsilon^2 \log^{-1}\frac{1}{\epsilon})})$. Podczas gdy najlepsze wyniki z

[4] W podsumowaniu skupiamy się na *metodzie zliczającej* ponieważ daje lepsze wyniki niż *LSH*.

	Wstępne przetwarzanie	**Zapytania**
szybkie wstępne przetw. $c \geqslant 2d^{\lvert 1/2-1/p \rvert}$ (*)	$\mathcal{O}(d^{\omega-1}n)$	$\mathcal{O}(d^{\omega-1} + dn^{\mathcal{O}(\frac{1}{\alpha})})$
szybkie zapytanie $c \geqslant 2d^{\lvert 1/2-1/p \rvert}$ (**)	$\mathcal{O}(d^{\omega-1}n + dn^{1+\mathcal{O}(\frac{1-\nu}{\alpha})}/\log n)$	$\mathcal{O}(d^{\omega-1} + n^{\nu})$

Tabela 2: Wyniki dla techniki zliczającej $p \in [1, \infty]$. $\alpha = \log c - |1/2 - 1/p| \log d$.

gwarancjami Monte Carlo [11] osiągają czas zapytania$\mathcal{O}(d + n^{1-\mathcal{O}(\epsilon^2)+o(1)})$ oraz czas wstępnego przetwarzania $\mathcal{O}(dn + n^{1+o(1)})$.

W wersji z szybkim czasem zapytania nasz algorytm z gwarancjami typu Las Vegas ma złożoność zapytania równą $\mathcal{O}(d^{\omega-1})$ oraz złożoność wstępnego przetwarzania równą $n^{\mathcal{O}(1/\epsilon^2 \log 1/\epsilon)}$. Z kolei najlepsze wyniki z gwarancjami typu Monte Carlo [11, 10] mają czas zapytania równy $n^{o(1)}$ oraz czas wstępnego przetwarzania równy $n^{\mathcal{O}(1/\epsilon^2)+o(1)}$ dla [11] lub czas zapytania równy $\mathcal{O}(d \log d + \log^3 n)$ oraz czas wstępnego przetwarzania równy $n^{\mathcal{O}(1/\epsilon^2)}$ dla [10].

Część prezentowanych wyników była efektem pracy zbiorowej i została wcześniej opublikowana. Wyniki dla $c = \Omega(\sqrt{d})$ były wynikiem zbiorowej pracy i zostały opublikowane w [1]. Ulepszone wyniki dla $c = \Omega(\sqrt{\log \log n})$ zostały opublikowane w [2]. Wszystkie pozostałe algorytmy i dowody algorytmów dla dowolnych $c > 1$ nie zostały do tej pory opublikowane. Wyniki te nie tylko osłabiają ograniczenia na c, ale także poprawiają złożoność algorytmów.

Sieci społecznościowe

W tym dziale rozważamy problem najbliższych sąsiadów z innej perspektywy. Badamy sieci społecznościowe w kontekście rozpowszechniania informacji lub alternatywnie rozpowszechniania się plotek. W tym kontekście sąsiedzi nie muszą być do siebie podobni, są natomiast związani w następujący sposób. Sąsiadem użytkownika A jest użytkownik, który gotów jest rozpowszechniać informacje opublikowane przez użytkownika A. Ta relacja jest asymetryczna i nie może być bezpośrednio zredukowana do przypadku przestrzeni metrycznej przedstawionej w poprzednim dziale. Sąsiedztwem danego użytkownika nazywamy zbiór jego sąsiadów, to jest zbiór użytkowników, którzy z dużym prawdopodobieństwem będą rozpowszechniać jego wiadomości.

Typowym zastosowaniem do obliczania sąsiedztwa jest optymalizacja marketingu wirusowego. Celem jest wybranie możliwie niewielkiej liczby osób wpływających na sieć (ang. influencer), tak aby zmaksymalizować liczbę użytkowników, do których docierają treści marketingowe. W szczególności interesujące jest badanie dynamiki kaskad informacyjnych.[5] Dla danego momentu w czasie szacujemy szanse, że dana kaskada będzie bardzo duża, np. na podstawie dynamiki jej wzrostu.

Kluczowe dla obliczania sąsiedztw jest zrozumienie procesu rozpowszechniania się informacji. Modelowaniu tego procesu poświęcona jest w głównej mierze ta część pracy. Przedstawiamy wyniki, które uzyskaliśmy w modelowaniu procesu w rzeczywistych sieciach społecznościowych, jak również w grafach losowych.

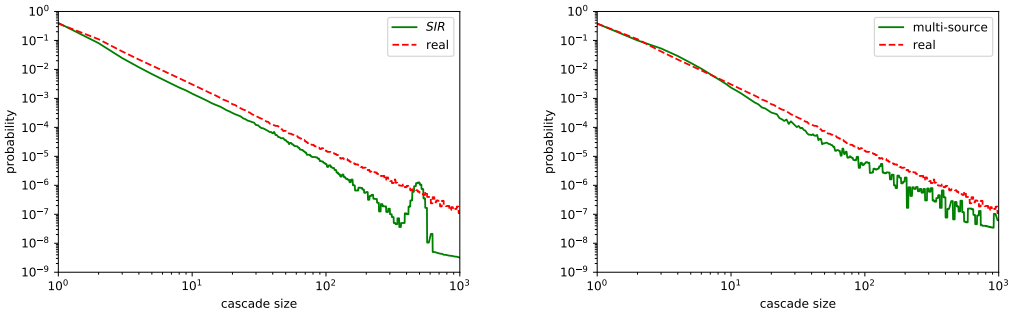
Modele w rzeczywistych sieciach społecznościowych

W tej części badamy znane modele i ich adekwatność do sieci Twittera [14]. W szczególności rozważamy model (*zainfekowany, podatny, zdrowy*) (ang. *Susceptible, Infected, Recovered – SIR*).[6] Ponadto proponujemy metodę porównywania modeli rozprzestrzeniania się plotki. Dla danego modelu wskazujemy wartość liczbową, która ocenia jak bardzo dany model różni się od stanu rzeczywistego. Argumentujemy, że właściwym sposobem porównywania różnych modeli rozpowszechniania informacji jest test Kolmogorova–Smirnova (KS test). KS test mierzy odległość między rzeczywistym rozkładem kaskad i rozkładem generowanym przez model rozpowszechniania informacji.

Ponadto proponujemy nowe modele, które lepiej pasują do rzeczywistego rozkładu wielkości kaskad: model *exp–SIR* i *model wielu źródeł*. W modelu

[5] Kaskada składa się ze wszystkich użytkowników, którzy rozpowszechnili daną informacje. Kaskada jest pewnym przybliżeniem sąsiedztwa, sąsiad użytkownika A, to użytkownik, który często znajduję się w kaskadach które rozpoczynają się w A.

[6] W tym modelu plotka zaczyna się od jednego zainfekowanego węzła. Każdy zainfekowany węzeł może zainfekować każdego ze swoich podatnych obserwatorów z pewnym stałym prawdopodobieństwem, a następnie przechodzi do stanu zdrowego. Proces powtarza się, aż wszystkie zainfekowane węzły zmienią się w zdrowe. Istnieją różne warianty tego modelu. W modelu podstawowym każdy węzeł może infekować dowolny inny węzeł. Rozszerzenie tego algorytmu umożliwia infekowanie tylko węzłów, które są połączone w danym grafie. Inne warianty tego algorytmu definiują różne reguły infekcji. Zainfekowany węzeł może na przykład zainfekować każdego z podatnych sąsiadów niezależnie z pewnym stałym prawdopodobieństwem. W innym podejściu, zainfekowany węzeł może zarazić wszystkich podatnych sąsiadów naraz z pewnym stałym prawdopodobieństwem.



(a) SIR vs. Real

(b) Multi source vs. Real

Rysunek 1: Porównanie rozkładów wielkości kaskad dla różnych modeli. W modelu SIR obserwujemy nadreprezentację kaskad większych niż 1000.

exp–SIR uwzględniamy fakt, że bliski związek z osobą która wytworzyła informację zwiększa szansę rozpowszechnienia tej informacji. W *modelu wielu źródeł* zakładamy, że informacje mogą być rozpowszechniane poza badaną siecią: przez kontakt osobisty, telewizje, radio, gazety, telefon itp. Pokazujemy, że oba modele unikają problemu nadreprezentacji dużych kaskad, który był obecny w poprzednich modelach oraz że rozkład kaskad generowany przez te modele jest znacznie bliższy rzeczywistemu (zob. Rysunek 1.). Podsumowanie wyników testu KS dla różnych modeli znajduje się w Tabeli 3.

Tabela 3: Wyniki testu KS pomiędzy rzeczywistym rozkładem wielkości kaskad a rozkładem uzyskanym w danym modelu.

Model	K-S test
SIR	0.0447
exp–SIR	0.0207
multi-source	0.0116

Rozpowszechnianie informacji w grafach losowych.

W dalszej części pracy badamy teoretyczne właściwości znanych procesów rozpowszechniania informacji w grafach losowych. Rozważamy acykliczny skierowany graf Erdősa-Rényi [15] i pokazujemy podstawowe właściwości

modelu SIR w tym grafie. Acykliczny graf Erdősa-Rényi jest konstruowany w następujący sposób. Załóżmy, że wierzchołki grafu są oznaczone liczbami naturalnymi od 1 do n. Skierowany acykliczny graf Erdősa-Rényi jest losowym grafem, dla którego każda krawędź (i, j) dla $i < j$ jest próbkowana niezależnie z prawdopodobieństwem p. Za pomocą tego mechanizmu próbkowania możemy wytworzyć dowolny skierowany graf acykliczny. Analizujemy podstawowy model rozpowszechniania informacji w skierowanych acyklicznych grafach Erdősa-Rényiego. Pokazujemy, że te modele spełniają podstawową własność rzeczywistych kaskad: rozkład rozmiarów kaskad spełnia prawo potęgowe. Ponadto pokazujemy podstawowe właściwości rozważanych grafów losowych. W poniższym twierdzeniu wykazujemy, że rozkład stopni w grafach Erdősa-Rényiego jest w przybliżeniu jednostajny.

Twierdzenie. *Załóżmy, że X to rozkład stopni w n–wierzchołkowym skierowanym acyklicznym grafie Erdős–Rényiego z prawdopodobieństwem krawędzi p, wtedy:*

1. $\mathbb{P}[X = l] \leqslant \frac{1}{pn}$,
2. $\mathbb{P}[X = l] \geqslant \frac{1}{2pn}$ *dla* $l < p(n-1)$,
3. $\mathbb{P}[X = l] \leqslant \frac{\exp(-\frac{\epsilon^2}{2+\epsilon}(n-1)p)}{pn}$ *dla* $l = (1+\epsilon)p(n-1)$ *i* $\epsilon > 0$,
4. $\mathbb{P}[X = l] \geqslant \frac{1-\exp(-\frac{\epsilon^2}{2}(n-1)p)}{pn}$ *dla* $l = (1-\epsilon)p(n-1)$ *i* $1 > \epsilon > 0$.

Rozkład stopni jest kluczową właściwością charakteryzującą graf.

Najważniejszym wynikiem tej części rozprawy jest scharakteryzowanie rozkładu kaskad w losowo skierowanym acyklicznym grafie Erdősa-Rényiego. Niech $s_{n,k}$ będzie prawdopodobieństwem, że rozmiar kaskady w modelu SIR wynosi k, przy założeniu, że kaskada zaczyna się od losowego wierzchołka w n–wierzchołkowym skierowanym acyklicznym grafie Erdős-Rényiego. Poniższe twierdzenie daje asymptotyczne wartości $s_{n,k}$ dla dużych n

Twierdzenie. *Niech p będzie parametrem skierowanego acyklicznego grafu Erdősa-Rényiego, α niech będzie parametrem modelu SIR oraz $\beta = 1 - \alpha p$, wtedy $s_{n,k} \sim (n(1-\beta^k))^{-1}$.*

Dla małych αp powyższa wartość jest w przybliżeniu równa $(n\alpha pk)^{-1}$, co implikuje rozkład potęgowy.

Wyniki przedstawione w tej części rozprawy stanowią wynik pracy zbiorowej. Wyniki dotyczące modelowania rozpowszechniania informacji na Twitterze zostały przedstawione w [3]. Wyniki dotyczące rozkładu kaskad w grafach losowych zostały przedstawione w [4].

Artykuły zawarte w rozprawie doktorskiej

[1] Andrzej Pacuk, Piotr Sankowski, Karol Wegrzycki, and Piotr Wygocki. Locality-sensitive hashing without false negatives for ℓ_p. In *COCOON*, volume 9797 of *Lecture Notes in Computer Science*, pages 105–118. Springer, 2016.

[2] Piotr Sankowski and Piotr Wygocki. Approximate nearest neighbors search without false negatives for ℓ_2 for $c > \sqrt{\log \log n}$. In *ISAAC*, volume 92 of *LIPIcs*, pages 63:1–63:12, 2017.

[3] Andrzej Pacuk, Piotr Sankowski, Karol Wegrzycki, and Piotr Wygocki. There is something beyond the twitter network. In *HT*, pages 279–284. ACM, 2016.

[4] Karol Wegrzycki, Piotr Sankowski, Andrzej Pacuk, and Piotr Wygocki. Why do cascade sizes follow a power-law? In *WWW*, pages 569–576. ACM, 2017.

Literatura

[5] Piotr Indyk and Rajeev Motwani. Approximate nearest neighbors: Towards removing the curse of dimensionality. In *Proceedings of the Thirtieth Annual ACM Symposium on Theory of Computing*, STOC '98, pages 604–613, New York, NY, USA, 1998. ACM.

[6] Ryan Williams. A new algorithm for optimal 2-constraint satisfaction and its implications. *Theor. Comput. Sci.*, 348(2):357–365, December 2005.

[7] Alexandr Andoni and Piotr Indyk. Near-optimal hashing algorithms for approximate nearest neighbor in high dimensions. *Commun. ACM*, 51(1):117–122, 2008.

[8] Ryan O'Donnell, Yi Wu, and Yuan Zhou. Optimal lower bounds for locality-sensitive hashing (except when q is tiny). *ACM Trans. Comput. Theory*, 6(1):5:1–5:13, March 2014.

[9] Alexandr Andoni and Ilya Razenshteyn. Optimal data-dependent hashing for approximate near neighbors. In *Proceedings of the Forty-Seventh Annual ACM on Symposium on Theory of Computing, STOC 2015, Portland, OR, USA, June 14-17, 2015*, pages 793–801, 2015.

[10] Nir Ailon and Bernard Chazelle. The fast johnson-lindenstrauss transform and approximate nearest neighbors. *SIAM J. Comput.*, 39(1):302–322, May 2009.

[11] Alexandr Andoni, Thijs Laarhoven, Ilya Razenshteyn, and Erik Waingarten. Optimal hashing-based time-space trade-offs for approximate near neighbors. In *Proceedings of the Twenty-Eighth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '17, pages 47–66, Philadelphia, PA, USA, 2017.

[12] Piotr Indyk. On approximate nearest neighbors in non-euclidean spaces. In *39th Annual Symposium on Foundations of Computer Science, FOCS '98, November 8-11, 1998, Palo Alto, California, USA*, pages 148–155, 1998.

[13] William Johnson and Joram Lindenstrauss. Extensions of Lipschitz mappings into a Hilbert space. In *Proceedings of the conference in modern analysis and probability (New Haven, Conn., 1982)*, volume 26 of *Contemporary Mathematics*, pages 189–206. American Mathematical Society, 1984.

[14] twitter.com. Twitter api. `https://dev.twitter.com/rest/public`, 2016.

[15] P. Erdős and A Rényi. On the evolution of random graphs. In *Publication of the mathematical institute of the hungarian academy of sciences*, pages 17–61, 1960.