Metody bioinformatyki (MBI)

Wykład 12 - mikromacierze DNA, struktury drugorzędowe RNA

Robert Nowak

2014L

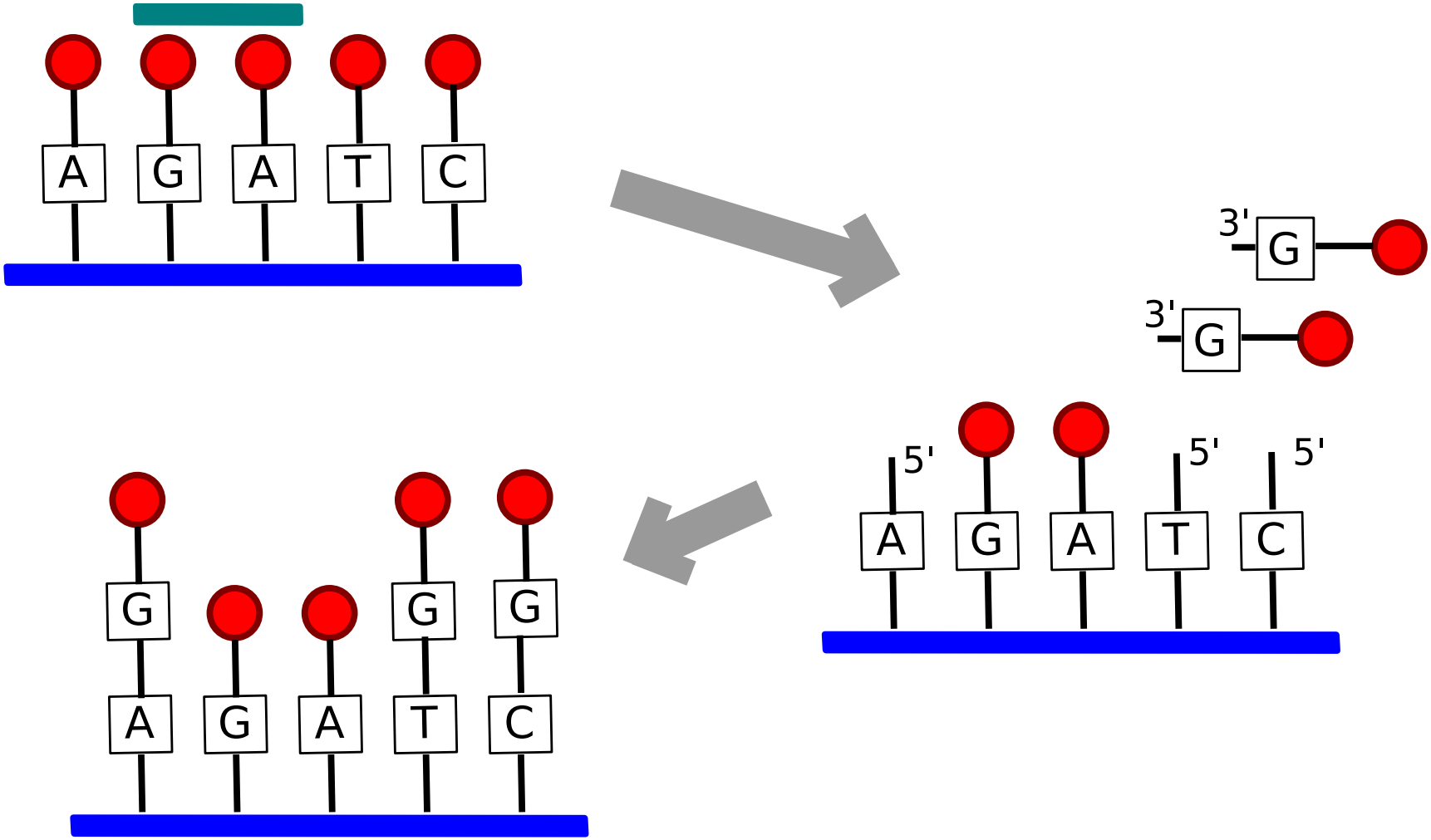
mikromacierze DNA

Metoda badawcza, pozwalająca badać obecność wielu cząsteczek DNA lub RNA jednocześnie, utworzona do odczytywania sekwencji

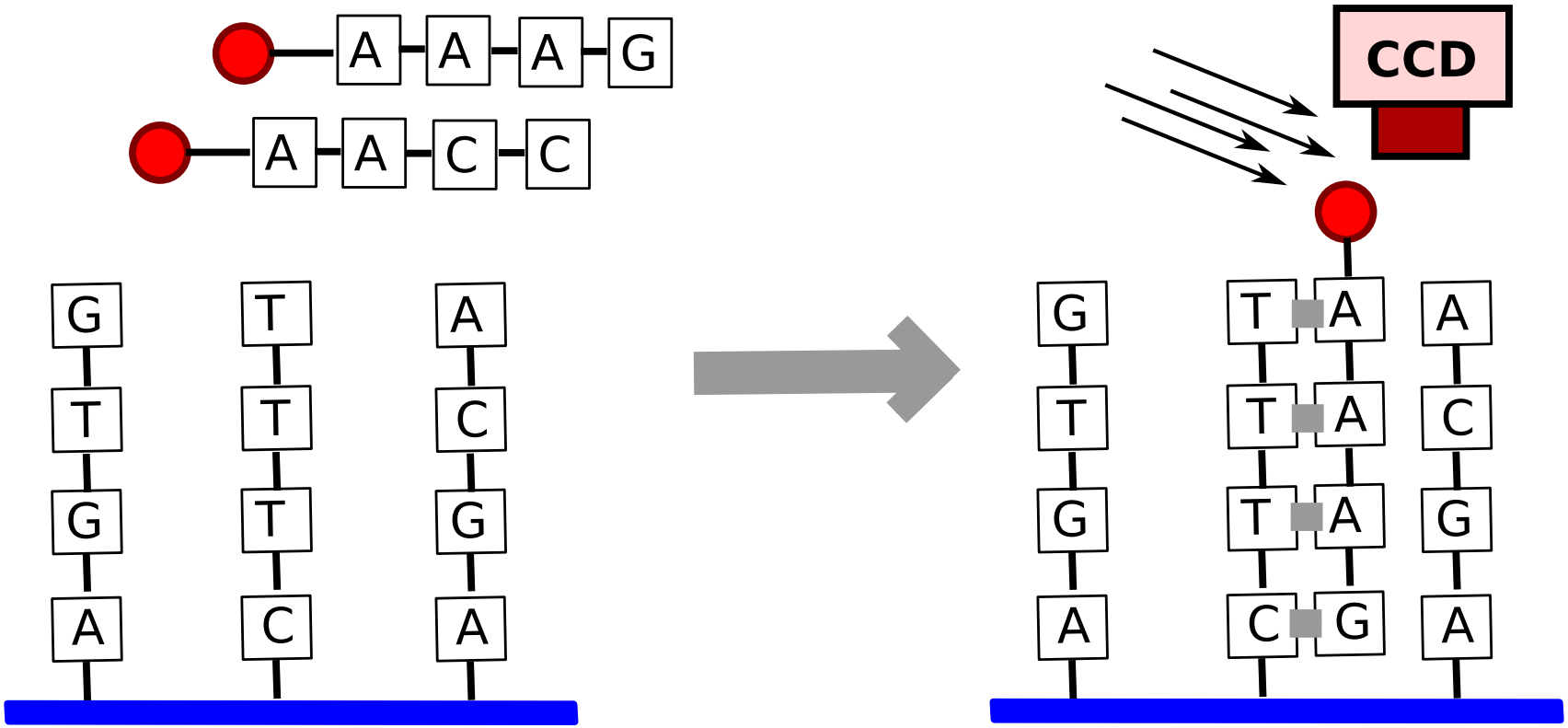
Zastosowanie mikromacierzy:

- analiza DNA, badanie wiele markerów jednocześnie
- analiza genów ulegających ekspresji, badanie cząsteczek mRNA

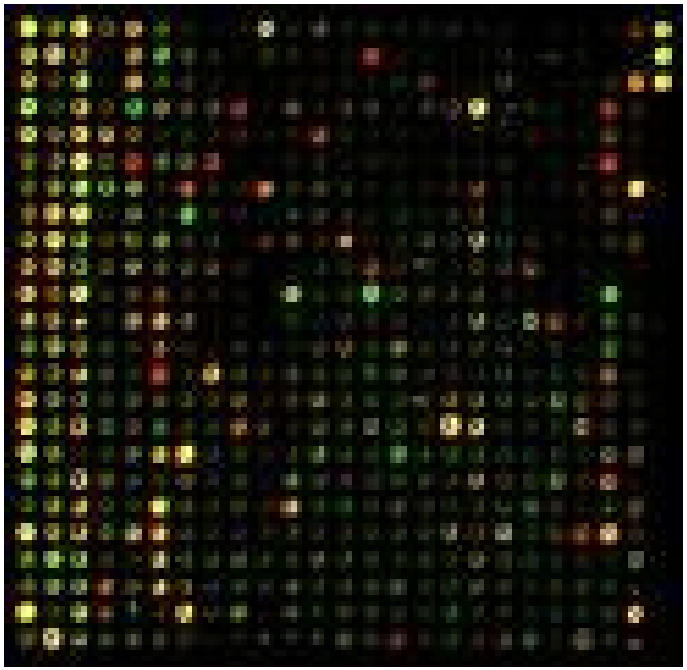
tworzenie mikromacierzy DNA



badanie roztworów za pomocą mikromacierzy DNA



wynik badania



Dla każdego doświadczenia:

- m atrybutów ($m \approx 10^5$)
- wartości:
  - binarne: występuje, nie występuje
  - rzeczywiste: intensywność $< 0, 1 >$

- zwykle przeprowadza się $n \approx 100$ doświadczeń
- mamy macierz $n \times m$, gdzie $n \ll m$ elementów x_{ij}
- problem znalezienia atrybutów istotnych

normalizacja danych

- usuwanie atrybutów, które mają tą samą wartość dla wszystkich przykładów
- obliczenie średniej i odchylenia standardowego

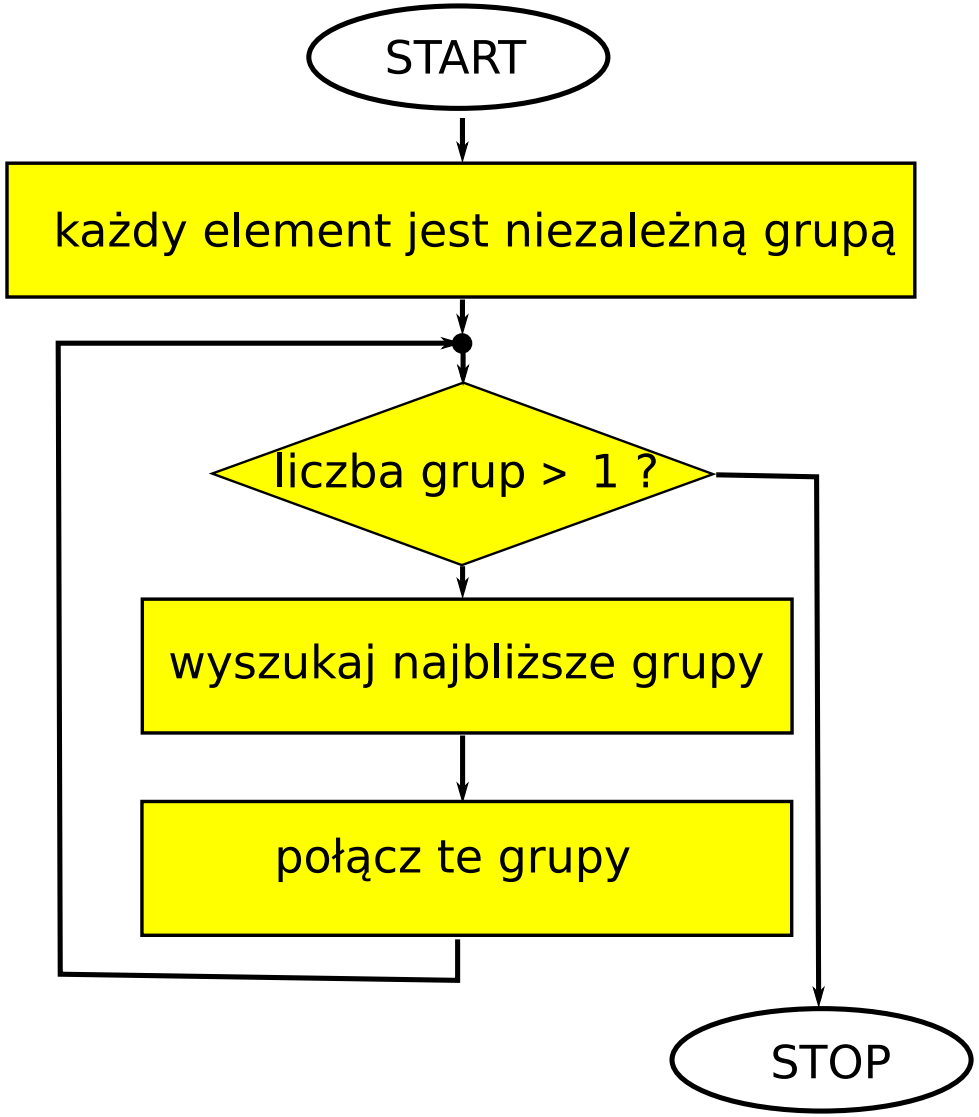
$$\mu_j = \frac{1}{n}\sum_{i=0}^{n} x_{ij} \text{ średnia}$$

$$\sigma_j = \sqrt{\frac{1}{n}\sum_{i=0}^{n}(x_{ij} - \mu_j)^2} \text{ odchylenie standardowe}$$

- przekształcenie danych, atrybut j ma $\mu = 0$, $\sigma = 1$

$$z_{ij} = \frac{x_{ij} - \mu_j}{\sigma_j}$$

algorytm grupowania hierarchicznego



- wymaga definicji odległości pomiędzy elementami
- wymaga definicji odległości pomiędzy grupami

definicja odległości

x, y oznaczają punkty ze zbiorów danych, każdy punkt ma m cech (atrybutów) $x =< x_1, x_2, ..., x_m >$

- odległość Euklidesowa

$$d_{xy} = \sqrt{\sum_{i=1}^{m}(x_i - y_i)^2}$$

- odległość Manhattan

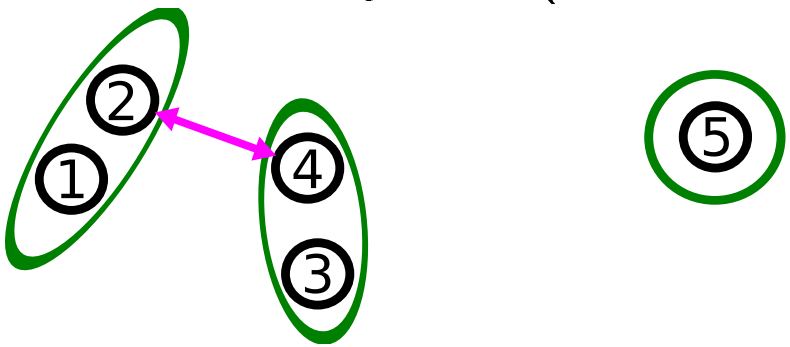
$$d_{xy} = \sum_{i=1}^{m}|x_i - y_i|$$

- korelacja

$$d_{xy} = \sum_{i=1}^{m} x_i y_i$$

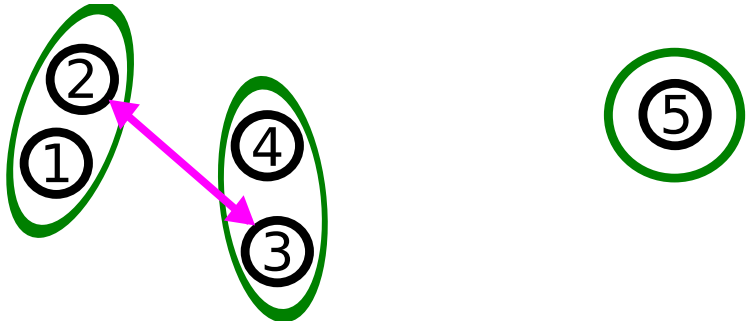
odległości grup

pojedyncze wiązanie (najmniejsza odległość)



$$d_s(X, Y) = \min_{x \in X, y \in Y} d_{xy}$$

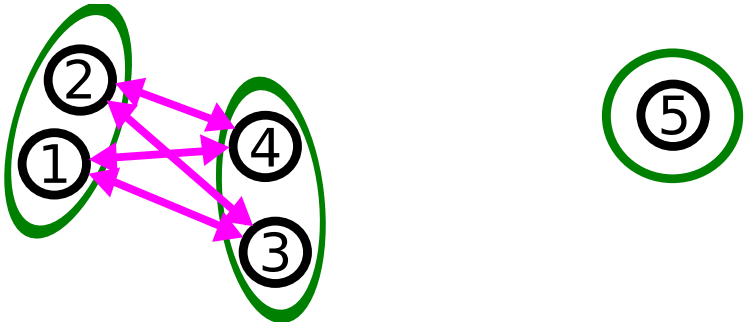
pełne wiązanie (największa odległość)



$$d_f(X, Y) = \max_{x \in X, y \in Y} d_{xy}$$

odległości grup (2)

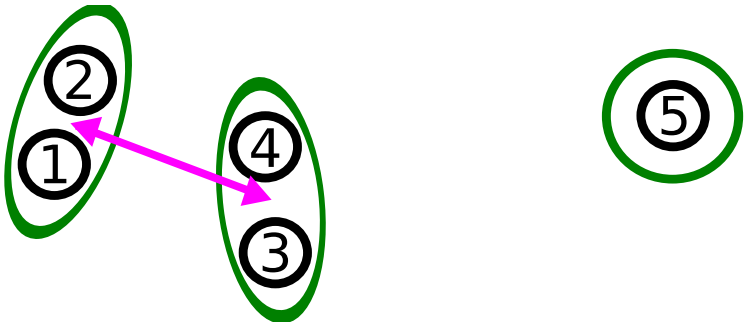
średnia odległość



$$d_a(X,Y) = \frac{1}{|X||Y|} \sum_{x \in X, y \in Y} d_{xy}$$

gdzie $|X|$ to ilość elementów w X

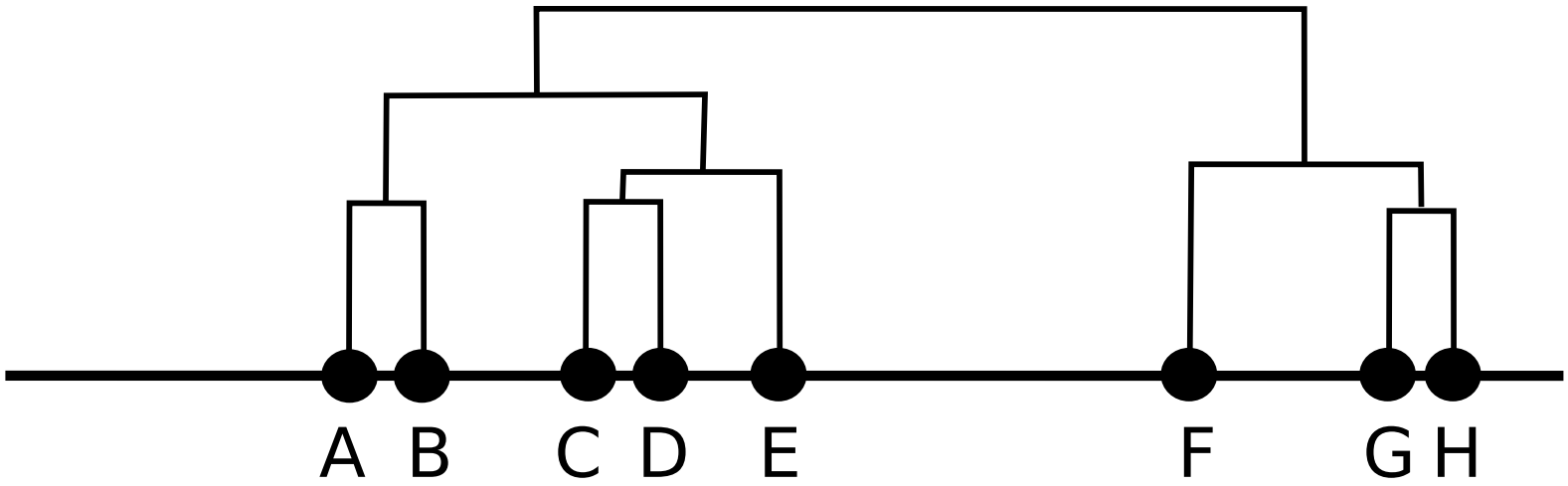
odległość pomiędzy środkami



$$d_c(X,Y) = d_{x'y'}$$

gdzie x' to średnia elementów $x \in X$

algorytm grupowania hierarchicznego (przykład)



- gdy dwie grupy: { {A, B, C, D, E}, {F, G, H }}
- gdy trzy grupy: { {A, B}, {C, D, E}, {F, G, H} }
- gdy cztery grupy: { {A, B}, {C, D, E}, {F}, { G, H} }
- itd.

algorytm grupowania hierarchicznego (przykład 2)

Tabela odległości pomiędzy obiektami A, B, C, D:

	A	B	C	D
A	0	2	6	11
B		0	4	9
C			0	5
D				0

Dla dwóch grup:

- jeżeli odległość między grupami to pojedyncze wiązanie (minimalna odległość pomiędzy elementami)?
  {A,B,C}{D}
- jeżeli odległość między grupami to pełne wiązanie (maksymalna odległość pomiędzy elementami)?
  {A,B}{C,D}

algorytm grupowania hierarchicznego - złożoność

- liczba przykładów: n, liczba atrybutów: m
- liczba kroków algorytmu $n - 1$
- każda iteracja:
  - $n(n-1)/2$ razy oblicza odległość
  - koszt obliczenia odległości $O(m)$
  - koszt iteracji $O(n^2 m)$

$$\mathbf{O(n^3 m)}$$

algorytm K-średnich (grupowanie niehierarchiczne)

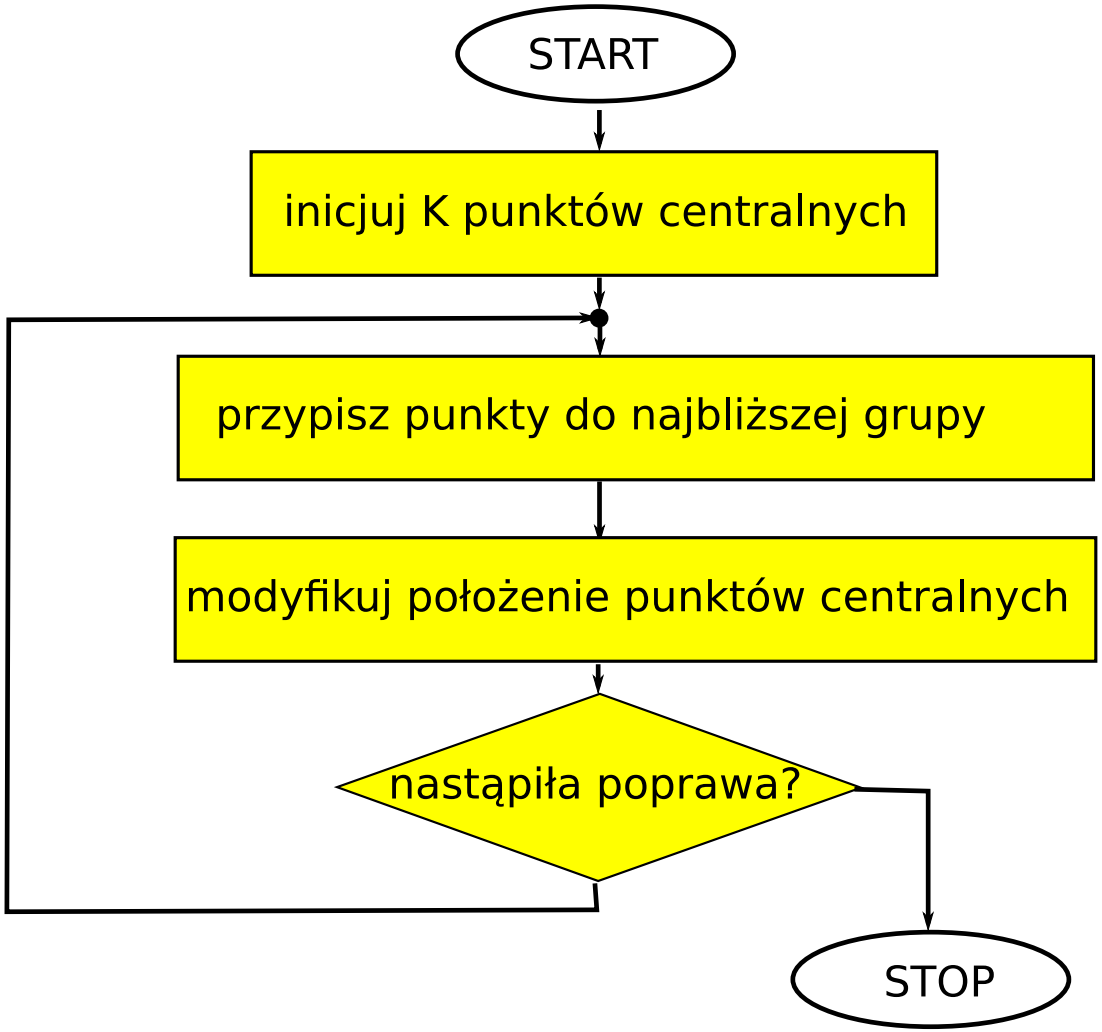
zakłada podział na K grup

- odległość $x =< x_1, x_2, ..., x_m >$ od c

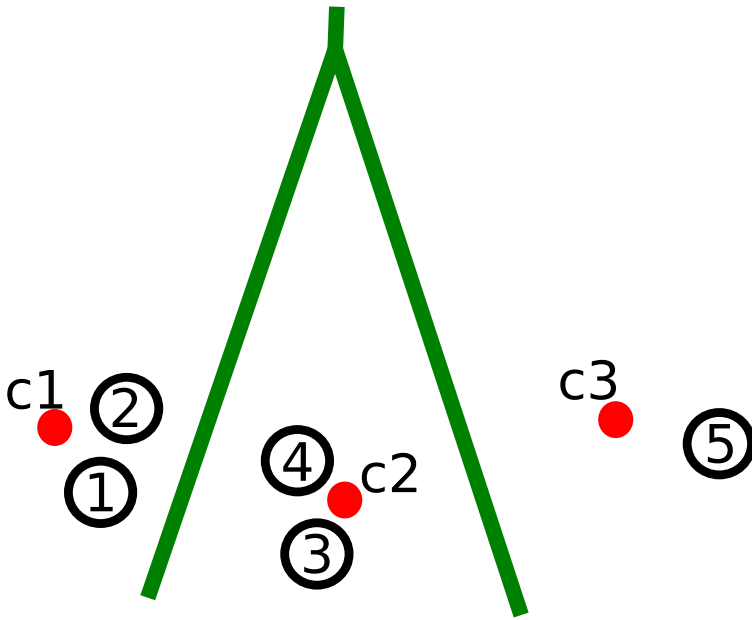
$$d_{xc} = \sum_{i=1}^{m} (x_i - c_i)^2$$

- funkcja błędu (którą minimalizujemy)

$$E = \sum_{c} \sum_{x \in c} d_{xc}$$



algorytm K-średnich (2)

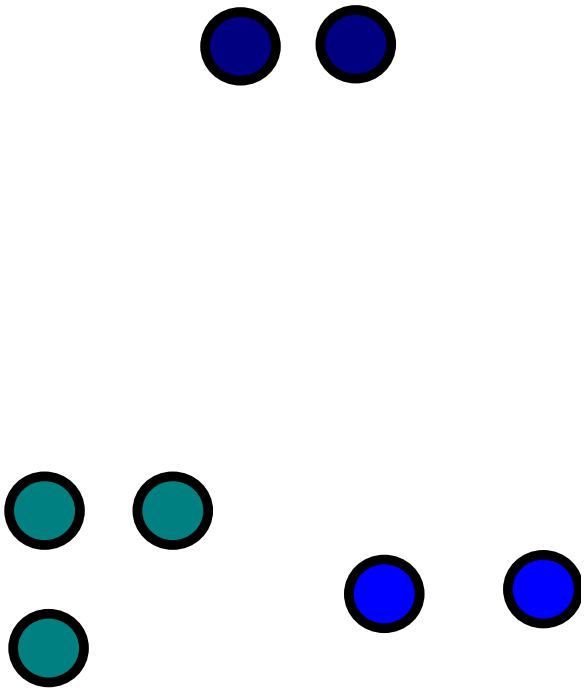


Algorytm optymalizacyjny:

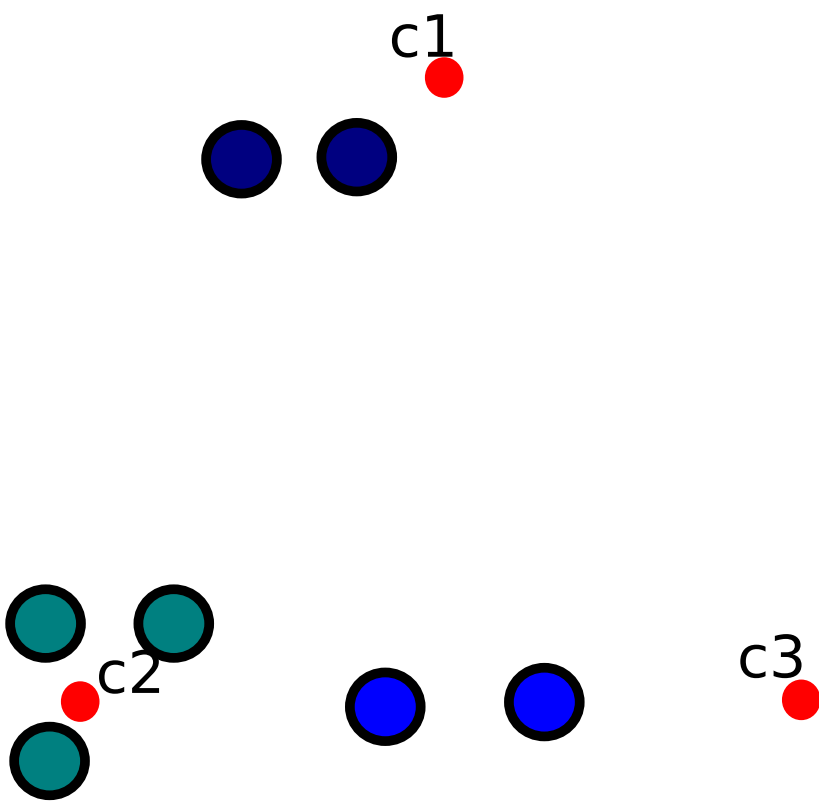
- inicjacja: losowa
- funkcja celu:

$$\arg\min_{C_1, C_2, \ldots, C_k} \sum_{c=1}^{k} \sum_{x_i \in C_c} \sum_{j=1}^{m} (x_{ij} - c_{cj})^2$$

algorytm K-średnich (przykład)

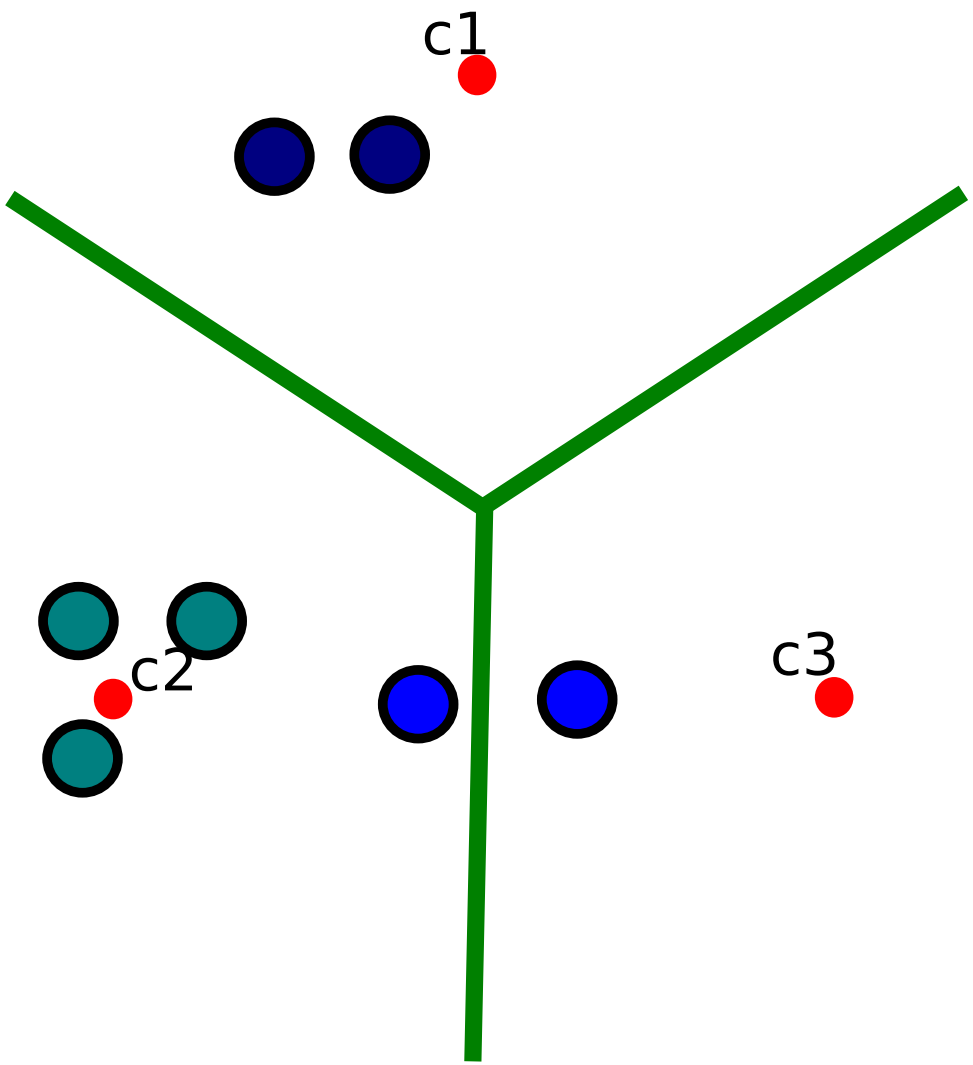


algorytm K-średnich (przykład)



- poszukiwane 3 grupy
- inicjacja punktów centralnych

algorytm K-średnich (przykład)

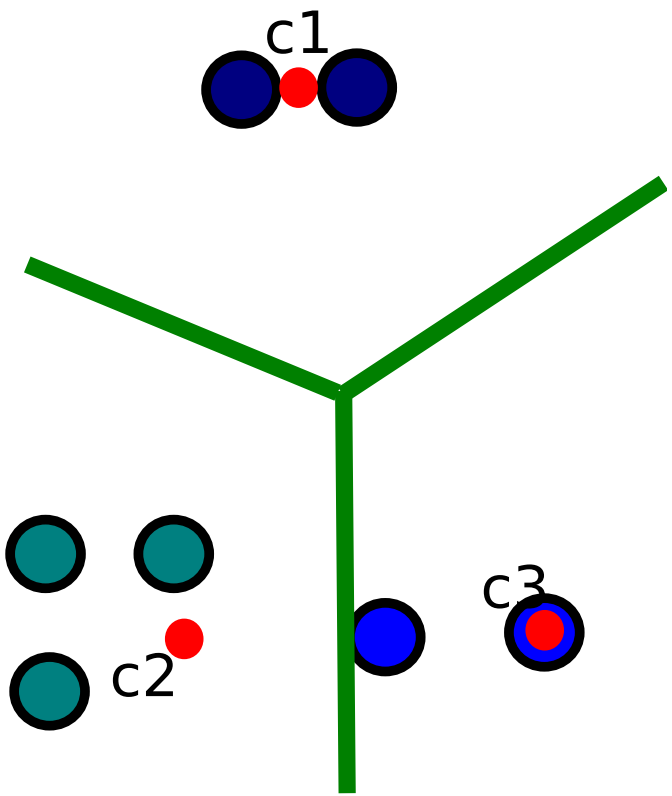


obliczenie przykładów należących do danej grupy

algorytm K-średnich (przykład)

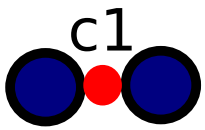
aktualizacja położenia
punktów centralnych

algorytm K-średnich (przykład)

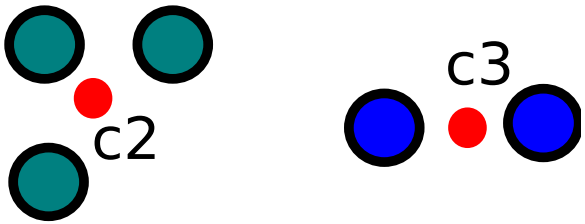


obliczenie przykładów należących do danej grupy

algorytm K-średnich (przykład)



aktualizacja punktów centralnych



algorytm K-średnich - złożoność

- liczba przykładów: n, liczba atrybutów m, liczba kroków algorytmu $p \approx n$
- każda iteracja:
  - koszt obliczenia odległości $O(m)$
  - koszt znalezienia grupy: dla n punktów k razy oblicza odległość od punktu centralnego, więc $O(knm)$
  - koszt obliczenia nowego punktu środkowego, dla n punktów oblicza średnią $O(nm)$
  - iteracja $O(knm + nm) = O(knm)$

$$\mathbf{O(kn^2m)}$$

algorytm znacznie wydajniejszy niż grupowanie hierarchiczne $O(n^3m)$, ponieważ $k \ll n$, ale problemy z właściwą inicjacją

algorytm analizy składowych głównych (PCA)

- pozwala na redukcję wymiaru problemu
- transformuje (liniowo) przestrzeń atrybutów dostarczając nowych współrzędnych

Dane wejściowe:

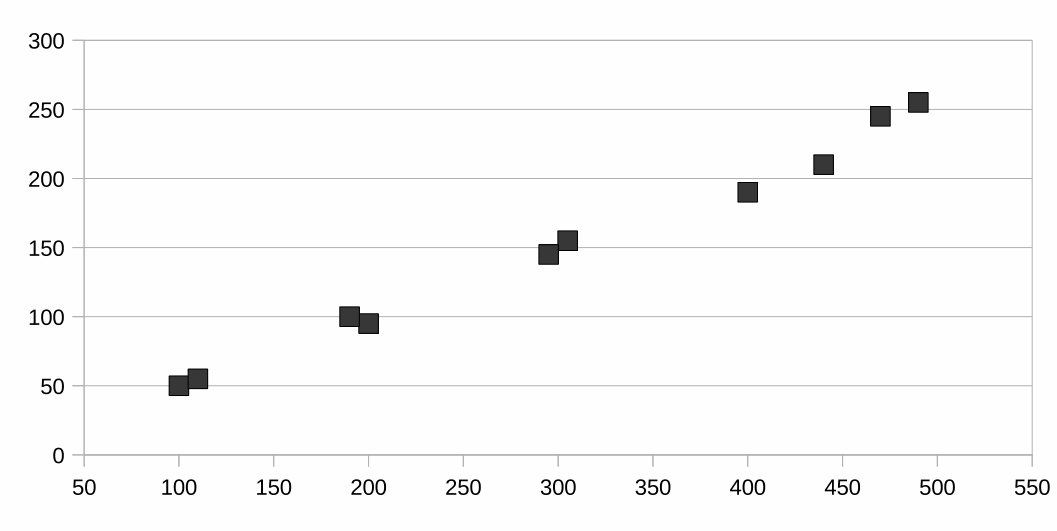
	atrybut			
pomiar	1	2	...	m
1	x_{11}	x_{12}	...	x_{1m}
2	x_{21}	x_{22}	...	x_{2m}
...	...	...	...	...
n	x_{n1}	x_{n2}	...	x_{nm}

- obliczenie μ_j oraz σ_j
- normalizacja danych

$$z_{ij} = \frac{x_{ij} - \mu_j}{\sigma_j}$$

algorytm PCA (2)

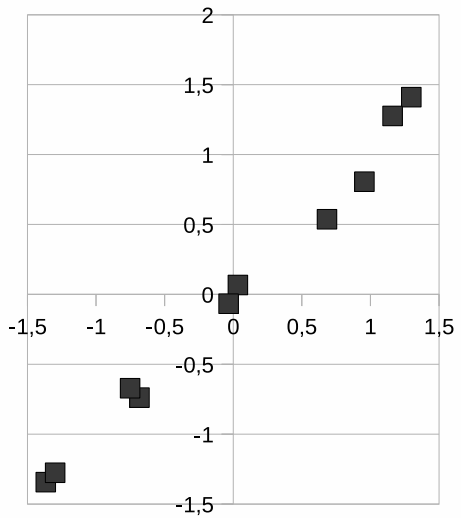
dane wejściowe (10 przykładów, 2 atrybuty):



$$\mu_1 = 300$$
$$\sigma_1 = 146.4$$
$$\mu_2 = 150$$
$$\sigma_2 = 74.4$$

normalizacja:

$$z_{i1} = \frac{x_{i1} - \mu_1}{\sigma_1}$$
$$z_{i2} = \frac{x_{i2} - \mu_2}{\sigma_2}$$



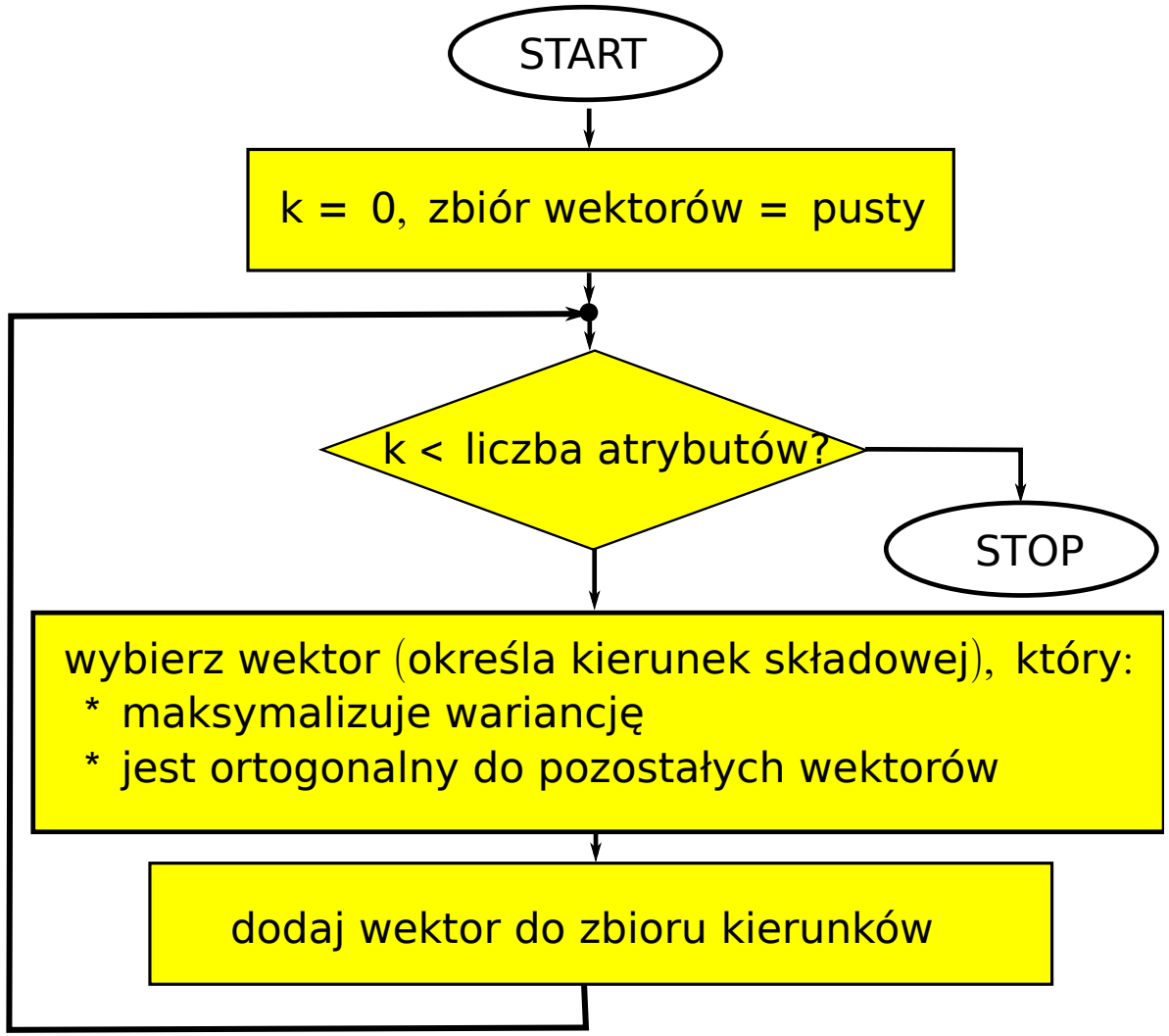
algorytm PCA (3)

po normalizacji wszystkie atrybuty mają parametry:

- $\mu = 0$
- $\sigma = 1$

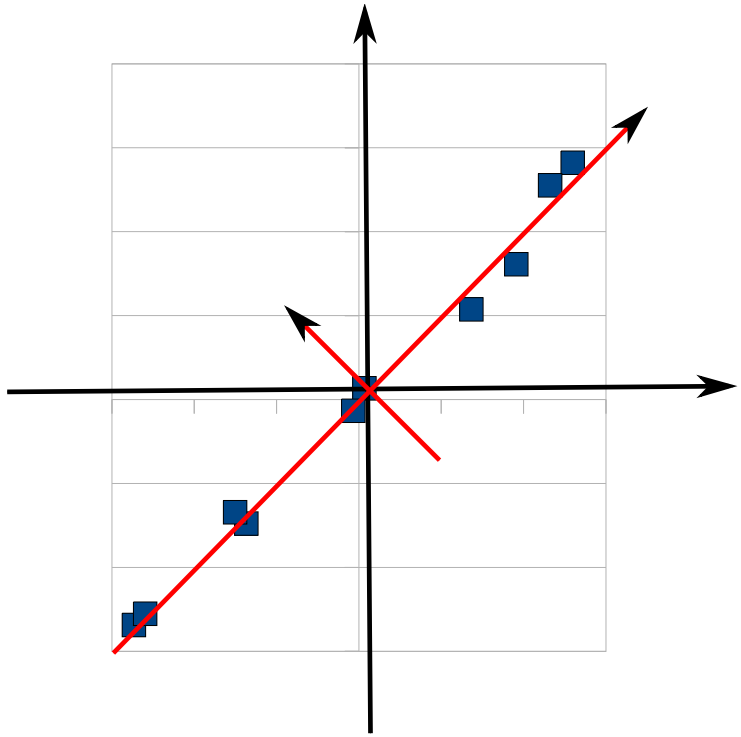
chcemy znaleźć nowy układ współrzędnych

- zakładamy tylko przekształcenia liniowe (obroty, odbicia)



algorytm PCA (4)

Kierunki składowych dla rozpatrywanego przykładu:



Założenia algorytmu PCA:

- rozpatrywane przekształcenia liniowe
- maksymalizowana jest wariancja (wariancja - klasyczna miara zróżnicowania)
- nowe kierunki składowych są ortogonalne

algorytm PCA (5) - kowariancja

$$\mathbf{Z} = \begin{bmatrix} z_{11} & z_{12} & \dots & z_{1m} \\ z_{21} & z_{22} & \dots & z_{2m} \\ \dots & \dots & \dots & \dots \\ z_{n1} & z_{n2} & \dots & z_{nm} \end{bmatrix} \mathbf{a_i} = \begin{bmatrix} z_{1i} \\ z_{2i} \\ \dots \\ z_{ni} \end{bmatrix} \mathbf{a_j} = \begin{bmatrix} z_{1j} \\ z_{2j} \\ \dots \\ z_{nj} \end{bmatrix}$$

dane po normalizacji: $\mu_{a_i} = \frac{1}{n}\sum_{k=0}^{n} z_{ki} = 0, \sigma^2_{a_i} = \frac{1}{n}\sum_{k=0}^{n} z^2_{ki} = 1$

Kowariancja - miarą liniowej zależności pomiędzy a_i i a_j

$$\sigma_{a_i a_j} = \frac{1}{n}\sum_{k=0}^{n} z_{ik} z_{jk} = \frac{1}{n}\mathbf{a_i}\mathbf{a_j}^T \text{ gdzie } \mathbf{a_j}^T = \begin{bmatrix} z_{1j} & z_{2j} & \dots & z_{nj} \end{bmatrix}$$

$$-1 \leq \sigma_{a_i a_j} \leq 1$$

algorytm PCA (6) - macierz kowariancji

$$\mathbf{C_Z} = \frac{1}{n}\mathbf{Z}\mathbf{Z}^T = \begin{bmatrix} \sigma_1^2 & \sigma_{12} & \dots & \sigma_{1m} \\ \sigma_{21} & \sigma_2^2 & \dots & \sigma_{2m} \\ \dots & \dots & \dots & \dots \\ \sigma_{n1} & \sigma_{n2} & \dots & \sigma_m^2 \end{bmatrix} = \begin{bmatrix} 1 & \sigma_{12} & \dots & \sigma_{1m} \\ \sigma_{21} & 1 & \dots & \sigma_{2m} \\ \dots & \dots & \dots & \dots \\ \sigma_{n1} & \sigma_{n2} & \dots & 1 \end{bmatrix}$$

Ponieważ $\sigma_{ij} = \sigma_{ji}$ macierz $\mathbf{C_Z}$ jest symetryczna

- sumaryczna wariancja

$$\sum_{j=0}^{m} \sigma_i = m$$

- po zmianie (rotacja, odbicie) układu współrzędnych sumaryczna wariancja nie zmieni się

algorytm PCA (7) - przekształcenie układu współrzędnych

$\mathbf{Y} = \mathbf{PZ}$, gdzie $\mathbf{P}$ jest macierzą przekształcenia

, macierz $\mathbf{P}$ zawiera wektory, które są kierunkami składowych

$$\mathbf{P} = [\mathbf{p_1}, \mathbf{p_2}, ..., \mathbf{p_m}]$$

Wykorzystując algorytmy algebry liniowej przekształca się przestrzeń, aby macierz kowariancji była diagonalna

$$\mathbf{C_Y} = \begin{bmatrix} \lambda_1 & 0 & ... & 0 \\ 0 & \lambda_2 & ... & 0 \\ ... & ... & ... & ... \\ 0 & 0 & ... & \lambda_m \end{bmatrix}$$

algorytm PCA (8) - jak znaleźć przekształcenie $\mathbf{P}$

$$\mathbf{C_Y} = \frac{1}{n}\mathbf{YY}^T = \frac{1}{n}(\mathbf{PZ})(\mathbf{PZ})^T = \frac{1}{n}\mathbf{PZZ}^T\mathbf{P}^T = \mathbf{PC_Z P}^T$$

dla macierzy symetrycznej $\mathbf{A}$, macierzy jej wektorów własnych $\mathbf{E}$ zachodzi zależność:

$$\mathbf{A} = \mathbf{EDE}^T, \text{ gdzie } \mathbf{D} \text{ jest macierzą diagonalną}$$

więc: $\mathbf{P}$ jest macierzą wektorów własnych macierzy C_Z

algorytm PCA (9) - przykład

Dla rozpatrywanego przykładu:

$$\mathbf{C_z} = \begin{bmatrix} 1 & 0.994 \\ 0.994 & 1 \end{bmatrix}$$

po rozkładzie na wartości własne i wektory własne:

$$\begin{bmatrix} 1 & 0.994 \\ 0.994 & 1 \end{bmatrix} = \begin{bmatrix} \frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{2} \\ \frac{\sqrt{2}}{2} & -\frac{\sqrt{2}}{2} \end{bmatrix} \begin{bmatrix} 1.994 & 0 \\ 0 & 0.006 \end{bmatrix} \begin{bmatrix} \frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{2} \\ \frac{\sqrt{2}}{2} & -\frac{\sqrt{2}}{2} \end{bmatrix}$$

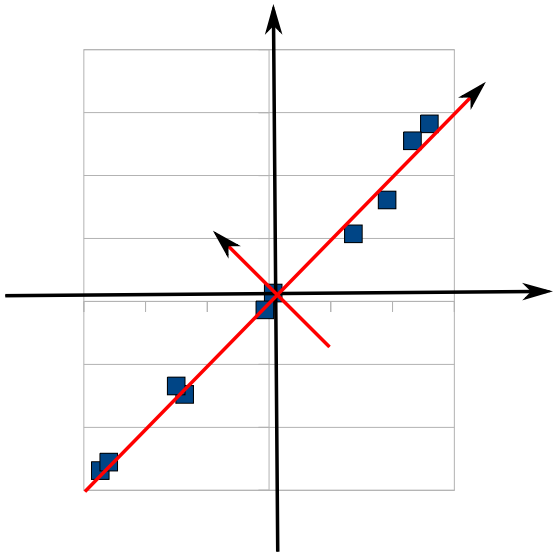
Nowe kierunki

$$\mathbf{p_1} = \begin{bmatrix} \frac{\sqrt{2}}{2} \\ \frac{\sqrt{2}}{2} \end{bmatrix}, \mathbf{p_2} = \begin{bmatrix} \frac{\sqrt{2}}{2} \\ -\frac{\sqrt{2}}{2} \end{bmatrix}, \lambda_1 = 1.994, \lambda_2 = 0.006$$

algorytm PCA (10) - przykład

$$\lambda_1 = 1.994, \lambda_2 = 0.006$$

- Po zamianie współrzędnych i eliminacji drugiego wymiaru będziemy mieli 99.7% wariancji dla pierwszego wymiaru (utracimy tylko 0.3% wariancji)
- W ten sposób można wybrać tylko istotne atrybuty



algorytm PCA (11) - podsumowanie

- algorytm nie posiada parametrów
- liniowo przekształca przestrzeń atrybutów
- wykorzystuje macierz korelacji, eliminuje kowariancję czyli liniowe zależności pomiędzy atrybutami
- pozwala na redukcję wymiarów, opuszczanie atrybutów o mniejszym znaczeniu (kompresja informacji)

Popularne kryterium dobierania ilości składowych (kryterium Kaisera-Guttmana)

należy zachować składowe, dla których wartości własne są większe od 1, czyli wkład składowej większy, niż wkład pojedynczej zmiennej

Czułość i selektywność

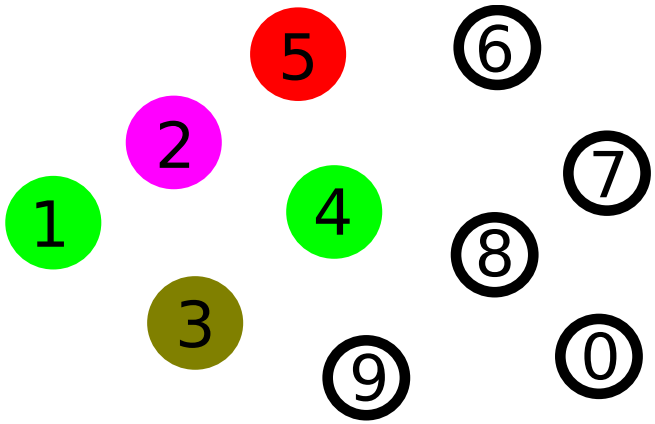
macierz pomyłek

		stan	
		plus	minus
wynik testu	dodatni	prawdziwie dodatni	fałszywie dodatni
	ujemny	fałszywie ujemny	prawdziwie ujemny

$$\text{czułość} = \frac{\text{prawdziwie dodatni}}{\text{prawdziwie dodatni} + \text{fałszywie ujemny}}$$

$$\text{selektywność} = \frac{\text{prawdziwie dodatni}}{\text{prawdziwie dodatni} + \text{fałszywie dodatni}}$$

Czułość i selektywność (2)



TP = 4	FP = 2
FN = 1	TN = 3

czułość = 0.8
selektywność = 0.67

Test na kolor:

nr	stan	wynik testu
0	NIE	NIE
1	TAK	NIE
2	TAK	TAK
3	TAK	TAK
4	TAK	TAK
5	TAK	TAK
6	NIE	TAK
7	NIE	TAK
8	NIE	NIE
9	NIE	NIE

Biologia syntetyczna

Biologia syntetyczna

Biologia syntetyczna – dziedzina inżynierii, projektowanie i realizacja sztucznych systemów biologicznych.

- chemiczna synteza DNA o zadanej sekwencji, 1970 r.
- synteza sztuczego genu, 1972 r.[1]
- synteza sztucznego genomu
  - wirus, 7500 bp, 2002 r.[2]
  - bakteria, 1 Mbp, 2010 r.[3]

Narzędzia informatyczne stosowane do:

- symulacje reakcji i procesów biologicznych
- przewidywania struktur cząsteczek
- optymalizacji warunków reakcji

[1] Khora, Total synthesis of the structural gene for an alanine transfer ribonucleic acid from yeast

[2] Cello ..., Chemical synthesis of poliovirus cDNA: generation of infectious virus in the absence of natural template

[3] Gibson ..., Creation of a bacterial cell controlled by a chemically synthesized genome

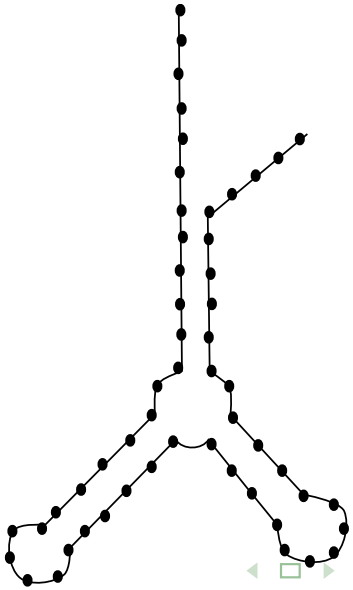
Struktura pierwszorzędowa, drugorzędowa i trzeciorzędowa

- struktura pierwszorzędowa - sekwencja symboli
- badanie struktur drugorzędowych DNA, RNA i białek (uwzględnienie oddziaływania nukleotydów lub aminokwasów)
- badanie struktur trzeciorzędowych (struktura atomów w przestrzeni 3D)

struktura pierwszorzędowa:

GCCGAUUAAA
CCACAGAAUC
CACCUGUGUC
UUUUUCCCAU
AUAUGGCUCG
GGAG...

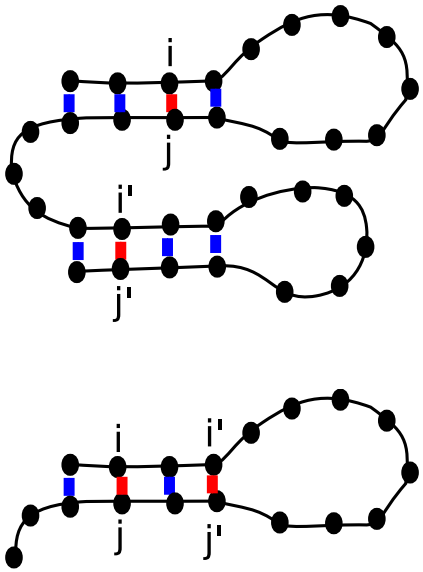
struktura drugorzędowa:



Struktura drugorzędowa - notacje

Sekwencja RNA $S = x_1 x_2 \ldots x_n$
Struktura drugorzędowa: zbiór par (i, j), gdzie $1 \leq i < j \leq n$, takich, że:

- $j - i > 3$ (pętle zewnętrzne nie mogą być krótsze niż 4 nukleotydy)
- jeżeli (i, j) oraz (i', j') są dwoma parami zasad to:
  - $i < j < i' < j'$
    (i,j) poprzedza (i',j')
  - $i' < j' < i < j$
    (i',j') poprzedza (i,j)
  - $i < i' < j' < j$
    (i,j) obejmuje (i',j')
  - $i' < i < j < j'$
    (i',j') obejmuje (i,j)



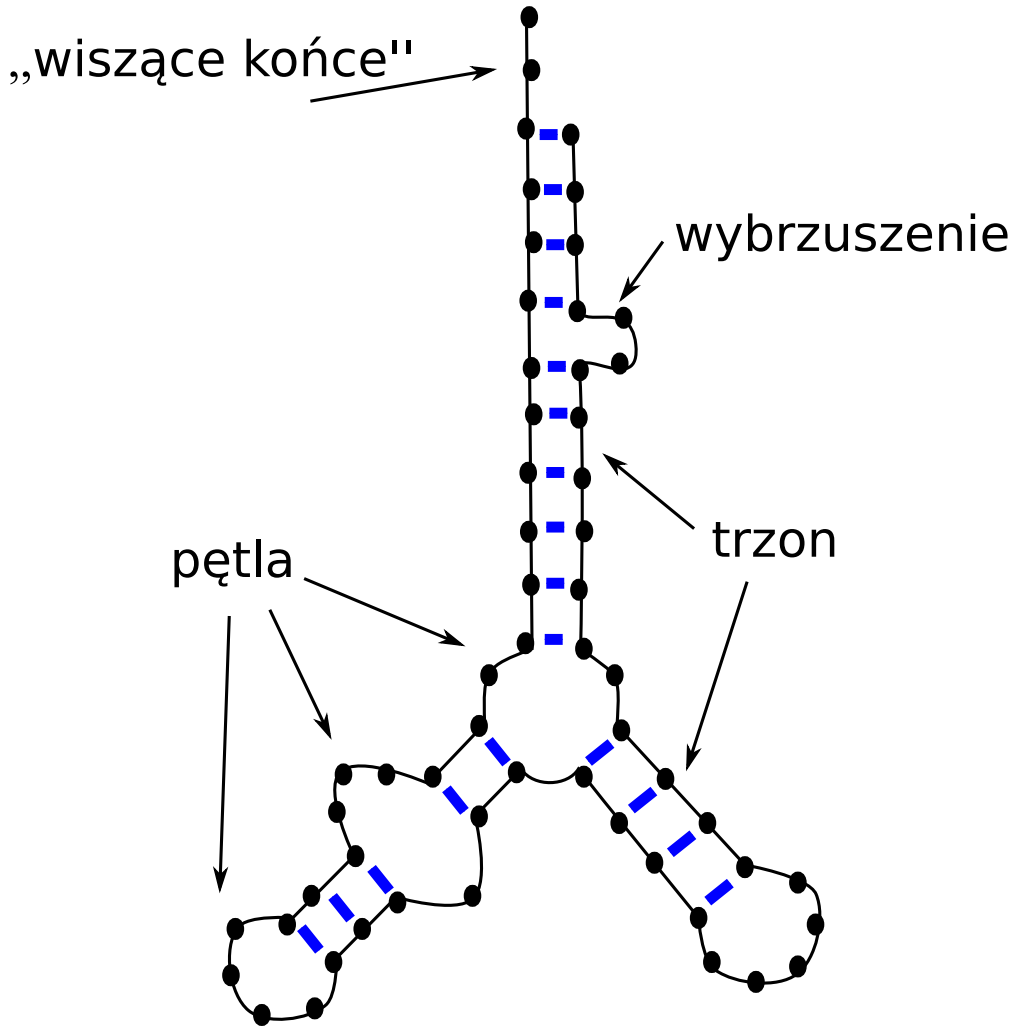
Rodzaje połączeń wewnątrz struktur drugorzędowych

trzon (ang. *(stem)*)

pętle:

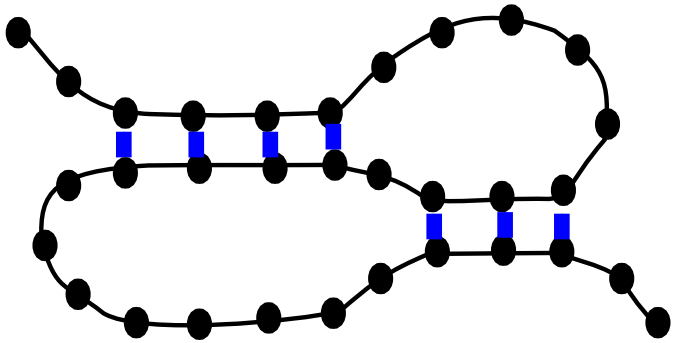
- wybrzuszenie (ang. *bulge loop*)
- pętla wewnętrzna (ang. *interior loop*)
- pętla zewnętrzna (ang. *hairpin loop*)
- pętla wieloramienna (ang. *multi-branched loop*)

wiszące końce

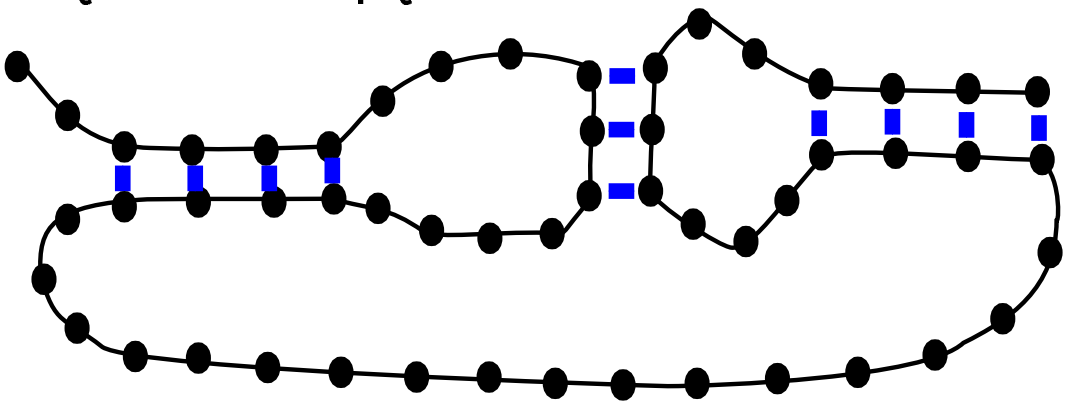


Struktury trzeciorzędowe - przykłady

- pseudo-węzły



- wiązanie dla pętli



Algorytmy oparte o minimalizację energii swobodnej

- cząsteczka przyjmuje strukturę o najniższej energii
- energia w zależności od siły wiązania dla poszczególnych par
- energia dla struktury: suma dla poszczególnych par

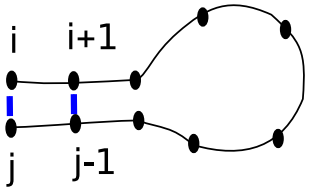
$$E(S) = \sum_{i,j \in S} e(x_i, x_j)$$

Przykładowa macierz energii:

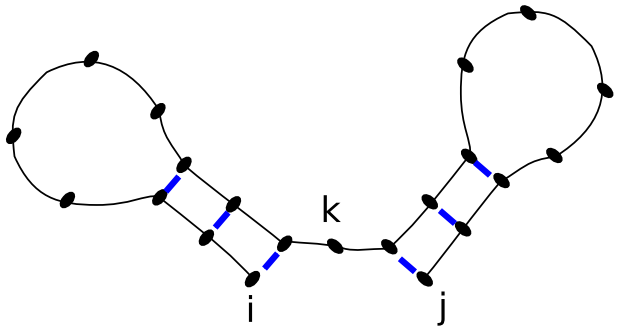
e(i,j)	A	C	G	U
A	0	0	0	2
C	0	0	3	0
G	0	3	0	1
U	2	0	1	0

Algorytm Nussinov

- inicjacja - pętle zewnętrzne nie mogą być krótsze niż 4 nukleotydy
- tworzenie pary

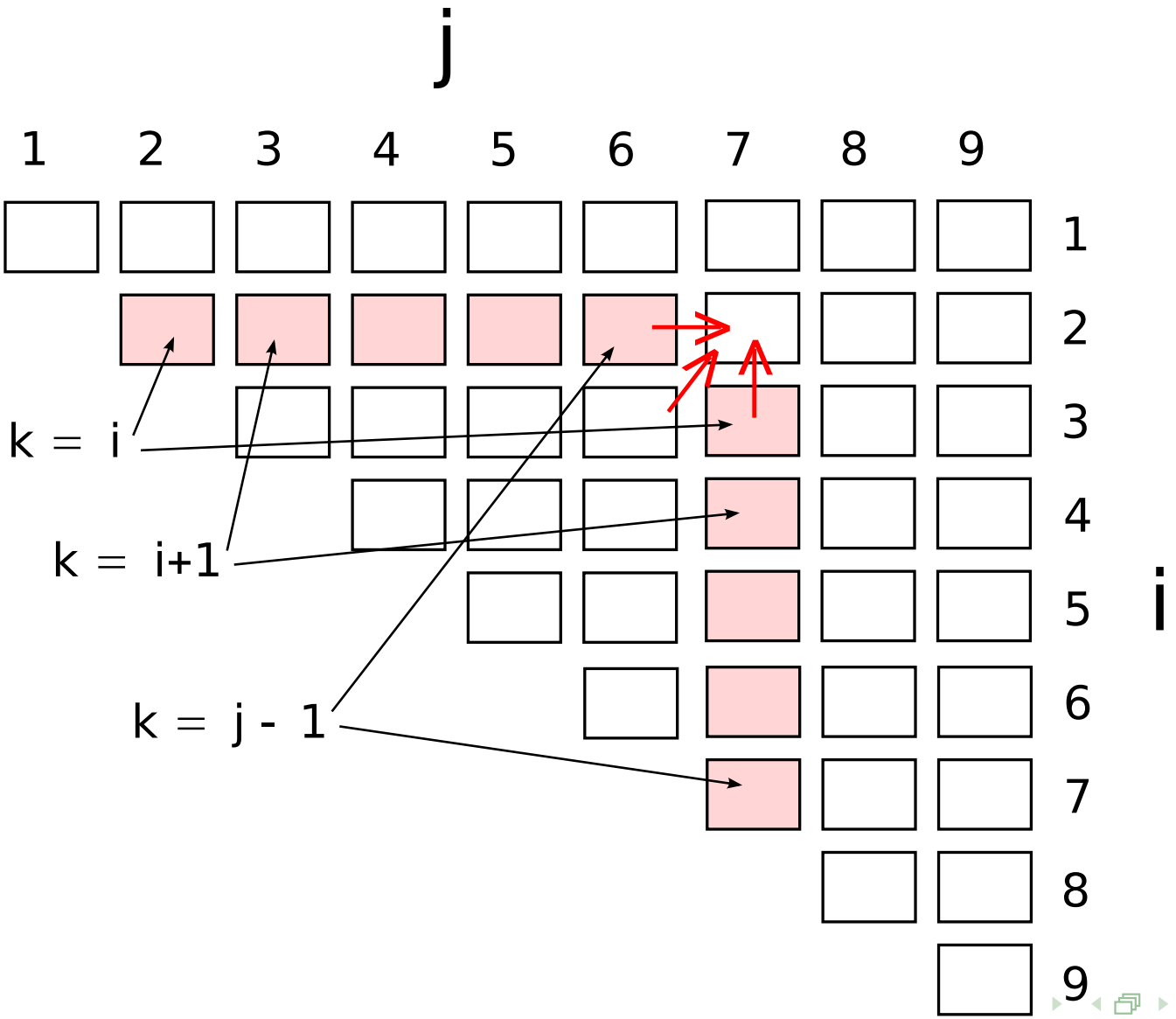


- podział sekwencji

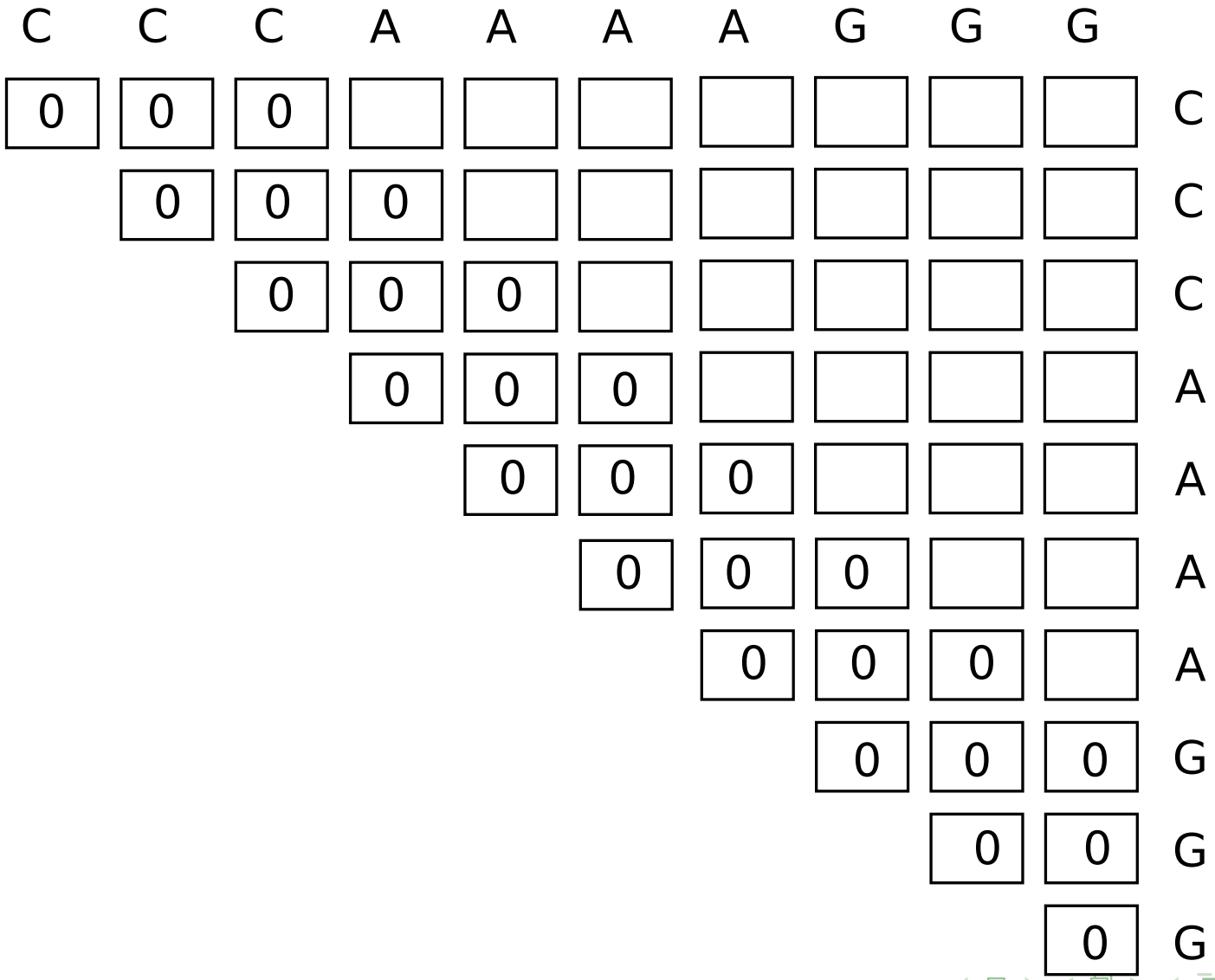


$$F(i,j) = \max \begin{cases} 0 & j - i \leq 3 \\ F(i+1, j-1) + e(x_i, x_j) & \text{połączenie} \\ F(i+1, j) & x_i \text{ bez pary} \\ F(i, j-1) & x_j \text{ bez pary} \\ \max_{k:i \leq k < j} F(i,k) + F(k+1,j) & \text{podział} \end{cases}$$

Algorytm Nussinov - przykład



Algorytm Nussinov - przykład

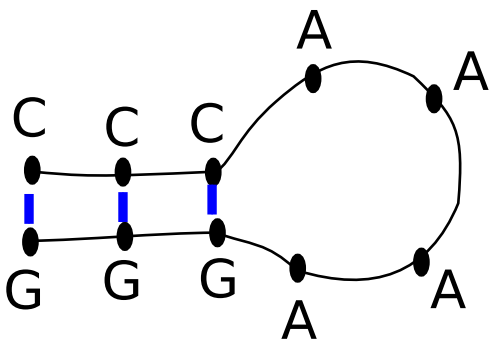


Algorytm Nussinov - przykład

C	C	C	A	A	A	A	G	G	G	
0	0	0	0	0	0	0	3	6	9	C
	0	0	0	0	0	0	3	6	6	C
		0	0	0	0	0	3	3	3	C
			0	0	0	0	0	0	0	A
				0	0	0	0	0	0	A
					0	0	0	0	0	A
						0	0	0	0	A
							0	0	0	G
								0	0	G
									0	G

Algorytm Nussinov - przykład

	C	C	C	A	A	A	A	G	G	G	
	0	0	0	0	0	0	0	3	6	9	C
		0	0	0	0	0	0	3	6	6	C
			0	0	0	0	0	3	3	3	C
				0	0	0	0	0	0	0	A
					0	0	0	0	0	0	A
						0	0	0	0	0	A
							0	0	0	0	A
								0	0	0	G
									0	0	G
										0	G

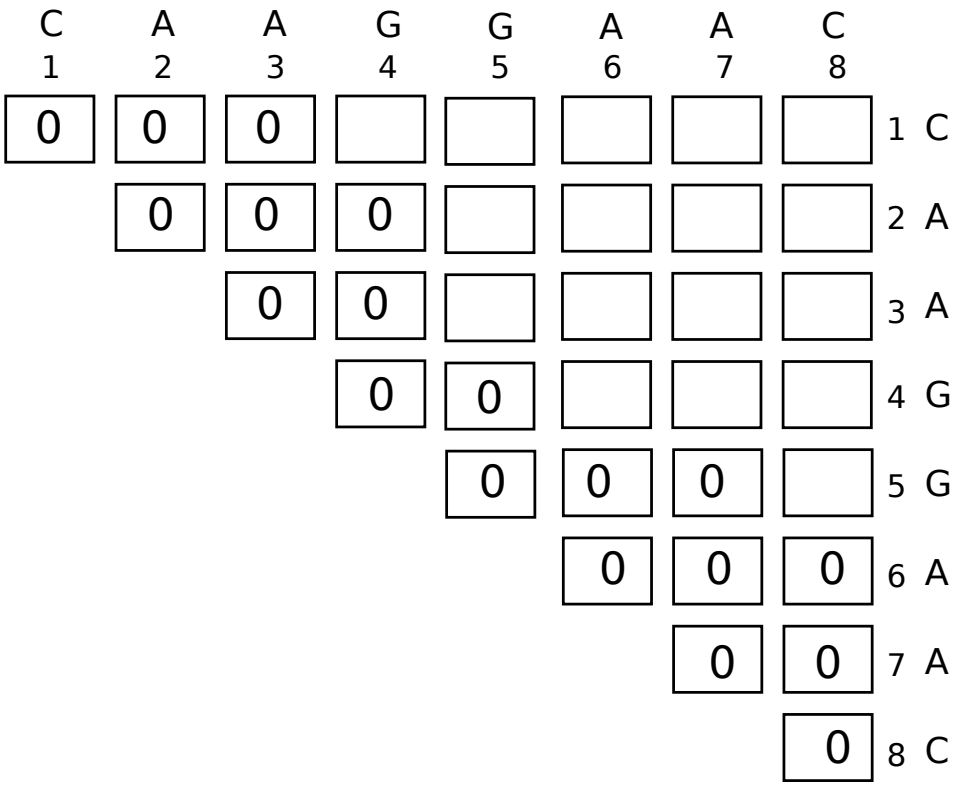


Algorytm Nussinov - przykład (2)

CAAGGAAC

	A	C	G	T
A	0	0	0	2
C	0	0	3	0
G	0	3	0	0
T	2	0	0	0

$$F(i,j) = \max \begin{cases} 0 & j - i \leq 2 \\ F(i+1, j-1) + e(x_i, x_j) \\ F(i+1, j) \\ F(i, j-1) \\ \max_{k:i \leq k < j} F(i,k) + F(k+1, j) \end{cases}$$

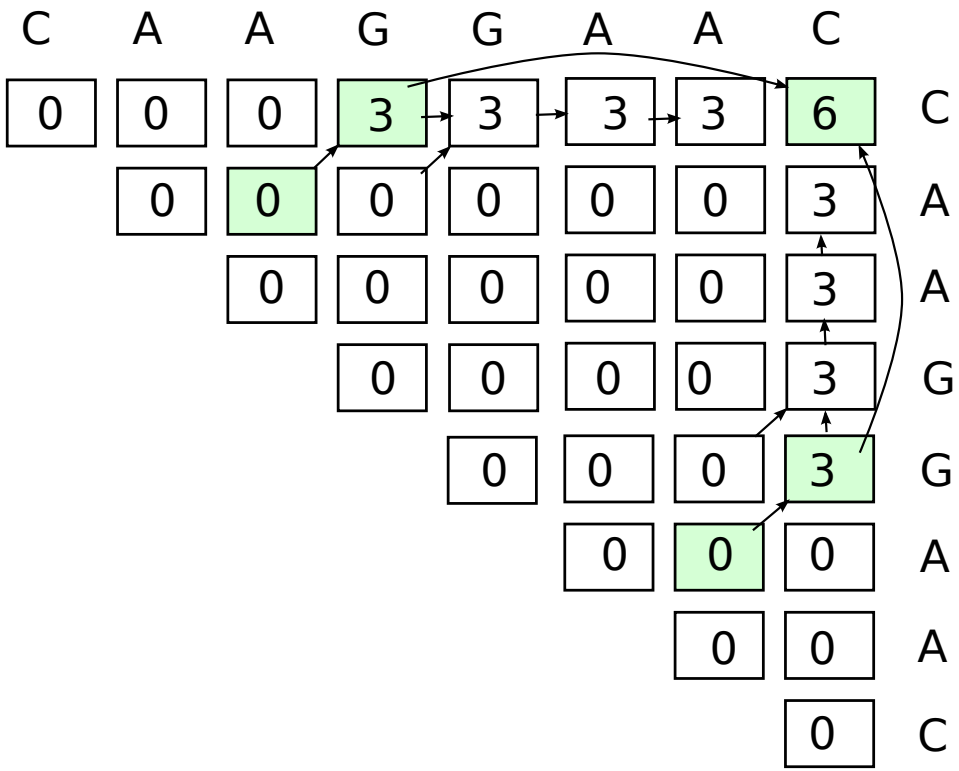


Algorytm Nussinov - przykład (2)

CAAGGAAC

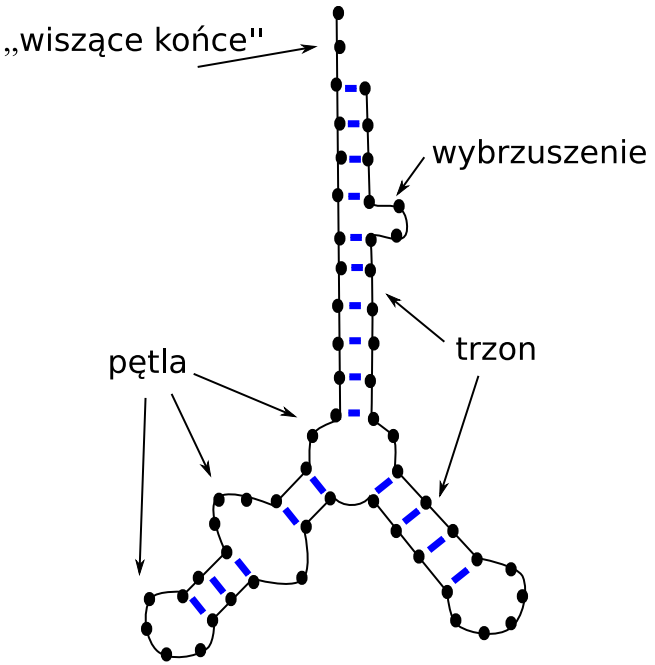
	A	C	G	T
A	0	0	0	2
C	0	0	3	0
G	0	3	0	0
T	2	0	0	0

$$F(i,j) = \max \begin{cases} 0 & j - i \leq 2 \\ F(i+1, j-1) + e(x_i, x_j) \\ F(i+1, j) \\ F(i, j-1) \\ \max_{k:i \leq k < j} F(i,k) + F(k+1, j) \end{cases}$$

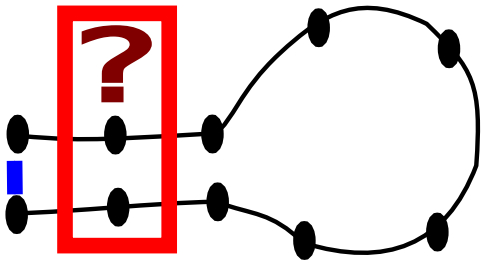


Algorytm Zukera

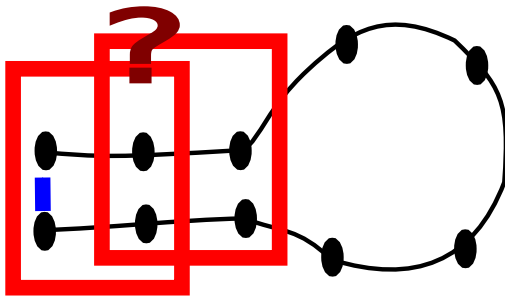
- elementy stabilizujące
  - trzonki
- elementy destabilizujące:
  - pętle zewnętrzne (spinki)
  - pętle wewnętrzne (w tym wybrzuszenia)
  - pętle wieloramienne



analizie podlegają pary par nukleotydów (a nie pojedyncze nukleotydy)



Nussinov



Zuker

Algorytm Zukera (2)

- wykorzystuje programowanie dynamiczne
- dostarczane są energie (wynikające z pomiarów) dla:
  - szpilek e_H
  - połączonych par e_S
  - pętli e_L
- oblicza macierz V - elementy (i, j) tworzą parę, oraz W - element (i, j) jest częścią pętli wieloramiennej

$$W(i,j) = \max \begin{cases} W(i+1,j) \\ W(i,j-1) \\ max_{k:i \leq k < j} W(i,k) + W(k+1,j) \\ V(i,j) \end{cases}$$

Algorytm Zukera (3)

$$V(i,j) = \max \begin{cases} e_H(i,j) \\ \text{ostatnie wiązanie przed szpilką} \\ \\ e_S(i,i+1,j,j-1) + V(i+1,j-1) \\ \text{trzonek (połączenie)} \\ \\ \max_{i<i'<j'<j : i'-i+j'-j>2} e_L(i,i',j,j') + V(i',j') \\ \text{pętla wewnętrzna} \end{cases}$$

Algorytmy minimalizacji energii swobodnej - podsumowanie

- Algorytm Nussinov
  - bada oddziaływania między pojedynczymi nukleotydami
  - uwzględnia ograniczenia dla 'szpilek'
- Algorytm Zukera
  - bada oddziaływania między dwiema parami nukleotydów
  - inaczej obliczana energia dla pętli, trzonka, itd.

Problemy

- duża średnia liczba sub-optymalnych struktur, nie pozwala uwzględniać wszystkich możliwości
- nie uwzględniane oddziaływania trzeciorzędowe (pseudo-węzły, itp.)
- łańcuchy RNA niekoniecznie muszą przyjmować strukturę o minimalnej energii, konformacja może być wymuszona kinetyką tworzenia struktury drugorzędowej

Dziękuję