Wyszukiwanie podobnych podgrafów przy pomocy statystyki chi-kwadrat

Maciej Brzeski

Seminarium: Matematyka dyskretna

10 czerwca 2021

Neighbor-Aware Search for Approximate Labeled Graph Matching using the Chi-Square Statistics (2017)

Sourav Dutta, Pratik Nayek, Arnab Bhattacharya

Wstęp

- Grafy etykietowane pozwalają reprezentować wiele danych pojawiających się w rzeczyswistości - związki chemiczne, grafy wiedzy
- Mając zapytanie w formie grafu oraz zbiór grafów (w postaci osobnych grafów, lub jednego, dużego) chcemy znaleźć grafy (podgrafy), które są mu najbliższe
- Podejścia związane z izomorfizmem lub graph matching'iem są wolne i nie mają zastosowania dla dużych zbiorów danych
- W konsekwencji chcemy znaleźć metodę, która tylko w sposób przybliżony będzie oceniała podobieństwo grafów

Zastosowanie

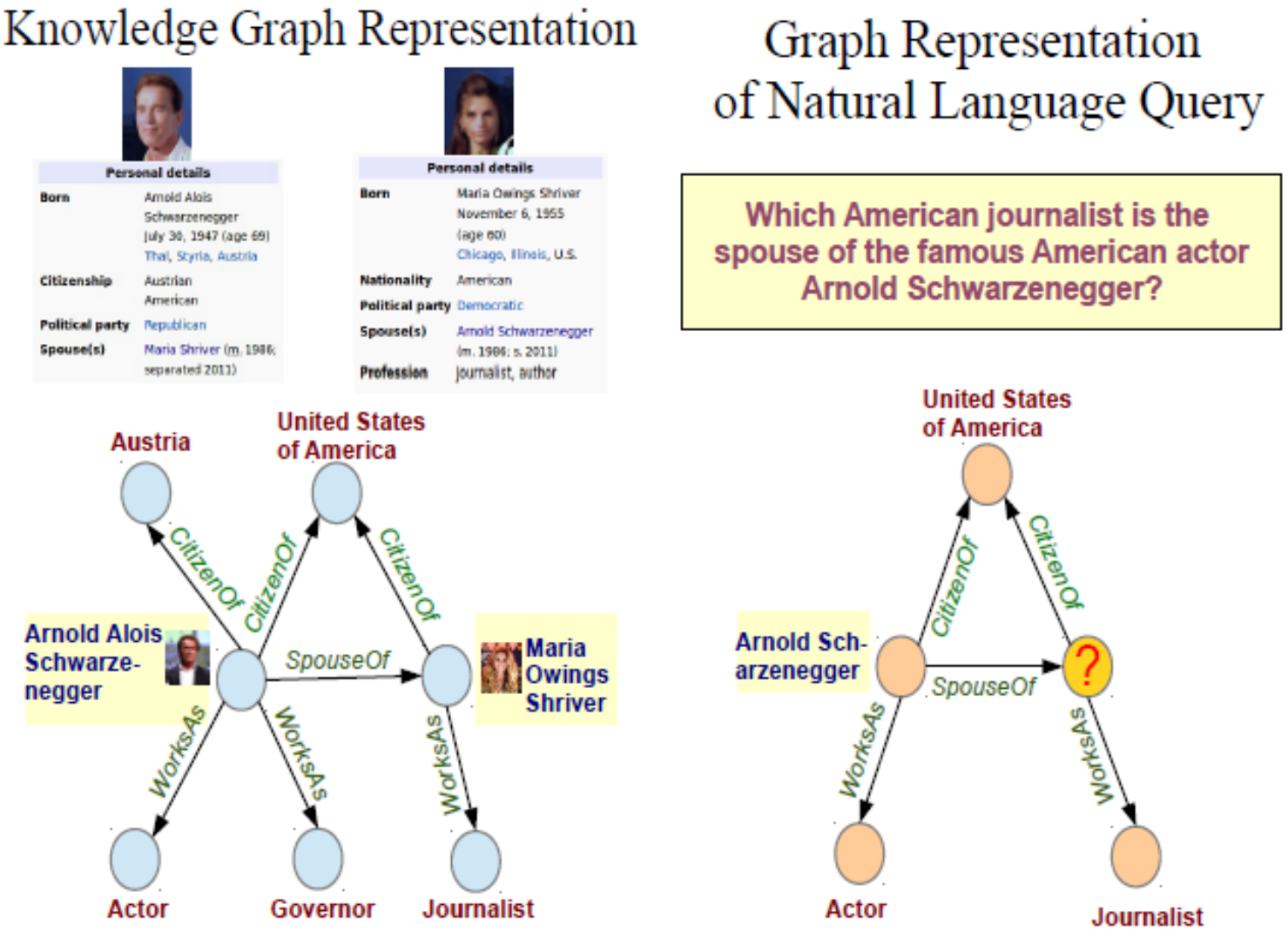


Figure 1: Approx. subgraph matching for question answering.

Motywacja dla istotności statystycznej

- Stopień podobieństwa strukturalnego jest kluczowy dla przybliżonego dopasowania grafów. W porównywanych pracach róbowano to uchwycić przez dodawanie różnych cech, jak liczba wierzchołków z daną etykietą, liczbą krawędzi, rozkładem stopni wierzchołków, wskaźniki centralności, średnica.
- Z drugiej strony w taki sposób nie jesteśmy wstanie uchwycić relacji etykiet. Zakładając jednostajny rozkład etykiet, prawdopodobieństwo, że w danym regionie grafu większość etykiet będzie identyczna jest dość niskie, co może być istotną informacją
- Autorzy wspominają, że takie podejście pozwala również uwzględnić częstotliwość występowania danych etykiet, ale w pracy zakładają jedynie jednostajny rozkład etykiet

Test zgodności chi-kwadrat

Statystyka χ^2 liczy róznicę między oczekiwaną i zaobserwowaną wielkością zdarzeń. W przypadku zliczeń przedstawia się wzorem:

$$\chi^2 = \sum_{\forall i} \frac{(O_i - E_i)^2}{E_i}, \tag{1}$$

gdzie O_i jest liczbą zaobserwowanych zdarzeń, a E_i ich wartościami oczekiwanymi.

Definicja Problemu

Definition

Mając dany nieskierowany graf G i graf zapytania Q, gdzie wierzchołki obu grafów są etykietowane należy znaleźć k podgrafów G najbardziej podobnych do Q.

Definicja Problemu

Definition

Mając dany nieskierowany graf G i graf zapytania Q, gdzie wierzchołki obu grafów są etykietowane należy znaleźć k podgrafów G najbardziej podobnych do Q.

Autorzy publikacji, w skrócie, do problemu podeszli w następujący sposób:

- zaproponowano aproksymacyjny algorytm match'ujący, używający statystyki chi-kwadrat do uwzględnienia zarówno etykiet, jak i cech strukturalnych grafu
- przedstawiono dwa algorytmy: VELSET i NAGA. Pierwszy z nich jest bardziej ogólny i efektywny dla danych zaszumionych. Drugi z nich znacznie bardziej ogranicza liczbę przeglądanych wierzchołków, przez co jest szybszy
- przeprowadzono testy na danych rzeczywistych porównując zarówno efektywność jak i czas wykonania do aktualnie uznanych algorytmów state-of-the-art

Szkielet algorytmu

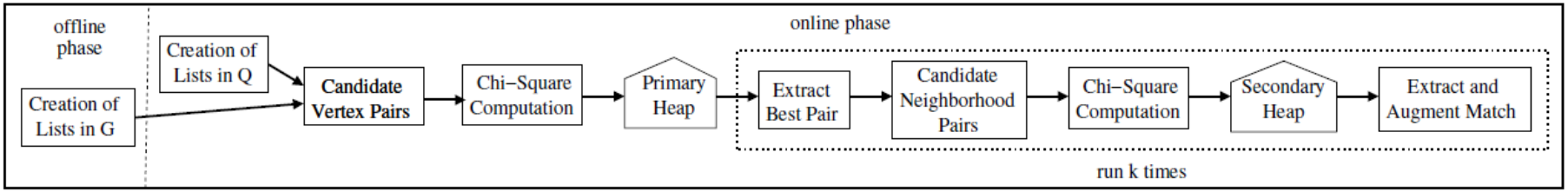


Figure 3: Algorithmic framework overview for VELSET and NAGA.

- W fazie off-line, mając tylko graf G liczymy mapowanie etykiet na wierzchołki i listę etykiet sąsiadów. Gdy pojawi się zapytanie grafowe Q robimy analogiczne obliczenia. Jeśli Q jest małe, obliczenia list są mało kosztowne.
- Następnie znajdujemy potencjalne pary wierzchołków, które mogą być dopasowane do siebie. Dla każdej pary obliczamy statystykę chi-kwadrat. Im wyższa, tym większa szansa, że para jest dobrze dopasowana.
- Dla wybranych wierzchołków szukamy w ich sąsiedztwie kolejnych par, dla których wykonujemy analogiczne obliczenia. Proces kontynuujemy do dopasowania wszystkich wierzchołków z Q.

VELSET

Pirwszą propozycją jest *VErtex Label Similarity on Edge Triplets (VELSET)*

VELSET

Pirwszą propozycją jest *VErtex Label Similarity on Edge Triplets (VELSET)*

Preprocessing - Pierwszym krokiem jest policzenie odwróconego mapowania dla etykiet (dla etykiety chcemy otrzymać listę wierzchołków), oznaczoną przez IL^G oraz IL^Q. Następnie wyznaczamy listę etykiet sąsiedztwa (oznaczoną przez LNL). Przez LNL_u^G i LNL_v^Q oznaczamy listy etykiet dla wierzchołków $u \in G$, $v \in Q$.

Konstrukcja zbioru par

Dla każdej etykiety w grafie Q wyszukujemy podobne etykiety w G i wszystkie odpowiadające im wierzchołki.

$$VP = \{\langle u, v\rangle | u \in G \wedge v \in Q \wedge l_u^Q \simeq l_v^Q\}, \tag{2}$$

W pracy do określenia podobieństwa etykiet została użyta metryka Jaccarda: $JS(s,t) = \frac{|s \cap t|}{|s \cup t|}$ (gdzie $|s \cap t|$ oznacza liczbę wspólnych liter, a $|s \cup t|$ liczbę liter w obu wyrazach).

Istotność statystyczna

Następnie musimy wybrać najbardziej znaczącą parę ze zbioru VP. Dla każdej pary liczymy statystykę chi-kwadrat bazując na „trójkach etykiet”

Istotność statystyczna

Następnie musimy wybrać najbardziej znaczącą parę ze zbioru VP. Dla każdej pary liczymy statystykę chi-kwadrat bazując na „trójkach etykiet"

Generowanie trójek - trójkę wierzchołków o środku u nazywamy dowoloną trójkę $\langle x, u, y \rangle$, gdzie x i y sąsiadują z u. Dla każdej możliwej trójki liczmy odpowiadające im trójki etykiet $\langle I_x^G, I_u^G, I_y^G \rangle$. Do obliczeń zakładamy, że kolejność etykiet w danej trójce nie ma znaczenia.

Stopień nakładania się etykiet - Dla pary wierzchołków $\langle u, v \rangle$ trójki *vinQ* match'ujemy z trójkami wygenerowanymi przez $u \in G$. Rozważmy etykiety $\langle l_x^G, l_u^G, l_y^G \rangle$ i $\langle l_a^Q, l_v^G, l_b^Q \rangle$ dla $(u, v) \in VP$ o środku w u i v. Bazując na podobieństwie etykiet, stopień podobieństwa trójek klasyfikujemy na 3 grupy:

- s_2 : obie etykiety sąsiednich wierzchołków są podobne

$$s_2 : (l_x^G \simeq l_a^Q \wedge l_y^G \simeq l_b^Q) \tag{3}$$

- s_1 : jedna z etykiet sąsiednich wierzchołków jest podobna

$$s_1 : (l_x^G \simeq l_a^Q \wedge l_y^G \nsim l_b^Q) \vee (l_x^G \nsim l_a^Q \wedge l_y^G \simeq l_b^Q) \tag{4}$$

- s_0 : żadna z etykiet sąsiednich wierzchołków nie jest podobna

$$s_0 : (l_x^G \nsim l_a^Q \wedge l_y^G \nsim l_b^Q) \tag{5}$$

Dla każdej trójki próbujemy uzyskać najwyższe podobieństwo. Następnie dopasowywany wierzchołek jest reprezentowany przez ciąg symboli s_i, O, odpowiadający dopasowaniu trójek.

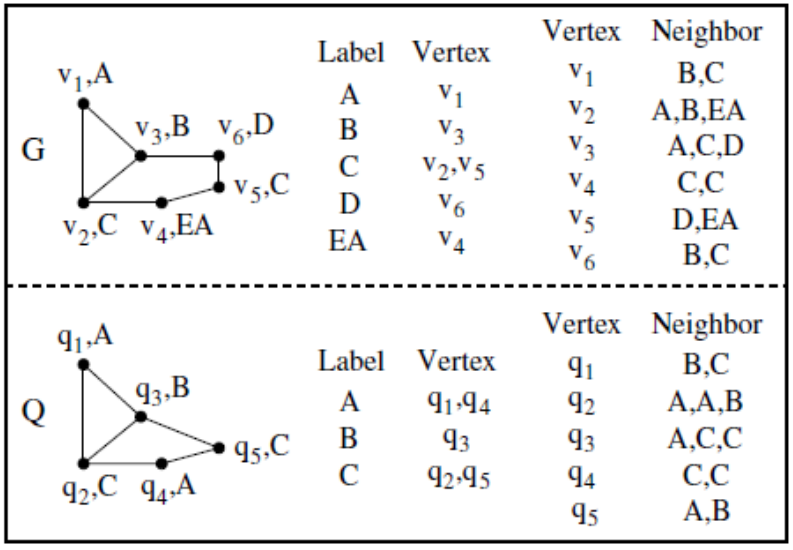


Figure 2: Running example: Input graph G and query Q.

Rozważmy parę $\langle v_3, q_3 \rangle$ z naszego przykładu. Dla wierzchołka q_3 mamy trzy trójki etykiet: $\{A, B, C\}$, $\{C, B, C\}$ oraz $\{C, B, A\}$. Ich najlepsze dopasowania to odpowiadnio $\{A, B, C\}$, $\{C, B, D\}$ i $\{D, B, A\}$. W pierwszym przypadku zgadzają się obie etykiety, w dwóch kolejnych jedna - zatem ta para wierzchołków jest reprezentowana przez ciąg s_2, s_1, s_1

Oczekiwana liczba wystąpień symbolu - wyliczony ciąg symboli O_u określa istotność dopasowania wierzchołka $u \in G$ Do policzenia statystyki χ^2 potrzebujemy wyznaczyć prawdopodobieństwa pojawienia się poszczególnych symboli.

Dla uproszczenia autorzy założyli, że mamy L grup etykiet o jednakowej szansie pojawienia się (ponadto, że w danej grupie wszystkie są do siebie podobne). Szansa, że dany sąsiad wierzchołka $u \in G$ nie jest podobny do wierzchołka $v \in Q$ jest równa $(1 - \frac{1}{L})$. Zakładając niezależność przypisanych etykiet prawdopodobieństwo, że nie jest podobny żaden sąsiad wynosi $(1 - \frac{1}{L})^d$, gdzie $d = deg(u)$. Wówczas szansa, że obie etykiety z trójki nie mają żadnych podobnych w sąsiedztwie wierzchołka u wynosi:

$$P(s_0) = ((1 - \frac{1}{L})^d)^2 \tag{6}$$

Z kolei prawdopodobieństwo na uzyskanie jednego dopasowanego wierzchołka wynosi:

$$P(s_1) = 2 \cdot (1 - (1 - \frac{1}{L})^d) \cdot ((1 - \frac{1}{L})^d), \tag{7}$$

a prawdopodobieństwo na uzyskanie obu dopasowanych:

$$P(s_2) = (1 - (1 - \frac{1}{L})^d)^2 \tag{8}$$

Zuważmy na koniec, że długość ciągu znaków s jest równa możliwej liczby trójek dla $v \in Q$ - $len_v = \binom{deg(v)}{2}$

Statystyka chi-kwadrat - mnożąc powyższe prawdopodobieństwa przez długość ciągu len_v otrzymujemy oczekiwane wartości pojawienia się symboli s_i. Biorąc ich zaobserwowaną liczbę z aktualnego dopasowania wyliczamy statystykę $\chi 2$.

Do zamodelowania zarówno podobieństwa etykiet, jak i strukturalnego, policzoną statystkę chi-kwadrat mnożmy przez podobieństwo etykiet. Do głównego kopca wstawiamy trójkę:

$$\langle u, v, (\chi^2_{u,v} \times JS(l_u, l_v)) \rangle \tag{9}$$

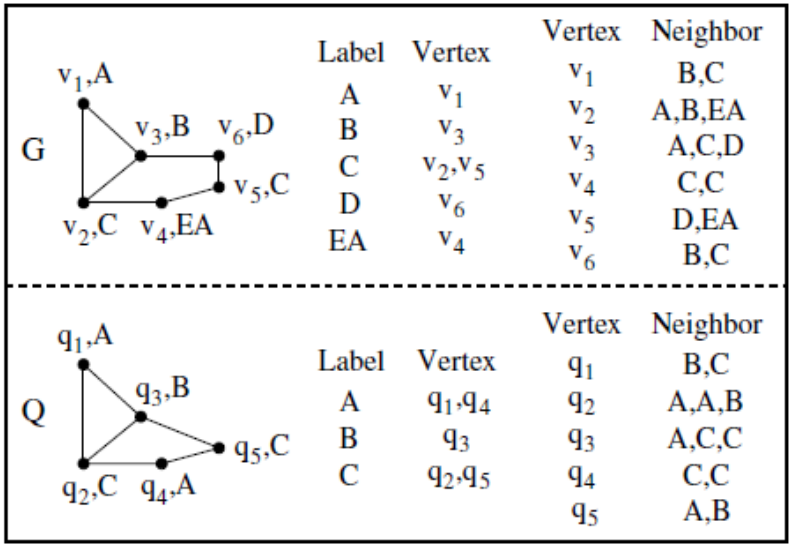


Figure 2: Running example: Input graph G and query Q.

Wracając do przykładowej pary $\langle v_3, q_3 \rangle$ mamy ciąg symboli s_2, s_1, s_1. Wobec tego $O[I_{v_3}] = [0, 2, 1]$. Biorąc $L = 4$ i $d = deg(q_3) = 3$ oraz wyliczone prawdopodobieństwa dla v_3 otrzymujemy wartości oczekiwane $E[I_{v_3}] = 3 \cdot [0.178, 0.488, 0.334]$. Wobec tego ostatecznie otrzymujemy $\chi^2_{v_3,q_3} = 0.731$

Generowanie przybliżonego dopasowania

- Po konstrukcji głównego kopca (z podobnymi parami wierzchołków) wybieramy parę $\langle u, v \rangle$ z najwyższą statystyką χ^2.
- Tworzymy poboczny kopiec do exploracji sąsiedztwa wybranego wierzchołka. Dodajemy tam wszystkie pary wierzchołków sąsiadujące z u i v z głównego kopca. Wybieramy parę o najwyższej statystyce χ^2
- Proces powtarzamy aż znaleziony podgraf będzie rozmiaru Q (lub poboczny kopiec będzie pusty).
- Przy wyszukiwaniu kolejnych podgrafów powtarzamy proces wybierając najlepszą parę z VP, które nie była dopasowana we wcześniejszym przebiegu.

Algorithm 1 VELSET

Input: Database graph G, Query graph Q, Number of matches k
Output: Top-k subgraphs of G similar to Q

1: Compute IL^G and LNL^G of input graph G ▷ Offline
2: Compute IL^Q and LNL^Q of query graph Q
3: $VP \leftarrow \{\langle u, v\rangle \mid u \in G \wedge v \in Q \wedge l_u^G \simeq l_v^Q\}$
4: Primary max-heap $PH \leftarrow \Phi$
5: **for all** $\langle u, v\rangle \in VP$ **do**
6: $T_u, T_v \leftarrow$ triplets centered at u, v respectively
7: Initialize O_u to empty string of length $len_v = \binom{deg(v)}{2}$
8: **for all** triplet $t_v \in T_v$ **do**
9: $t_u \leftarrow$ largest overlapping match of t_v
10: Symbol $s \leftarrow$ match of t_u with t_v ▷ s is s_2, s_1 or s_0
11: Append O_u with s
12: **end for**
13: $E_u \leftarrow len_v \times [Pr(s_0), Pr(s_1), Pr(s_2)]$ ▷ Eqs. (6), (7), (8)
14: Compute $\chi^2_{\langle u,v\rangle}$ using expected E_u and observed O_u ▷ Eq. (1)
15: Insert $\langle u, v, \chi^2_{\langle u,v\rangle}.JS(l_u, l_v)\rangle$ into PH
16: **end for**
17: **for** $i = 1$ to k **do**
18: **repeat**
19: $\langle u, v\rangle \leftarrow$ Extract(PH)
20: **until** neither u nor v are marked "done"
21: `Match(i)` $\leftarrow \langle u, v\rangle$
22: Mark u and v as "done"
23: Secondary max-heap $SH \leftarrow \Phi$
24: $U', V' \leftarrow$ adjacent nodes of u, v respectively
25: Insert $\{\langle u' \in U', v' \in V', \chi^2_{\langle u',v'\rangle}.JS(l_{u'}, l_{v'})\rangle \mid l_{u'}^G \simeq l_{v'}^Q\}$ into SH
26: **while** $|$`Match(i)`$| \neq |Q|$ and $SH \neq \Phi$ **do**
27: Expand `Match(i)` by "best" unmarked vertex pair $\langle u', v'\rangle$ from SH
28: Insert unmarked vertex pairs from neighborhood of u', v' into secondary heap SH
29: **end while**
30: **end for**
31: **return** Top-k matches `Match(1), ..., Match(k)`

Analiza złożoności

- Złożoność głównie zależy od konstrukcji par podobnych wierzchołków i operacji na kopcu
- Dla grafu G o n wierzchołkach i zapytaniu Q o p wierzchołkach skonstruowanie odwrotnego mapowania zajmuje $O(p)$ (i O(n) w fazie preprocessingu)
- Zakładając, że obliczanie podobieństwa na dwóch etykietach jest w czasie stałym, złożoność konstrukcji VP wynosi $|IL^Q| \times |IL^G|$, co w pesymistycznym przypadku (gdy wszystkie etykiety są do siebie podobne) daje $v = O(n \cdot p)$.
- Każdą podobną parę dodajemy do kopca, co daje $O(v \log v)$
- Procedurę powtarzamy k razy - $O(k \cdot v \log v)$

NAGA

- algorytm VELSET używa podobieństwa etykiet, dzięki czemu radzi sobie lepiej z zaszumionymi grafami
- z drugiej strony powoduje to znacznie większe obliczenia
- jako alternatywę autorzy zaproponowali algorytm Neighbor-Aware Greedy Algorithm (NAGA)

Preprocessing - analogicznie jak dla VELSET tworzymy odwrócone mapowanie etykiet na wierzchołki oraz listę etykiet sąsiadów.

Preprocessing - analogicznie jak dla VELSET tworzymy odwrócone mapowanie etykiet na wierzchołki oraz listę etykiet sąsiadów.

Konstrukcja zbioru par - pierwszą różnicą jest to, że uwzględniamy tylko pary wierzchołków o takich samych etykietach:

$$VP = \{\langle u, v\rangle | u \in G \wedge v \in Q \wedge l_u^Q = l_v^Q\}, \tag{10}$$

co znacząco zmniejsza zbiór przeglądanych wierzchołków.

Istotność statystyczna

Kolejnym krokiem jest wybór najbardziej znaczącej pary wierzchołków. NAGA różni się od VELSET'u tym, że bierze pod uwagę etykiety wszystkich sąsiadów naraz.

Istotność statystyczna

Kolejnym krokiem jest wybór najbardziej znaczącej pary wierzchołków. NAGA różni się od VELSET'u tym, że bierze pod uwagę etykiety wszystkich sąsiadów naraz.

Wybór par wierzchołków - z wierzchołków ze zbioru VP są wybierane para, dla której przecięcie etykiet sąsiadów LNL_u^G i LNL_v^Q jest największe. Jednakże to rozważanego przecia brane są pod wagę zbiory. W rzeczywistych danych etykiety rzadko się powtarzają, więc taka redukcja jestraczej mało istotna.
Pozostałe pary zawierające jeden z wybranych wierzchołków są usuwane i każdy wierzchołek $v \in Q$ ma najwyżej jedno dopasowanie.

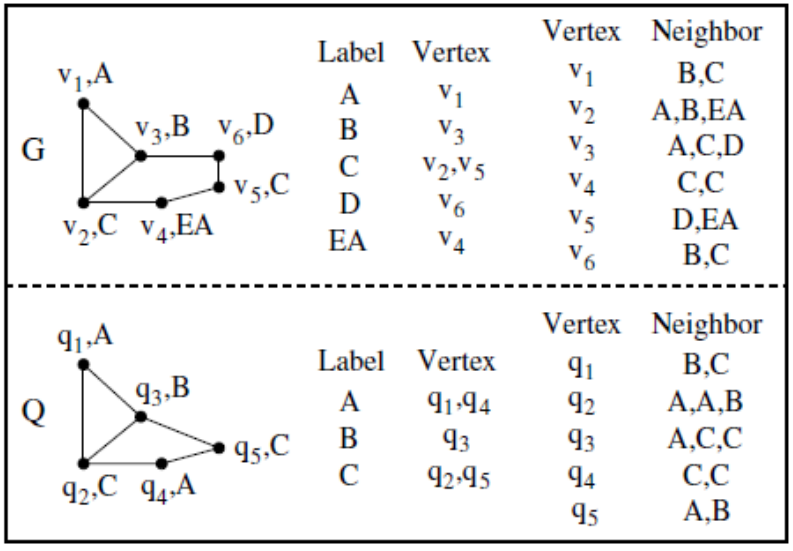


Figure 2: Running example: Input graph G and query Q.

W naszym grafie rozważmy wierzchołek q_2. Wówczas $LNL_{q_2} = \{A, A, B\}$ jest modyfikowana do zbioru $LNL'_{q_2} = \{A, B\}$. Dla grafu G 2 inne wierzchołki mają taką samą etykietę: v_2 i v_5. Ponieważ przecięcie z $LNL'_{v_2} = \{A, B, EA\}$ jest większe niż z $LNL'_{v_5} = \{D, EA\}$, q_2 jest dopasowane do v_2, i para $\langle v_2, q_2 \rangle$ jest umieszczona w głównym kopcu (a para $\langle v_5, q_2 \rangle$ zostaje usunięta z potencjalnych par).

Dopasowanie wierzchołków - Dla wybranych par chcemy policzyć statystykę χ^2. Listę etykiet w sąsiednich wierzchołkach porządkujemu leksykograficznie. Następnie listy z obu wierzchołków dzielimy oknem szerokości 2. Dla każdej pary etykiet. Rozważmy okno $l_x^G l_y^G$ z LNL_u. Szukamy najbardziej dopasowanego okna $l_a^Q l_b^Q$ z LNL_v. Podobnie jak dla algorytmu VELSET dopasowanie dzielimy na 3 kategorie:

- s_2 : obie etykiety w oknie są podobne

$$s_2 : (l_x^G \simeq l_a^Q \wedge l_y^G \simeq l_b^Q) \tag{11}$$

- s_1 : jedna z etykiet jest podobna

$$s_1 : (l_x^G \simeq l_a^Q \wedge l_y^G \nsim l_b^Q) \vee (l_x^G \nsim l_a^Q \wedge l_y^G \simeq l_b^Q) \tag{12}$$

- s_0 : żadna z etykiet nie jest podobna

$$s_0 : (l_x^G \nsim l_a^Q \wedge l_y^G \nsim l_b^Q) \tag{13}$$

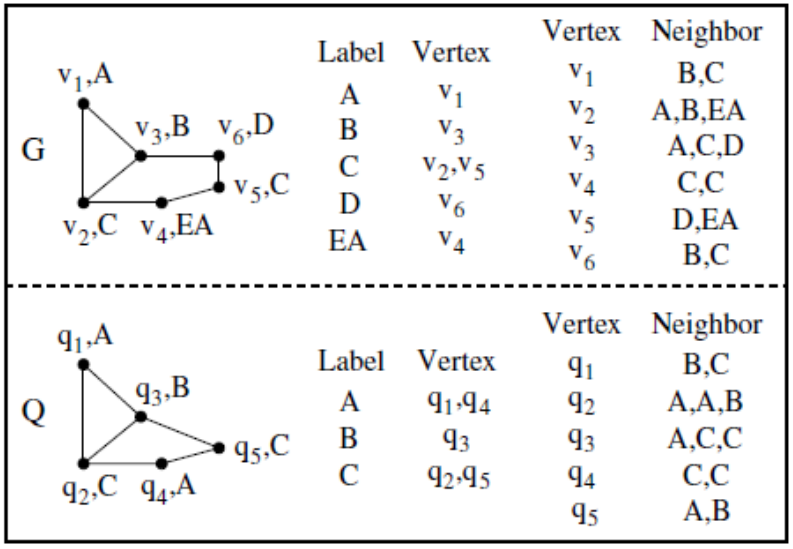


Figure 2: Running example: Input graph G and query Q.

Rozważmy dla przykładu parę $\langle v_2, q_2 \rangle$. Listy etykiet sąsiadów wyglądają odpowiednio $LNL_{v_2} = \{A, B, AE\}$ oraz $LNL_{q_2} = \{A, A, B\}$. Pierwsze okno v_2 to $\{A, B\}$ i jego najlepsze dopasowanie to drugie okno q_2 - $\{A, B\}$. Obie etykiety są identyczne, więc otrzymujemy symbol s_2. Dla drugiego okna $\{B, EA\}$ również okno $\{A, B\}$ jest najlepiej dopasowane. Tym razem jedna etykieta się zgadza, więc otrzymuujemy symbol s_1.

Oczekiwana liczba wystąpień symbolu - liczymy podobnie, jak dla algorytmu VELSET uwzględniając, że bierzemy pod uwagę tylko identyczne etykiety. Wówczas $L = |\Sigma|$ jest wielkością alfabetu nad etykietami. Dla W możliwych okien otrzymujemy:

$$P(s_0) = ((1 - \frac{1}{|\Sigma|})^W)^2 \tag{14}$$

$$P(s_1) = 2 \cdot (1 - (1 - \frac{1}{|\Sigma|})^W) \cdot ((1 - \frac{1}{L})^W) \tag{15}$$

$$P(s_2) = (1 - (1 - \frac{1}{|\Sigma|})^W)^W \tag{16}$$

Liczba okien W zależy od liczby sąsiadów porównywanego wierzchołka $v \in Q$.

Statystyka chi-kwadrat - policzone w ten sposób prawdopodobieństwa mnożomy przez długość ciągu len_v otrzymując oczekiwaną wartość występień poszczególnych symboli. Porównując z zaobserwowanymi liczbami występień liczymy statystykę χ^2. Trójka $\langle u, v, \chi^2_{u,v} \rangle$ jest wstawiana do głównego stosu jako kandydat. Procedurę powtarzamy dla każdej pary wierzchołków.

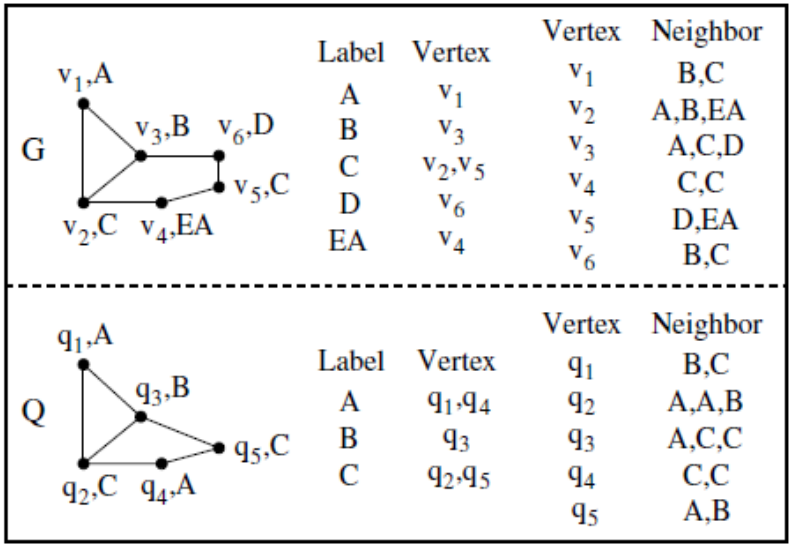


Figure 2: Running example: Input graph G and query Q.

Dla naszego przykładowego grafu, liczba symboli dla $\langle v_2, q_2 \rangle$ wynosi $O(I_{v_2}) = [0, 1, 1]$ dla odpowiednio s_0, s_1 i s_2. Używając formuł na prawdopodobieństwo pojawienia się tych symboli, oczekiwana liczba symboli ($W = 2$, $|\Sigma| = 5$) wynosi $E(I_{v_2}) = 2 \cdot [0.096, 0.4608, 0.1296]$. Ostatecznie statystyka chi-kwadrat wynosi $\chi^2_{v_2,q_2} = 2.9424$

Generowanie przybliżonego dopasowania

Wyszukiwanie całego podgrafu wygląda podobnie jak w algorytmie VELSET. Główną różnicą jest to, że wybierając wierzchołki zachłannie rozważa jedynie te o identycznych etykietach.

Algorithm 2 NAGA

Input: Database graph G, Query graph Q, Number of matches k
Output: Top-k subgraphs of G similar to Q

Compute IL^G and LNL^G of input graph G ▷ Offline
Compute IL^Q and LNL^Q of query graph Q
$VP \leftarrow \{\langle u, v\rangle \mid u \in G \wedge v \in Q \wedge l_u^G = l_v^Q\}$
Primary max-heap $PH \leftarrow \Phi$
for all $\langle u, v\rangle \in VP$ **do**
    $S_u, S_v \leftarrow$ label strings of LNL_u, LNL_v respectively
    Initialize O_u to empty string of length $len_v = |LNL_v| - 1$
    for all window $w_v \in S_v$ **do**
        $w_u \leftarrow$ largest overlapping match of w_v
        Symbol $s \leftarrow$ match of w_u with w_v ▷ s is s_2, s_1 or s_0
        Append O_u with s
    end for
    $E_u \leftarrow len_v \times [Pr(s_0), Pr(s_1), Pr(s_2)]$ ▷ Eqs. (13) – (15)
    Compute $\chi^2_{\langle u,v\rangle}$ using expected E_u and observed O_u ▷ Eq. (1)
    Insert $\langle u, v, \chi^2_{\langle u,v\rangle}\rangle$ into PH
end for
for $i = 1$ to k **do**
    repeat
        $\langle u, v\rangle \leftarrow$ Extract(PH)
    until neither u nor v are marked "done"
    `Match(i)` $\leftarrow \langle u, v\rangle$
    Mark u and v as "done"
    Secondary max-heap $SH \leftarrow \Phi$
    $U', V' \leftarrow$ adjacent nodes of u, v respectively
    Insert $\{\langle u' \in U', v' \in V', \chi^2_{\langle u',v'\rangle}\rangle \mid l_{u'}^G = l_{v'}^Q\}$ into SH
    while $|$`Match(i)`$| \neq |Q|$ and $SH \neq \Phi$ **do**
        Expand `Match(i)` by "best" unmarked vertex pair $\langle u', v'\rangle$ from SH
        Insert unmarked vertex pairs from neighborhood of u', v' into secondary heap SH
    end while
end for
return Top-k matches `Match(1), ..., Match(k)`

Analiza złożoności

- pesymistyczna złożoność jest identyczna jak w VELSET
- redukujemy liczbę rozpatrywanych par wierzchołków biorąc jedynie o identycznych etykietach
- podobieństwo wierzchołków liczymy na uporządkowanej liście etykiet (zamiast do rozpatrywania wszystkich trójek)
- dla rzeczywistych danych NAGA jest bardziej podatna na zaburzenia, ale znacznie szybsza

Brakujące etykiety i krawędzie etykietowane

- zaproponowane algorytmy są zdefiniowane na etykietowanych (wierzchołkowo) grafach
- grafy w rzeczywistych danych mają czasem brakujące eykiety. Autorzy twierdzą, że da się rozszerzyć algorytmy na tego typu grafy, początkowo dopasowywując wierzchołki z etykietami, a następnie, przy propagacji - pozwalając na dopasowanie wierzchołka bez etykiety do dowolnego innego wierzchołka (bez szczegółów)
- krawędzie z etykietami można zastąpić dodatkowym wierzchołkiem z etykietą

Eksperymenty

Eksperymenty

Zbiory danych:

- YAGO Entity Relationship Graph - zbiór danych zawierający rózne podmioty (m.in. osoby, organizacje) zebrane z Wikipedii i ich połączeń przez linki
- IMDB Network - zbiór danych zawierający połączenia między filmami, aktorami, reżyserami itp.

Generowanie zapytań - zapytania zostały wygenerowane poprzez wyodrębnienie podgrafów ze zbioru danych, a następnie poddanym:

- zaszumieniu struktury poprzez dodanie/usunięcie pewnej liczby krawędzi
- zmiany etykiet w niektórych wierzchołkach na losowe wyrazy

Ostatecznie wygenerowano 4 rodzaje zapytań:

- exact match - bez żadnych modyfikacji
- noisy edges - poddane jedynie dodaniu/usunięciu krawędzi
- noisy labels - poddane jedynie zmianie etykiet
- combined - poddane obu modyfikacjom

Metody porównawcze - do porównania użytko algorytmów NeMa: Fast graph search with label similarity (2013), i SIM-T: Using local similarity measures to efficiently address approximate graph matching (2014)

Ewaluacja

- Dla każdego wygenerowanego grafu Q oryginalny podgraf w G jest uznawany jako „ground truth”
- Szanse, że taki graf będzie pasował gdzie indziej są niewielkie. Ponadto ciężko ocenić jakość takiej odpowiedzi. Wobec tego eksperymenty przeprowadzono jedynie dla $k = 1$
- Do ewaluacji użyto metryk precision, recall i F1. Minimalne podobieństwo wierzchołków dla algorytmu VELSET ustwiono na 0,5

Table 1: Average F1 score (%) and running time results over varying graph matching queries for YAGO dataset.

Query Scenario	Quality Performance												Running Time Performance (in sec)			
	NeMa			SIM-T			VELSET			NAGA			NeMa	SIM-T	VELSET	NAGA
	P	R	F1	P	R	F1	P	R	F1	P	R	F1				
Exact Match	99.11	86.08	92.14	82.19	82.19	82.19	95.24	95.24	95.24	98.91	98.91	**98.91**	224.94	564.83	275.99	**177.58**
Noisy Edges	99.57	86.46	92.55	86.40	86.40	86.40	96.63	96.63	96.63	97.32	97.32	**97.32**	226.14	644.25	272.22	**185.70**
Noisy Labels	98.61	84.62	91.08	68.72	68.72	68.72	95.26	95.26	**95.26**	97.32	52.78	68.44	226.67	544.70	262.69	**163.96**
Combined	99.55	86.59	92.62	71.63	71.63	71.63	94.67	94.67	**94.67**	95.64	53.18	68.36	225.48	631.97	248.19	**196.70**
Average	**99.21**	85.94	92.10	77.23	77.23	77.23	95.45	**95.45**	**95.45**	97.30	75.55	85.06	225.81	596.44	264.77	**180.98**

Table 2: Average F1 score (%) and running time results over varying graph matching queries for IMDB dataset.

Query Scenario	Quality Performance												Running Time Performance (in sec)			
	NeMa			SIM-T			VELSET			NAGA			NeMa	SIM-T	VELSET	NAGA
	P	R	F1	P	R	F1	P	R	F1	P	R	F1				
Exact Match	92.70	92.31	92.50	80.29	80.29	80.29	95.95	95.95	95.95	100	100	**100**	29.43	152.75	43.32	**7.03**
Noisy Edges	91.22	90.98	91.10	79.68	79.68	79.68	98.94	98.94	98.94	100	100	**100**	29.50	170.18	42.35	**6.98**
Noisy Labels	92.65	62.16	74.40	60.31	60.31	60.31	98.90	98.90	**98.90**	100	53.73	69.90	29.33	153.02	42.79	**6.34**
Combined	91.44	58.61	71.43	63.48	63.48	63.48	97.84	97.84	**97.84**	98.33	45.92	62.61	28.98	171.76	45.31	**6.51**
Average	92.00	76.01	83.25	70.94	70.94	70.94	97.91	**97.91**	**97.91**	**99.58**	74.91	85.50	29.31	161.93	43.44	**6.71**

- **Exact Match:** Przy szukaniu identycznych grafów zarówno VELSET, jak i NAGA dają wysokie wyniki - dla identycznego wierzchołka statystyka χ^2 będzie dawała wysokie wartości. Co nie zaskakuje - NAGA jest zdecydowanie szybsza. Dla danych IMDB - NAGA zawsze znajduje szukany graf (etykiety są unikalne), stąd F1 na poziomie 100%.

Table 1: Average F1 score (%) and running time results over varying graph matching queries for YAGO dataset.

Query Scenario	Quality Performance												Running Time Performance (in sec)			
	NeMa			SIM-T			VELSET			NAGA			NeMa	SIM-T	VELSET	NAGA
	P	R	F1	P	R	F1	P	R	F1	P	R	F1				
Exact Match	99.11	86.08	92.14	82.19	82.19	82.19	95.24	95.24	95.24	98.91	98.91	**98.91**	224.94	564.83	275.99	**177.58**
Noisy Edges	99.57	86.46	92.55	86.40	86.40	86.40	96.63	96.63	96.63	97.32	97.32	**97.32**	226.14	644.25	272.22	**185.70**
Noisy Labels	98.61	84.62	91.08	68.72	68.72	68.72	95.26	95.26	**95.26**	97.32	52.78	68.44	226.67	544.70	262.69	**163.96**
Combined	99.55	86.59	92.62	71.63	71.63	71.63	94.67	94.67	**94.67**	95.64	53.18	68.36	225.48	631.97	248.19	**196.70**
Average	**99.21**	85.94	92.10	77.23	77.23	77.23	95.45	**95.45**	**95.45**	97.30	75.55	85.06	225.81	596.44	264.77	**180.98**

Table 2: Average F1 score (%) and running time results over varying graph matching queries for IMDB dataset.

Query Scenario	Quality Performance												Running Time Performance (in sec)			
	NeMa			SIM-T			VELSET			NAGA			NeMa	SIM-T	VELSET	NAGA
	P	R	F1	P	R	F1	P	R	F1	P	R	F1				
Exact Match	92.70	92.31	92.50	80.29	80.29	80.29	95.95	95.95	95.95	100	100	**100**	29.43	152.75	43.32	**7.03**
Noisy Edges	91.22	90.98	91.10	79.68	79.68	79.68	98.94	98.94	98.94	100	100	**100**	29.50	170.18	42.35	**6.98**
Noisy Labels	92.65	62.16	74.40	60.31	60.31	60.31	98.90	98.90	**98.90**	100	53.73	69.90	29.33	153.02	42.79	**6.34**
Combined	91.44	58.61	71.43	63.48	63.48	63.48	97.84	97.84	**97.84**	98.33	45.92	62.61	28.98	171.76	45.31	**6.51**
Average	92.00	76.01	83.25	70.94	70.94	70.94	97.91	**97.91**	**97.91**	**99.58**	74.91	85.50	29.31	161.93	43.44	**6.71**

- **Noisy Edges:** Dla grafów ze zmodyfikowanymi jedynie krawędziami znów NAGA radzi sobie najlepiej, i ponownie na zbiorze IMDB uzyskuje wynik 100%. W przypadku gdy etykiety są niezmienione - strategia szukania podobnych wierzchołków przez VELSET wydaje się być niepotrzebna.

Table 1: Average F1 score (%) and running time results over varying graph matching queries for YAGO dataset.

Query Scenario	Quality Performance												Running Time Performance (in sec)			
	NeMa			SIM-T			VELSET			NAGA			NeMa	SIM-T	VELSET	NAGA
	P	R	F1	P	R	F1	P	R	F1	P	R	F1				
Exact Match	99.11	86.08	92.14	82.19	82.19	82.19	95.24	95.24	95.24	98.91	98.91	**98.91**	224.94	564.83	275.99	**177.58**
Noisy Edges	99.57	86.46	92.55	86.40	86.40	86.40	96.63	96.63	96.63	97.32	97.32	**97.32**	226.14	644.25	272.22	**185.70**
Noisy Labels	98.61	84.62	91.08	68.72	68.72	68.72	95.26	95.26	**95.26**	97.32	52.78	68.44	226.67	544.70	262.69	**163.96**
Combined	99.55	86.59	92.62	71.63	71.63	71.63	94.67	94.67	**94.67**	95.64	53.18	68.36	225.48	631.97	248.19	**196.70**
Average	**99.21**	85.94	92.10	77.23	77.23	77.23	95.45	**95.45**	**95.45**	97.30	75.55	85.06	225.81	596.44	264.77	**180.98**

Table 2: Average F1 score (%) and running time results over varying graph matching queries for IMDB dataset.

Query Scenario	Quality Performance												Running Time Performance (in sec)			
	NeMa			SIM-T			VELSET			NAGA			NeMa	SIM-T	VELSET	NAGA
	P	R	F1	P	R	F1	P	R	F1	P	R	F1				
Exact Match	92.70	92.31	92.50	80.29	80.29	80.29	95.95	95.95	95.95	100	100	**100**	29.43	152.75	43.32	**7.03**
Noisy Edges	91.22	90.98	91.10	79.68	79.68	79.68	98.94	98.94	98.94	100	100	**100**	29.50	170.18	42.35	**6.98**
Noisy Labels	92.65	62.16	74.40	60.31	60.31	60.31	98.90	98.90	**98.90**	100	53.73	69.90	29.33	153.02	42.79	**6.34**
Combined	91.44	58.61	71.43	63.48	63.48	63.48	97.84	97.84	**97.84**	98.33	45.92	62.61	28.98	171.76	45.31	**6.51**
Average	92.00	76.01	83.25	70.94	70.94	70.94	97.91	**97.91**	**97.91**	**99.58**	74.91	85.50	29.31	161.93	43.44	**6.71**

- **Noisy Labels:** Dla zmienionych etykiet NAGA radzi sobie już znacznie gorzej. VELSET daje najlepsze wyniki, również w porównaniu do algorytmu NeMa, przy nieco większym czasie działania. Etykiety w przypadku danych z IMDB są unikalne, stąd prawdopodobnie słabe wyniki NeMa na tym zbiorze danych. Podobnie NAGA przeszukuje wierzchołki jedynie o identcznych etykietach, stąd wysokie precision (nie wskazuje błędnych), ale już niskie recall - nie wskazuje poprawnych grafów gdy choć jeden wierzchołek ma inną etykietę

Table 1: Average F1 score (%) and running time results over varying graph matching queries for YAGO dataset.

Query Scenario	Quality Performance												Running Time Performance (in sec)			
	NeMa			SIM-T			VELSET			NAGA			NeMa	SIM-T	VELSET	NAGA
	P	R	F1	P	R	F1	P	R	F1	P	R	F1				
Exact Match	99.11	86.08	92.14	82.19	82.19	82.19	95.24	95.24	95.24	98.91	98.91	**98.91**	224.94	564.83	275.99	**177.58**
Noisy Edges	99.57	86.46	92.55	86.40	86.40	86.40	96.63	96.63	96.63	97.32	97.32	**97.32**	226.14	644.25	272.22	**185.70**
Noisy Labels	98.61	84.62	91.08	68.72	68.72	68.72	95.26	95.26	**95.26**	97.32	52.78	68.44	226.67	544.70	262.69	**163.96**
Combined	99.55	86.59	92.62	71.63	71.63	71.63	94.67	94.67	**94.67**	95.64	53.18	68.36	225.48	631.97	248.19	**196.70**
Average	**99.21**	85.94	92.10	77.23	77.23	77.23	95.45	**95.45**	**95.45**	97.30	75.55	85.06	225.81	596.44	264.77	**180.98**

Table 2: Average F1 score (%) and running time results over varying graph matching queries for IMDB dataset.

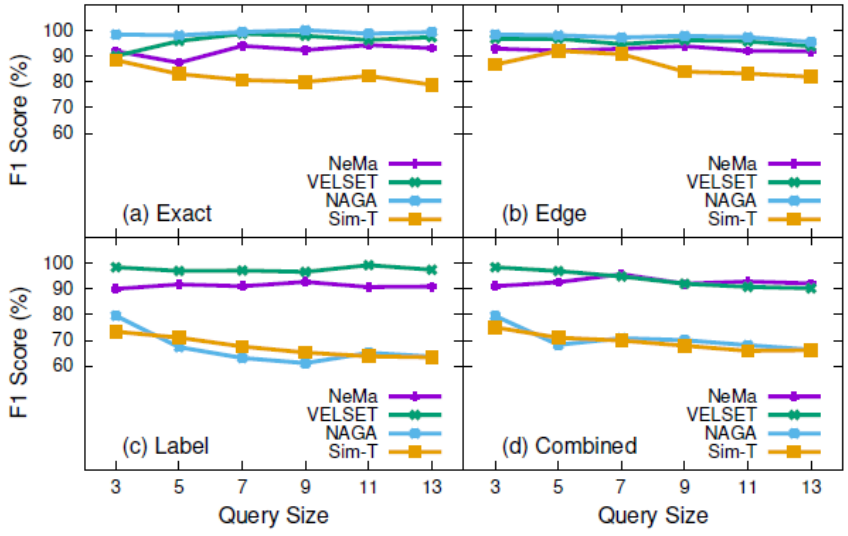
Query Scenario	Quality Performance												Running Time Performance (in sec)			
	NeMa			SIM-T			VELSET			NAGA			NeMa	SIM-T	VELSET	NAGA
	P	R	F1	P	R	F1	P	R	F1	P	R	F1				
Exact Match	92.70	92.31	92.50	80.29	80.29	80.29	95.95	95.95	95.95	100	100	**100**	29.43	152.75	43.32	**7.03**
Noisy Edges	91.22	90.98	91.10	79.68	79.68	79.68	98.94	98.94	98.94	100	100	**100**	29.50	170.18	42.35	**6.98**
Noisy Labels	92.65	62.16	74.40	60.31	60.31	60.31	98.90	98.90	**98.90**	100	53.73	69.90	29.33	153.02	42.79	**6.34**
Combined	91.44	58.61	71.43	63.48	63.48	63.48	97.84	97.84	**97.84**	98.33	45.92	62.61	28.98	171.76	45.31	**6.51**
Average	92.00	76.01	83.25	70.94	70.94	70.94	97.91	**97.91**	**97.91**	**99.58**	74.91	85.50	29.31	161.93	43.44	**6.71**

- **Combined:** W przypadku najbardziej zaszumionym NAGA daje również znacznie gorsze wyniki. Podobnie jak w poprzednim przypadku VELSET daje najlepsze wyniki, przy czasie działania zbliżonym do NeMa.

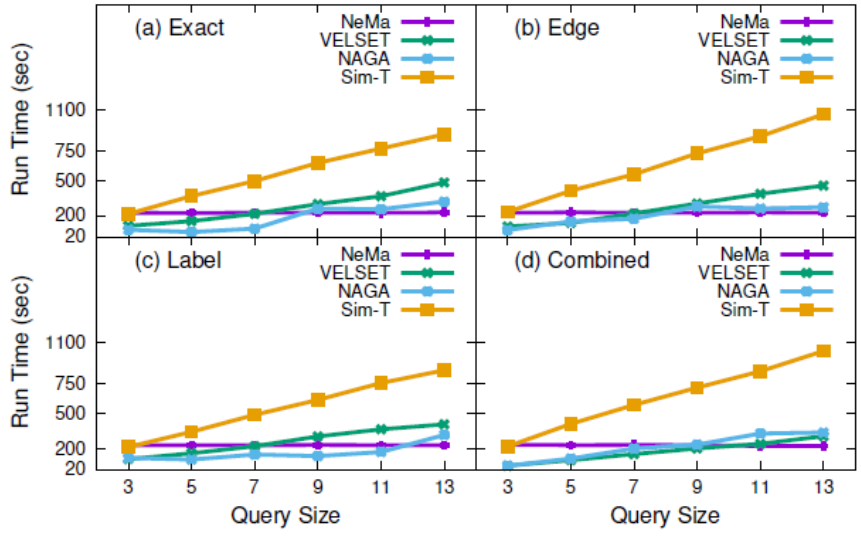
Table 3: Overall performance synopsis of the algorithms.

Methods	YAGO			IMDB		
	F1 (%)	Running Time (s)	Indexing Time (s)	F1 (%)	Running Time (s)	Indexing Time (s)
VELSET	**95.45**	264.77	1,205	**97.91**	43.44	209
NAGA	85.06	**180.98**	**250**	85.50	**6.71**	132
NeMa	92.10	225.81	∼10,000	83.25	29.31	∼10,000
SIM-T	77.23	596.44	374	70.94	161.93	**107**

- Biorąc pod uwagę wszystkie testy VELSET daje najlepsze wyniki
- Większa przewaga na danych IMDB wynika z unikalności etykiet
- NAGA, zgodnie z przewidywaniami, działa zdecydowanie najszybciej, i nadal średnio lepiej od NeMa
- również preprocessing dla proponowanych algorytmów jest mniejszy

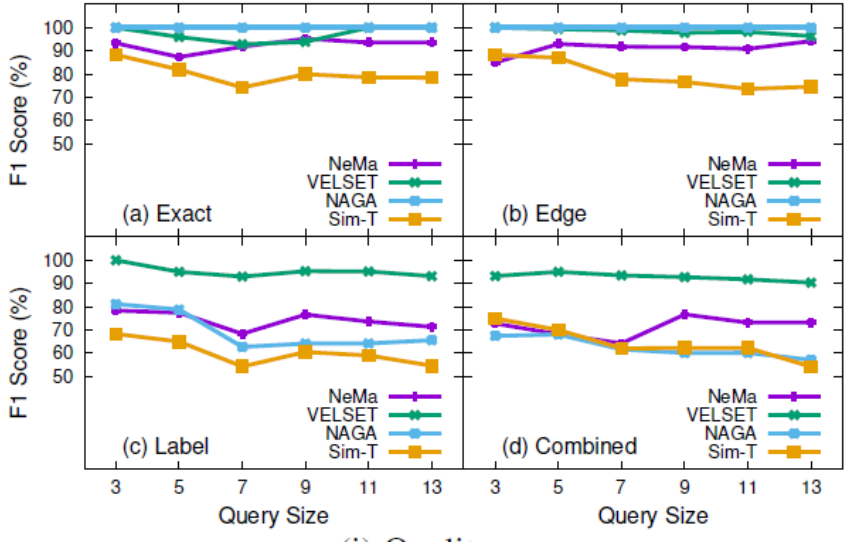


(i) Quality

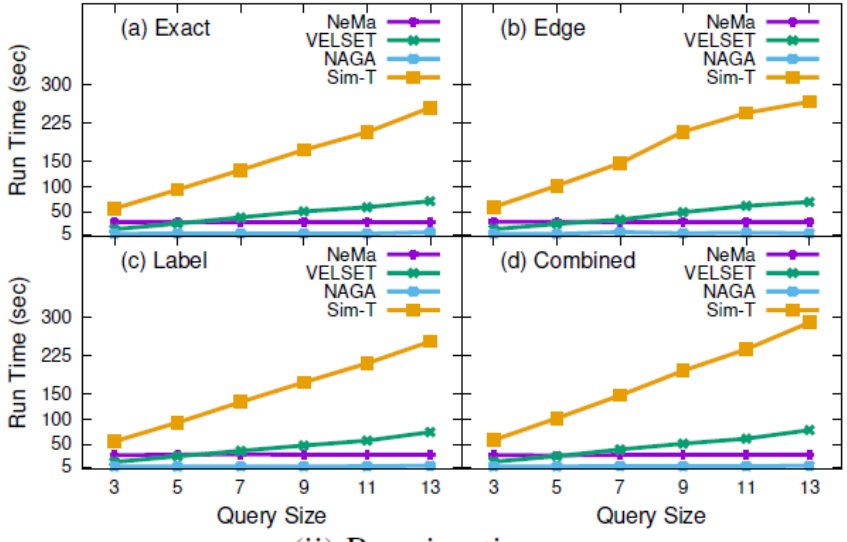


(ii) Running time

Figure 4: YAGO dataset: (i) F1 score and (ii) Running time performance for varying query sizes for different query scenarios.



(i) Quality

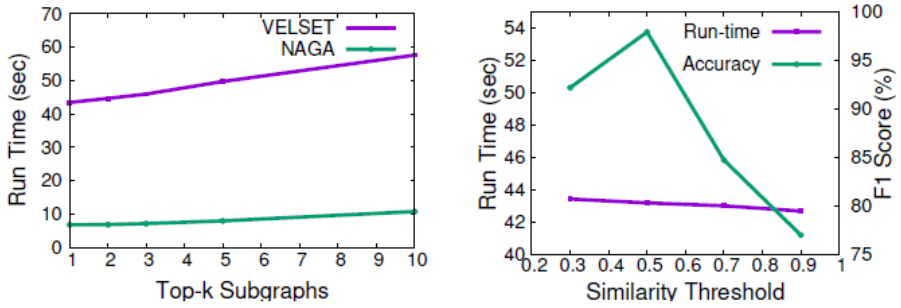


(ii) Running time

Figure 5: IMDB dataset: (i) F1 score and (ii) Running time performance for varying query sizes for different query scenarios.

Testowanie parametrów

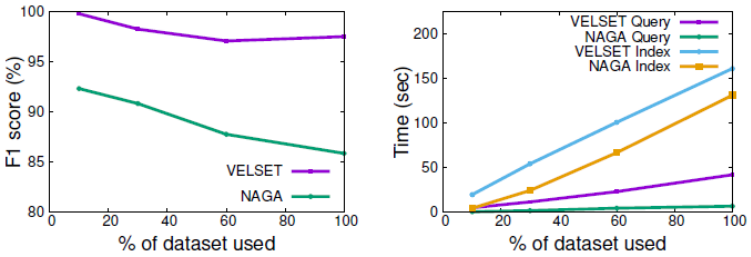
Testowanie parametrów



(a) Effect of k (b) Effect of τ

Figure 6: Performance on IMDB dataset: Effect of (a) size of answer set, k, on running time, (b) similarity threshold τ on VELSET.

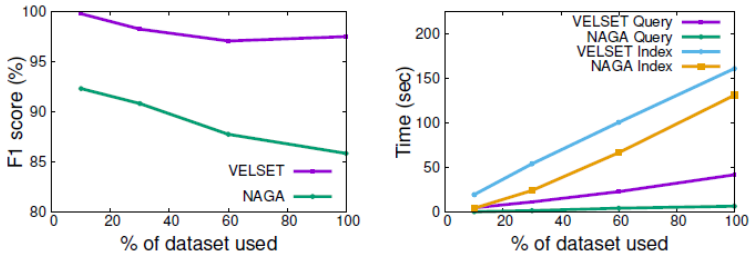
- **Top-k:** - przetestowano efektywność algorytmów dla większych wartości k. Liniowy wzrost jest zgodny z iteracyjnym podejściem w algorytmie. Autorzy twierdzą, że NAGA skaluje się lepiej (choć sam niższy współczynnik raczej wynika z tego, że NAGA jest szybsza)
- **Próg podobieństwa etykiet:** - próg kontroluje jak wiele par wierzchołków generujemy w algorytmie VELSET. Warto odnotować, że poniżej progu 0.5 algorytm działa nie tylko wolniej, ale też daje gorsze wyniki. Stąd też do porównania z innymi algorytmami użyto takiej wartości.



(a) Quality (b) Running time

Figure 7: Scalability results on IMDB dataset.

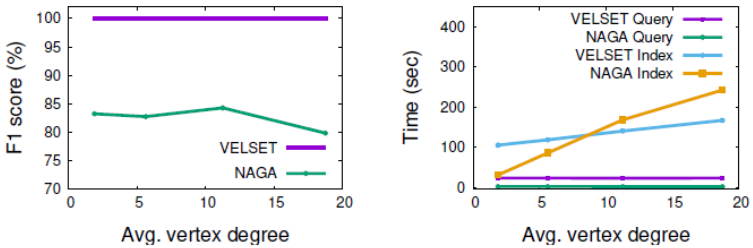
- **Skalowalność:** - przetestowano również algorytm dla różnej wielkości zapytań (od 10 do 100% wielkości grafu odpytywanego). Złożność wychodzi subliniowa.



(a) Quality (b) Running time

Figure 7: Scalability results on IMDB dataset.

- **Skalowalność:** - przetestowano również algorytm dla różnej wielkości zapytań (od 10 do 100% wielkości grafu odpytywanego). Złożność wychodzi subliniowa.



(a) Quality (b) Running time

Figure 8: Robustness results on dense Pokec dataset.

- **Gęstość:** - na koniec uwzględniono również gęstość danych (oba użyte zbiory były bardzo rzadkie). Do testu użyto „Pokec social network" (sieć społecznościowa on-line w Słowacji, ok. 1.6 mln wierzchołków i 30.6 mln krawędzi). Wyniki pokazują, że gęstość ma niewielki wpływ na efektywność algorytmu

Podsumowanie:

- zostało zaproponowane podejście wyszukiwania podgrafu opisując wierzchołki przez trójki etykiet (VELSET) lub uporządkowane etykiety sąsiadów (NAGA) i przy pomocy testu chi-kwadrat zachłannie przyporządkowuje kolejne pary wierzchołków
- eksperymenty pokazały, że podejścia dają lepsze wyniki, niż aktualne state-of-the-art algorymy
- publikacja dość szczegółowo opisuje kolejne kroki algorytmu
- brakuje dokładnego wytłumaczenia uproszczeń w stosowanych obliczeniach statystyki chi-kwadrat

Dziękuję za uwagę

Dziękuję za uwagę

Bibliografia:

- Sourav Dutta i in. - „Neighbor-Aware Search for Approximate Labeled Graph Matching using the Chi-Square Statistics”