Rozdział 1

Odległości na przestrzeni ciągów znaków

1.1. Podstawowe definicje

CZY TO JAKO CIĄGŁY TEKST CZY WSZYSTKO PISAĆ W DEFINICJI???

Niech $\Sigma = \{\Sigma_i\}$ będzie skończonym uporządkowanym alfabetem o wielkości $|\Sigma|$. *Napisem* nazywamy skończony ciąg znaków z Σ. Zbiór wszystkich napisów o długości n nad Σ jest oznaczony przez Σ^n, podczas gdy przez $\Sigma^* = \bigcup_{n=1}^{\infty} \Sigma^n$ rozumiemy zbiór wszystkich napisów utworzonych ze znaków z Σ [1].

O ile nie podano inaczej, używamy zmiennych s, t, u, v, w, x, y jako oznaczenie napisów oraz a, b, c do oznaczenia napisów jednoznakowych albo po prostu *znaków*. Pusty napis jest oznaczany przez ε. Przez $|s|$, dla każdego napisu $s \in \Sigma^*$, rozumiemy jego długość, czyli liczbę znaków w napisie. Ciąg zmiennych oznaczających napisy i/lub znaki oznaczają ich złączenie [1]. Dla rozróżnienia napisów od zmiennych reprezentujących napis, te pierwsze oznaczamy pismem maszynowym, np. `napis`.

Poprzez s_i rozumiemy i-ty znak z napisu s, dla każdego $i \in \{1, \ldots, |s|\}$. Podciąg kolejnych przylegających do siebie znaków z napisu nazywamy *podnapisem*. Podnapisem napisu s, który zaczyna się od i-tego znaku, a kończy na j-tym znaku, oznaczamy przez $s_{i:j}$, tj. $s_{i:j} = s_i s_{i+1} \ldots s_j$ dla $i < j$. Zakładamy również, że jeśli $j < i$, to $s_{i:j} = \varepsilon$ [1, 7].

Załóżmy, że napis s jest resprezentacją złączenia trzech, być może pustych, podnapisów w, x i y, tj. $s = wxy$. Wówczas podnapis w nazywamy *prefiksem*, natomiast podnapis y – sufiksem [1].

Podnapis złożony z kolejnych znaków napisu, o ustalonej długości q jest nazywany *q-gramem*. q-gramy o q równym jeden, dwa lub trzy mają specjalne nazwy: *unigram, bigram* i *trigram*. Jeśli $q > |s|$, to q-gramy napisu s są napisami pustymi [1].

Przykład 1.1. Niech Σ będzie alfabetem złożonym z 26 małych liter alfabetu łacińskiego oraz niech $s =$ `ela`. Wówczas mamy $|s| = 3$, $s \in \Sigma^3$ oraz $s \in \Sigma$. Co więcej, mamy $s_1 =$ `e`, $s_2 =$ `l`, $s_3 =$ `a`. Podnapis $1 : 2$ napisu s to $s_{1:2} =$ `el`. W napisie tym mamy do czynienia jedynie z q-gramami o q równym jeden, dwa oraz trzy: `e`, `l`, `a`; `el`, `la` oraz `ela` odpowiednio.

1.2. Odległości na przestrzeni ciągów znaków

W tym podrozdziale zajmiemy się odległościami na przestrzeni ciągów znaków. Można je podzielić na trzy grupy:

- oparte na operacjach edycyjnych (*edit operations*),
- oparte na q-gramach,
- miary heurystyczne.

Pierwszy rodzaj odległości jest najczęściej używany w algorytmach zajmujących się optymalnym dopasowaniem, dlatego też poświęcimy mu największą część niniejszego rozdziału. Odległości oparte na q-gramach ... (TUTAJ COŚ O NICH). Natomiast miary heurystyczne są rzadko stosowane, będąc zazwyczaj używane w konkretnych przypadkach. Miary te miały jednak swój wkład w historię optymalizacji napisów, zatem pokrótce opiszemy je pod koniec tego rozdziału.

1.2.1. Odległości oparte na operacjach edycyjnych

HISTORIA ODLEGLOSCI EDYCYJNYCH?

Odległość edycyjna $ED(s,t)$ pomiędzy dwoma napisami s i t to minimalna liczba operacji edycyjnych potrzebna do przetworzenia s w t (i ∞, gdy taki ciąg nie istnieje) [4]. *Ścisłą odległością edycyjną* nazywamy minimalną liczbę nie nakładających się operacji edycyjnych, które pozwalają przekształcić jeden napis w drugi, i które nie przekształcają dwa razy tego samego podnapisu [1].

Napis może zostać przetworzony w drugi poprzez ciąg przekształceń jego podnapisów. Ten ciąg nazywany jest *ścieżką edycyjną (śladem edycji?)*, podczas gdy przekształcenia są nazywane *bazowymi operacjami edycyjnymi*. Bazowe operacje edycyjne, które polegają na mapowaniu napisu s w napis t, są oznaczane przez $s \to t$. Zbiór wszystkich bazowych operacji edycyjnych oznaczamy przez $\mathbb{B}$ [1].

Bazowe operacje edycyjne są zazwyczaj ograniczone do:

- usunięcie znaku: `l` $\to \varepsilon$, tj. usunięcie litery `l` , np. `ela` $\to$ `ea`,
- wstawienie znaku: $\varepsilon \to$ `k`, tj. wstawienie litery `k`, np. `ela` $\to$ `elka`,
- zamiana znaku: `e` $\to$ `a`, tj. zamiana litery `e` na `a`, np. `ala` $\to$ `ela`,
- transpozycja: `el` $\to$ `le`, tj. przestawienie dwóch przylegających liter `e` i `l`, np. `ela` $\to$ `lea`.

Własność 1.1. *Zakładamy, że $\mathbb{B}$ spełnia następujące własności [1]:*

- *jeśli $s \to t \in \mathbb{B}$, to odwrotna operacja $t \to s$ również należy do $\mathbb{B}$;*
- *$a \to a \in \mathbb{B}$ (operacja identycznościowa dla jednego znaku należy do $\mathbb{B}$);*
- *zbiór $\mathbb{B}$ jest zupełny: dla dwóch dowolnych napisów s i t, istnieje ślad edycji, który przekształca s w t.*

Zauważmy, że zbiór $\mathbb{B}$ nie musi być skończony.

Podobieństwo dwóch napisów może być wyrażone jako długość scieżki edycyjnej, dzięki której jeden napis zostaje przekształcony w drugi:

Definicja 1.1. *Mając dany zbiór bazowych operacji edycyjnych,* odległość edycyjna $ED(s,t)$ *jest równa długości najkrótszej ścieżki edycyjnej, która przekształca napis s w napis t. Najkrótsza ścieżka, która przekształca napis s w napis t jest nazywana* optymalną *ścieżką edycyjną [1].*

Przykładowe odległości edycyjne: Hamminga, najdłuższego wspólnego podnapisu (*longest common substring*), Levenshteina, optymalnego dopasowania napisów (*optimal string alignment*), Damareu-Levenshteina.

Przykład 1.2. JAKIŚ PRZYKŁĄD SCIEZKI I OPTYMALNEJ SCIEZKI.

Odległość edycyjna może być również zostać uogólniona na minimalny koszt, dzięki któremu przekształcamy jeden napis w drugi. Można to zrobić na dwa sposoby. Po pierwsze, bazowe operacje edycyjne mogą mieć przydzielone koszty (wagi) $\delta(a \to b)$ [8]. Zazwyczaj koszt każdej operacji wynosi jeden, jednak można na przykład nadać transpozycji mniejszy koszt niż operacji wstawienia znaku. Dalej, można rozszerzyć funkcję kosztu δ na ścieżkę edycyjną $E = a_1 \to b_1, a_2 \to b_2, \ldots, a_{E|} \to b_{|E|}$ poprzez $\delta(E) = \sum_{i=1}^{|E|} \delta(a_i \to b_i)$ [1]. Odtąd poprzez odległość między napisem s a napisem t będzie rozumiany minimalny koszt ze wszystkich możliwych ścieżek przekształcających s w t. Odległości zdefiniowane w ten sposób zazwyczaj są nazwyane *uogólnionymi* odległościami edycyjnymi.
Po drugie, zbiór operacji edycyjnych $\mathbb{B}$ może zostać rozszerzony o ważone zamiany (substytucje) (pod)napisów, zamiast operacji edycyjnych wykonywanych na pojedynczych znakach [6]. Odległości zdefiniowane w ten sposób zazwyczaj są nazwyane *rozszerzonymi* odległościami edycyjnymi. Przykładowo, $\mathbb{B}$ może zawierać operację $x \to ks$ o koszcie jednostkowym. Wówczas rozszerzona odległość pomiędzy napisami `xero` i `ksero` wynosi jeden, podczas gdy standardowa odległość wyniosłaby dwa (NIE WIEM CZY NIE TRZEBA TU COŚ WIECEJ POWIEDZIEC, NP, PODAC DEFINICJI ODLEGLOSCI JAK TO JEST W BOYTSOVIE) [1].

Definicja 1.2. *Mając dany zbiór bazowych operacji edycyjnych $\mathbb{B}$ oraz funkcję δ, która nadaje koszt wszystkim bazowym operacjom edycyjnym z $\mathbb{B}$, ogólna(?) odległość edycyjna pomiędzy napisami s i t jest zdefiniowana jako minimalny koszt ścieżki edycyjnej, która przekształca s w t [1].*

Własność 1.2. *Zakładamy, że funkcja kosztu $\delta(s \to t)$ ma następujące własności [1]:*

- $\delta(s \to t) \in \mathbb{R}$ *(koszt operacji jest liczbą rzeczywistą),*
- $\delta(s \to t) = \delta(t \to s)$ *(symetria),*
- $\delta(s \to t) \geq 0,\ \delta(s \to s) = 0$ *i* $\delta(s \to t) = 0 \Rightarrow s = t$ *(pozytywna określoność ??),*
- $\forall \gamma > 0$ *zbiór bazowych operacji* $\{s \to t \in \mathbb{B} | \delta(s \to t) < \gamma\}$ *jest skończony (skończoność podzbioru bazowych operacji, których koszt jest ograniczony z góry).*

Zauważmy, że ostatnia własność jest automatycznie spełniona dla skończonego zbioru $\mathbb{B}$.

Twierdzenie 1.3. *Z własności 1.1 i 1.2 wynika, że:*

- *dla każdych dwóch napisów s i t, istnieje ścieżka o minimalnym koszcie, tj. właściwie (prawdziwie, odpowiednio?) zdefiniowana odległość edycyjna z s do t [1],*
- *ogólna odległość edycyjna z definicji 1.2 jest metryką [8].*

Dowód. Żeby udowodnić, że $ED(s,t)$ jest metryką, musimy pokazać, że $ED(s,t)$ istnieje i jest dodatnio określona, symetryczna oraz subaddytywna (tj. spełnia nierówność trójkąta).

Z własności 1.2 wynika, że funkcja kosztu jest nieujemna (JA TEGO NIW WIDZE) i że tylko identyczność ma koszt równy zero. Stąd, bez utraty ogólności, możemy rozważyć jedynie takie ścieżki edycyjne, które nie zawierają operacji identycznościowych. Zatem, jeśli $s = t$, to jedyna optymalna ścieżka (która nie zawiera operacji identycznościowych) jest pusta i ma zerowy koszt. Jeśli $s \neq t$, z zupełności zbioru bazowych operacji edycyjnych wynika, że istnieje jedna lub więcej ścieżek edycyjnych, które przekształcają s w t. Wszystkie te ścieżki składają się z operacji edycyjnych o ściśle dodatnim koszcie.

Niech γ będzie kosztem ścieżki przekształcającej s w t. Rozważmy zbiór A ścieżek edycyjnych, które przekształcają s w t i których koszt jest ograniczony z góry przez γ. Zbiór A jest niepusty i składa się z operacji edycyjnych o dodatnim koszcie mniejszym niż γ. Zbiór operacji bazowych, których koszt jest ograniczony z góry przez γ jest skończony, co dowodzi, że zbiór A jest również skończony. Ponieważ A jest niepusty i skończony, to ścieżki edycyjne o mininalnym (dodatnim) koszcie istnieją i należą do A. Stąd, $ED(s,t) > 0$ dla $s \neq t$, tj. odległość edycyjna jest dodatnio określona.

Żeby udowodnić symetrię odległości edycyjnej, rozważmy optymalną ścieżkę E, która przekształca s w t, oraz odpowiadającą jej odwrotną ścieżkę E_r, która przekształca t w s. Równość ich kosztów $\delta(E) = \delta(E_r)$ wynika z symetrii funkcji kosztu i symetrii zbioru operacji bazowych $\mathbb{B}$.

Żeby pokazać subaddytywność, rozważmy optymalną ścieżkę E_1, która przekształca s w t, optymalną ścieżkę E_2, która przekształca t w u, oraz złożenie ścieżek E_1E_2, które przekształca s w u. Z tego, że $\delta(E_1E_2) = \delta(E_1) + \delta(E_2) = ED(s,t) + ED(t,u)$ oraz $\delta(E_1E_2) \geq ED(s,u)$ wynika, że $ED(s,t) + ED(t,u) \geq ED(s,u)$. ■

Odległość edycyjna jest metryką, nawet gdy funkcja kosztu δ nie jest subaddytywna. Co więcej, ponieważ ciąg nakładających się operacji, które przekształcają s w t, mogą mieć mniejszy koszt niż $\delta(s \to t)$, $\delta(s \to t)$ może być większe niż $ED(s,t)$. Rozważmy, na przykład, następujący alfbet: $\{\mathtt{a}, \mathtt{b}, \mathtt{c}\}$, gdzie symetria i brak subaddytywności funkcji δ jest zdefiniowana następująco:

$$\begin{aligned} \delta(\mathtt{a} \to \mathtt{c}) = \delta(\mathtt{b} \to \mathtt{c}) &= 1 \\ \delta(\mathtt{a} \to \varepsilon) = \delta(\mathtt{b} \to \varepsilon) = \delta(\mathtt{c} \to \varepsilon) &= 2 \\ \delta(\mathtt{a} \to \mathtt{b}) &= 3 \end{aligned}$$

Można zobaczyć, że $3 = \delta(\mathtt{a} \to \mathtt{b}) > \delta(\mathtt{a} \to \mathtt{c}) + \delta(\mathtt{c} \to \mathtt{b}) = 2$. Stąd optymalna ścieżka edycyjna $(\mathtt{a} \to \mathtt{c}), \mathtt{c} \to \mathtt{b})$ przekształca $\mathtt{a}$ w $\mathtt{b}$ z kosztem równym 2.

Subaddytywność odległości edycyjnej pozwala używać metod właściwych przestrzeniom metrycznym, takich jak drzewo Burkharda-Kellera (NO IDEA WHAT THAT IS, NIE WIEM

CZY O TYM WSPOMINAC). Niemniej jednak, problem minimalizacji zbioru nakładających się operacji edycyjnych, może być trudny.

Definicja 1.3. Odległością Hamminga *[2] na* Σ^* *nazywamy:*

$$d_{hamming}(s,t) = \begin{cases} \sum_{i=1}^{|s|}[1-\delta(s_i,t_i)], & gdy\ |s| = |t|, \\ \infty, & w\ przeciwnym\ przypadku, \end{cases}$$

gdzie

$$\delta(s_i,t_i) = \begin{cases} 1, & gdy\ s_i = t_i, \\ 0, & w\ przeciwnym\ przypadku. \end{cases}$$

Łatwo zauważyć, że odległość Hamminga spełnia definicję metryki. Intuicyjnie rzecz biorąc odległość Hamminga zlicza liczbę indeksów, na których dwa napisy mają różny znak. Odległość ta przyjmuje wartości ze zbioru $\{0,\ldots,|s|\}$, gdy $|s| = |t|$, natomiast jest równa nieskończoności, gdy napisy mają różne długości.

[PIĘKNY RYSUNEK??]

Przykład 1.3. Odległość Hamminga między słowami `koza` i `foka` wynosi $d_{hamming}(\texttt{koza},\texttt{foka}) = 2$, natomiast między słowami `koza` i `foczka` wynosi ona $d_{hamming}(\texttt{koza},\texttt{foczka}) = \infty$.

Definicja 1.4. Odległością najdłuższego wspólnego podnapisu *[5] na* Σ^* *nazywamy:*

$$d_{lcs}(s,t) = \begin{cases} 0, & gdy\ s = t = \varepsilon, \\ d_{lcs}(s_{1:|s|-1}, t_{1:|t|-1}), & gdy\ s_{|s|} = t_{|t|}, \\ 1 + min\{d_{lcs}(s_{1:|s|-1}, t), d_{lcs}(s, t_{1:|t|-1})\}, & w\ przeciwnym\ przypadku, \end{cases}$$

Odległość najdłuższego wspólnego podnapisu również spełnia definicję metryki. Przyjmuje wartości ze zbioru $\{0, |s| + |t|\}$, przy czym maksimum jest osiągane, gdy s i t nie mają ani jednego wspólnego znaku. Odległość ta zlicza liczbę usunięć i wstawień, potrzebnych do przetworzenia jednego napisu w drugi.

Przykład 1.4. Odległość najdłuższego wspólnego podnapisu między słowami `koza` i `foka` wynosi: $d_{lsc}(\texttt{koza},\texttt{foka}) = 4$, bo $\texttt{koza} \xrightarrow[1]{us.\ k} \texttt{oza} \xrightarrow[1]{us.\ z} \texttt{oa} \xrightarrow[1]{wst.\ f} \texttt{foa} \xrightarrow[1]{wst.\ k} \texttt{foka}$.

Powyższy przykład pokazuje, że w ogólności nie ma unikalnej najkrótszej drogi transformacji jednego napisu w drugi, gdyż można zamienić kolejność usuwania (lub wstawiania) znaków i również uzyskać odległość równą 4.

Jak sugeruje nazwa, odległość najdłuższego wspólnego podnapisu, ma też inną interpretację. Poprzez wyrażenie *najdłuższy wspólny podnapis* rozumiemy najdłuższy ciąg utworzony przez sparowanie znaków z s i t nie zmieniając ich porządku. Wówczas odległość ta jest rozumiana jako liczba niesparowanych znaków z obu napisów. W powyższym przykładzie może to być zwizualizowane następująco:

[PIĘKNY RYSUNEK??]

Jak widać na rysunku, litery $|k|$, $|z|$, $|f|$ i $|k|$ pozostają bez pary, dając odległość równą 4.

Definicja 1.5. *Uogólnioną* odległością Levenshteina *[3] na* Σ^* *nazywamy:*

$$d_{lv}(s,t) = \begin{cases} 0, & gdy\ s = t = \varepsilon, \\ min\{ & \\ \quad d_{lv}(s, t_{1:|t|-1}) + w_1, & \\ \quad d_{lv}(s_{1:|s|-1}, t) + w_2, & \\ \quad d_{lv}(s_{1:|s|-1}, t_{1:|t|-1}) + [1 - \delta(s_{|s|}, t_{|t|})]w_3 & \\ \quad \}, & w\ przeciwnym\ przypadku, \end{cases}$$

gdzie w_1, w_2 *i* w_3 *to niezerowe liczby rzeczywiste, oznaczające kary za usunięcie, wstawienie oraz zamianę znaku.*

Odległość ta zlicza ważoną sumę usunięć, wstawień oraz zamian znaków, potrzebnych do przetworzenia jednego napisu w drugi. Gdy za wagi przyjmie się 1 mamy do czynienia ze zwykłą odległością Levenshteina, np. $d_{lv}(\texttt{koza}, \texttt{foka}) = 2$, bo $\texttt{koza} \xrightarrow[1]{zm.\ k\ na\ f} \texttt{foza} \xrightarrow[1]{zm.\ z\ na\ k} \texttt{foka}$. Powyższy przykład ilustruje, że dodatkowa elastyczność w porównaniu do odległości najdłuższego wspólnego podnapisu, daje mniejszą wartość odległości między napisami, jako że potrzebujemy jedynie dwóch zamian znaków [7].

Gdy za wagi przyjmiemy np. $(0.1, 1, 0.3)$, to $d_{lv}(\texttt{koza}, \texttt{foka}) = 0.6$, bo $\texttt{koza} \xrightarrow[0.3]{zm.\ k\ na\ f} \texttt{foza} \xrightarrow[0.3]{zm.\ z\ na\ k} \texttt{foka}$.

Uogólniona odległość Levenshteina spełnia definicję metryki, gdy $w_1 = w_2$. W przeciwnym przypadku nie spełnia ona założenia o symetrii, tj. podpunktu **??** definicji **??**. Jednakowoż, symetria zostaje zachowana przy jednoczesnej zamianie s i t oraz w_1 i w_2, jako że liczba usunięć znaków przy przetwarzaniu napisu s w napis t jest równa liczbie wstawień znaków przy przetwarzaniu napisu t w napis s [7]. Dobrze obrazuje to następujący przykład.

Przykład 1.5. Przyjmijmy za $(w_1, w_2, w_3) = (0.1, 1, 0.3)$. Wówczas uogólniona odległość Levenshteina dla napisów `koza` i `foczka` wynosi:

$$d_{lv}(\texttt{koza}, \texttt{foczka}) = 0.5, \tag{1.1}$$

bo

$$\texttt{koza} \xrightarrow[0.3]{zm.\ k\ na\ f} \texttt{foza} \xrightarrow[0.1]{wst.c} \texttt{focza} \xrightarrow[0.1]{wst.k} \texttt{foczka},$$

natomiast

$$d_{lv}(\texttt{foczka}, \texttt{koza}) = 2.3, \tag{1.2}$$

bo

$$\texttt{foczka} \xrightarrow[0.3]{zm.\ f\ na\ k} \texttt{koczka} \xrightarrow[1]{us.c} \texttt{kozka} \xrightarrow[1]{us.k} \texttt{koza}.$$

Gdy za wagi (w_1, w_2, w_3) przyjmiemy $(1, 0.1, 0.3)$, to uogólniona odległość Levenshteina wynosi:

$$d_{lv}(\texttt{koza}, \texttt{foczka}) = 2.3,$$

bo

$$\texttt{koza} \xrightarrow[0.3]{zm.\ k\ na\ f} \texttt{foza} \xrightarrow[1]{wst.c} \texttt{focza} \xrightarrow[1]{wst.\ k} \texttt{foczka},$$

czyli analogicznie, jak w przypadku 1.2. Natomiast

$$d_{lv}(\texttt{foczka}, \texttt{koza}) = 0.5,$$

bo

$$\texttt{foczka} \xrightarrow[0.3]{zm.\ f\ na\ k} \texttt{koczka} \xrightarrow[0.1]{us.c|} kozka \xrightarrow[0.1]{us.\ k} \texttt{koza},$$

czyli analogicznie, jak w przypadku 1.1.

Definicja 1.6. Odległością optymalnego dopasowania napisów *na* Σ^* *nazywamy:*

$$d_{osa}(s,t) = \begin{cases} 0, & gdy\ s = t = \varepsilon, \\ min\{ & \\ \quad d_{osa}(s, t_{1:|t|-1}) + w_1, & \\ \quad d_{osa}(s_{1:|s|-1}, t) + w_2, & \\ \quad d_{osa}(s_{1:|s|-1}, t_{1:|t|-1}) + [1 - \delta(s_{|s|}, t_{|t|})]w_3 & \\ \quad d_{osa}(s_{1:|s|-2}, t_{1:|t|-2}) + w_4,\ gdy\ s_{|s|} = t_{|t|-1}, s_{|s|-1} = t_{|t|} & \\ \quad \}, & w\ przeciwnym\ przypadku, \end{cases}$$

gdzie w_1, w_2*,* w_3 *iw*$_4$ *to niezerowe liczby rzeczywiste, oznaczające kary za odpowiednio usunięcie, wstawienie, zamianę oraz transpozycję znaków.*

Odległość optymalnego dopasowania napisów jest bezpośrednim rozszerzeniem odległości Levenshteina, która zlicza również liczbę transpozycji przylegających znaków, potrzebnych do przetworzenia jednego napisu w drugi. W przeciwieństwie do wcześniej zaprezentowanych odległości, nie spełnia ona nierówności trójkąta, tj. podpunktu **??** z definicji **??** [7]:

$$2 = d_{osa}(\texttt{ba}, \texttt{ab}) + d_{osa}(\texttt{ab}, \texttt{acb}) \leq d_{osa}(\texttt{ba}, \texttt{acb}) = 3,$$

gdyż

$$\texttt{ba} \xrightarrow[1]{transp.\ b\ i\ a} \texttt{ab} + \texttt{ab} \xrightarrow[1]{wst.c} \texttt{acb},$$

natomiast

$$\texttt{ba} \xrightarrow[1]{us.\ b} \texttt{a} \xrightarrow[1]{wst.\ c} \texttt{ac} \xrightarrow[1]{wst.\ b} \texttt{acb}.$$

W ostatnim przykładzie, zmniejszenie odległości poprzez zamianę liter $|a|$ i $|b|$, a następnie wstwienie litery $|c|$ spowodowałoby dwukrotne przekształcenie tego samego podnapisu. Z tego powodu odległość optymalnego dopasowania napisów bywa również nazywana *ścisłą odległością Damerau-Levenshteina* i jest często mylona z właściwą *odległością Damerau-Levenshteina*. Ta ostatnia pozwala na przekształcanie tego samego podnapisu wielokrotnie i jest metryką w rozumieniu definicjij **??**, ale nie spełnia założenia o nie przekształcaniu wielokrotnie tego samego podnapisu [7].

[MIARA DAMERAU-LEVENSHTEINA??? WTEDY ZMIENIC DEFINICJE O NIEPRZERABIANIU 2 RAZY TEGO SAMEGO PODNAPISU]

W przypadku odległości Levenshteina i odległości optymalnego dopasowania napisów, maksymalna odległość między napisami s i t wynosi $max\{|s|, |t|\}$. Jednakowoż, gdy liczba dopuszczalnych operacji edycyjnych rośnie, to liczba dopuszczalnych ścieżek między napisami wzrasta, co pozwala ewentualnie zmiejszyć odległość między napisami. Dlatego relację między zaprezentowanymi powyżej odległościami można podsumować następująco [7]:

$$\left.\begin{array}{r} \infty \geq |s| \geq d_{hamming}(s,t) \\ |s| + |t| \geq d_{lcs}(s,t) \\ max\{|s|, |t|\} \end{array}\right\} \geq d_{lv}(s,t) \geq d_{osa}(s,t) \geq 0.$$

Literatura

[1] Leonid Boytsov. Indexing methods for approximate dictionary searching: Comparative analysis. *J. Exp. Algorithmics*, 16:1.1:1.1–1.1:1.91, Maj 2011.

[2] R. W. Hamming. Error Detecting and Error Correcting Codes. *Bell System Techincal Journal*, 29:147–160, 1950.

[3] V. Levenshtein. Binary codes capable of correcting spurious insertions and deletions of ones. *Problems of Information Transmission*, 1:8–17, 1965.

[4] G. Navarro. A guided tour to approximate string matching. *ACM Computing Surveys*, 33(1):31–88, 2001.

[5] Saul B. Needleman, Christian D. Wunsch. A general method applicable to the search for similarities in the amino acid sequence of two proteins. *Journal of Molecular Biology*, 48(3):443–453, Marzec 1970.

[6] Esko Ukkonen. Algorithms for approximate string matching. *Inf. Control*, 64(1-3):100–118, Marzec 1985.

[7] Mark P. J. van der Loo. The stringdist Package for Approximate String Matching. *The R Journal*, 6:111–122, 2014.

[8] Robert A. Wagner, Michael J. Fischer. The string-to-string correction problem. *Journal of the ACM*, 21(1):168–173, January 1974.