ШИНЖЛЭХ УХААН ТЕХНОЛОГИЙН ИХ СУРГУУЛЬ
Мэдээлэл, Холбооны Технологийн Сургууль

F.CS213 Биоалгоритм

Pairwise Sequence Alignment

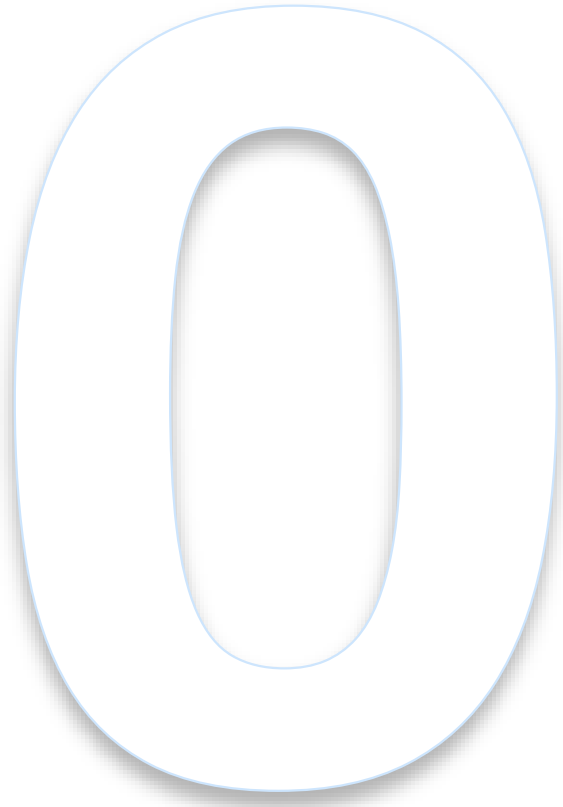
Хоёр дараалал зэрэгцүүлэлт

Лекц 5

Багш ***Ганбаатарын ГАНБАТ.*** *Өрөө* ***#304,*** *ganbatg@must.edu.mn,* ***99999467***

Лекцийн агуулга

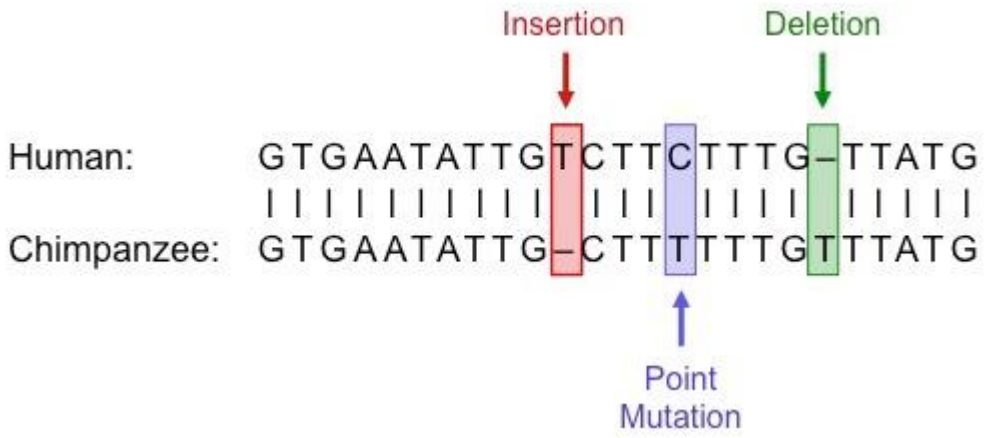
- Дарааллууд харьцуулах
- Дараалал зэрэгцүүлэлт
- Дараалал зэрэгцүүлэлт ба Оновчлолын асуудал
    - Зорилгын функц
    - Орлуулгын матриц
- Динамик программчлалын алгоритмууд
    - Глобал зэрэгцүүлэлт
    - Локал зэрэгцүүлэлт



Дарааллуудыг харьцуулах

Биоинформатикийн биологийн судалгаанд авчирсан гол давуу талуудын нэг нь ген болон уурагт кодлогдсон функцийг тайлбарлахад чиглэдэг.

- Дарааллын ***давхцал (similarity)*** дээр суурилан ***гомолог (homology)***-ийг тодруулах
  - Жнь: Гарал үүслийн хамаарал
  - Эдгээр ойлголтууд нь эквивалент биш
- Практикт
  - Давхцлын өндөр хувьтой бол гомолог буюу ижил төстэй үүргүүдтэй байх магадлал өндөр
  - Давхцлын хувь нэмэгдэх тусам энэ магадлал нэмэгддэг.
  - Туршилтаар баталгаажуулж шаардлагатай



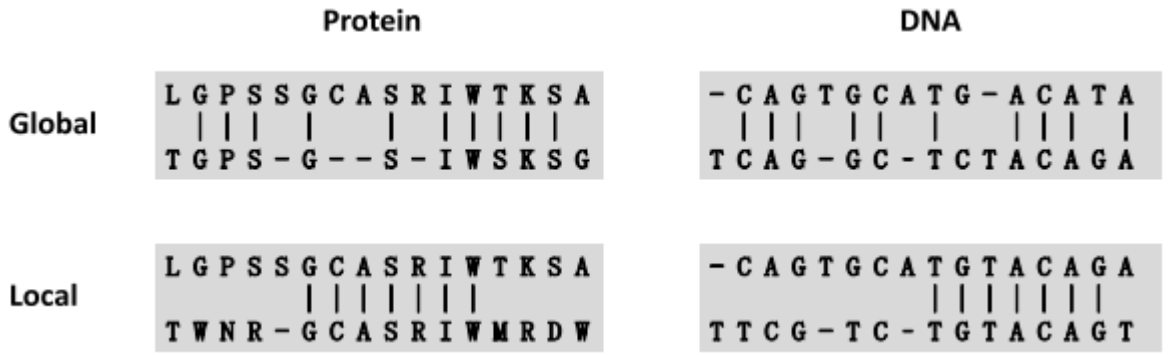
- Гарал үүслийн хувьд нийлдэг дарааллуудын хоорондох ялгааны эх үүсвэрийг авч үздэг: ***мутаци (mutations)***.
  - Нэг эсвэл хэд хэдэн нуклеотидоор өөр
  - Нуклеотидуудыг оруулах, устгах тохиолдол байдаг
  - Байрлал байрлалаар нь харьцуулах боломжгүй тул дарааллын харьцуулалтыг илүү төвөгтэй болгодог

ДАРААЛЛУУДЫГ ХАРЬЦУУЛАХ

Дараалал зэрэгцүүлэлт

Дарааллын *зэрэгцүүлэлт (alignment)*:
Тооцоолох процедур нь дарааллууд дээрх ижил дэсээр давхцаж буй тэмдэгтүүдийг хайдаг.

- *Тусгаарлагч (Gaps)* тэмдэгт хэрэглэн давхцлын тэмдэгтүүдийг улам олшруулдаг.
  - Энэ тохиолдолд ижил төстэй дараалал нь дарааллыг зөв тохируулсны дараа олон тооны ижил (эсвэл ижил) тэмдэгтүүдтэй байх болно.
  - Дарааллын зэрэгцүүлэлт, ижил төстэй байдлын ерөнхий санаа болдог
- ДНХ (эсвэл илүү ховор тохиолдолд РНХ), уураг.
  - Уургийн дараалал зэрэгцүүлэлт, үндсэндээ дарааллын аннотацын маш чухал хэрэгсэл
    - Жнь: дараалалд эсвэл дарааллын хэсгүүдэд үүрэг харгалзуулах
  - DNA дараалал зэрэгцүүлэлт, аннотацаас гадна полигентеик шинжилгээний хэрэглээнд ашиглагддаг.

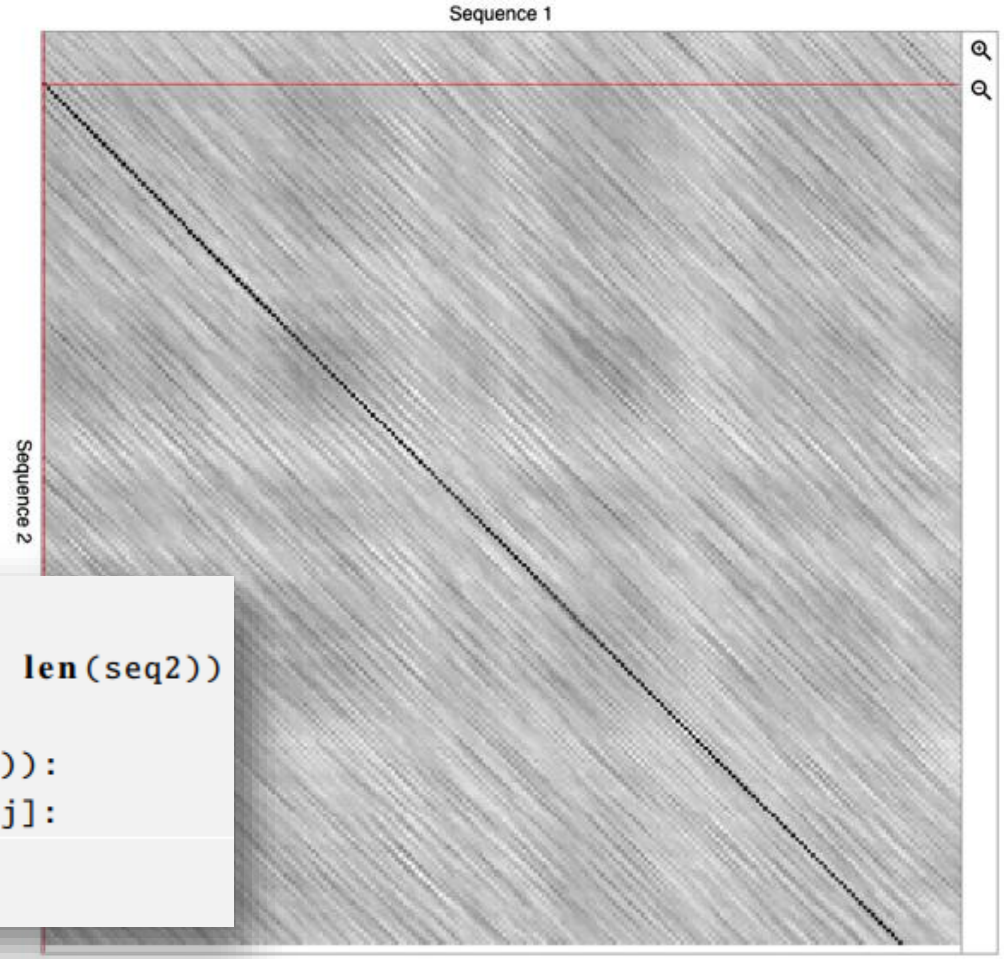


- *Глобал*: дарааллуудыг бүхэлд нь зэрэгцүүлэх
- *Локал*: хоёр дарааллын хамгийн сайн зэрэгцсэн хэсгийг олох зорилготой.

Визуал зэрэгцүүлэлт

Dot Plots: Хоёр дарааллын төстэй хэсгүүдийг визуалаар харах боломжтой

- Хоёр дарааллыг цэгийн матрицаар дүрслэх.
- Ижил төстэй тэмдэгтүүд хаана байгаа нь матрицын бүс дээр шууд харагдана,
- Өөр нэг чухал давуу тал бол нэг дараалал дээрх давталтыг хайх боломж.
  - Дараалал дээрх давталтыг диагоналдсан бараан хэсгүүдээр харж болно
  - Босоо/хэвтээ эгнээнүүдэд нэг тэмдэгтийн давталт.



dotlet хэрэгслээр зурсан цэгэн схем.

```
s1 = "CGATATAG"
s2 = "TATATATT"
mat1 = dotplot(s1, s2)
print_dotplot(mat1, s1, s2)
```

```
def create_mat(nrows, ncols):
    mat = []
    for i in range(nrows):
        mat.append([])
        for j in range(ncols):
            mat[i].append(0)
    return mat
```

```
def dotplot(seq1, seq2):
    mat = create_mat(len(seq1), len(seq2))
    for i in range(len(seq1)):
        for j in range(len(seq2)):
            if seq1[i] == seq2[j]:
                mat[i][j] = 1
    return mat
```

ДАРААЛЛУУДЫГ ХАРЬЦУУЛАХ

Өргөтгсөн Dot Plots

Ихэнх тохиолдолд өмнөх алгоритм нь “шуугиан (noise)” ихтэй тул үр дүнд нөлөөлөх нь түгээмэл.

- Матрицыг үүсгэхдээ дарааллын байрлал бүрийг агуулсан цонхны хувьд авч үзнэ
- Дарааллуудын харгалзах цонхнуудын ижил тэмдэгтийг тоолно.
- Хэрэв өгөгдсөн параметр болох stringency-ээс их бол л харгалзах нүдэнд тэмдэглэнэ, үгүй бол орхино.

```
s1 = "CGATATAGATT"
s2 = "TATATAGTAT"
mat2 = extended_dotplot(s1, s2, 5, 4)
print_dotplot(mat2, s1, s2)
```

```
def extended_dotplot (seq1, seq2, window, stringency):
    mat = create_mat(len(seq1), len(seq2))
    start = int(window/2)
    for i in range(start,len(seq1)-start):
        for j in range(start, len(seq2)-start):
            matches = 0
            l = j - start
            for k in range(i-start, i+start+1):
                if seq1[k] == seq2[l]: matches += 1
                l += 1
                if matches >= stringency: mat[i][j] = 1
    return mat
```

```
def print_dotplot(mat, s1, s2):
    import sys
    sys.stdout.write(" " + s2+"\n")
    for i in range(len(mat)):
        sys.stdout.write(s1[i])
        for j in range(len(mat[i])):
            if mat[i][j] >= 1:
                sys.stdout.write("*")
            else:
                sys.stdout.write(" ")
        sys.stdout.write("\n")
```

Дараалал зэрэгцүүлэлт ба Оновчлол

Дараалал зэрэгцүүлэлтийн процесс нь оновчлолын асуудал гэдгийг ойлгох хэрэгтэй. Ө/х хэд хэдэн боломжит шийдүүдээс хамгийн сайныг нь сонгодог.

- Дараалал зэрэгцүүлэлтийн хувьд боломжит шийдүүд нь дараалалд зай (gaps) оруулах байрлалуудын комбинац бна.
- Шийдүүдээс хамгийн сайныг нь сонгох буюу тус оновчлолын асуудлын хувьд оновчтой ***зорилгын функц (objective function)***-ийг тодорхойлох хэрэгтэй.
- Дараах байдлаар ***хос дараалал зэрэгцүүлэлтийн асуудал (pairwise sequence alignment problem)***-ыг тодорхойлно.
  - *Оролт:* Нэгдсэн цагаан толгой дээх хоёр дараалал; зэрэгцүүлэлтийн шийд бүрийг үнэлэх зорилтын функц;
  - *Гаралт:* Дараалалууд дээрх тэмдэгтүүд хоорондын оновчтой харгалзуулзалт (зайг тохиромжтой байрлалд оруулах)
- Боломжит шийдүүдийг тоолох байдлаар энэ асуудлын тооцооллын хүндрэлийг авч үзье.
  - Хоёр дараалал нь ижил n хэмжээтэй гэж үзье.
  - Аливаа байрлалын хувьд хоёр дараалал дээр зай (gaps) нь ижил байрлалд орохгүй учраас зэрэгцүүлэлт хамгийн уртдаа $2n$ урттай болно.

$$\binom{2n}{n} = \frac{(2n)!}{n!^2}$$

$n = 20$ үед ойролцоогоор 120 тэрбум шийд !!!

Зорилгын функц

Дараалал зэрэгцүүлэлтийн зорилгын функцэд *оноо (score)*-ны санааг авч үздэг: Ижил төстэй дараалал дээр зэрэгцүүлэлт их утгатай, эсрэг тохиолдолд бага.

- Ердийн зорилгын функц: Дарааллууд дээрх ижил тэмдэгтүүдийг тоолох.
  - *Ижил тэмдэгтэй (matches)* баганад 1, *ялгаатай тэмдэгттэй (mismatches)* болон ***зай (gaps)*** багананд 0 утга.
  - Зорилгын функцийн утга буюу оноо нь багана бүрийн хувьд тусгайлан тооцсон нийлбэр байна.

- Гурван тохиолдол бүрийн хувьд өөр өөр оноо өгснөөр зорилгын функцийн хувилбарууд үүсдэг.
  - *DNA/RNA: **Ижил тэмдэгттэй*** баганад +1 or +2 , ***Ялгаатай тэмдэгт*** болон ***зай*** багананд −3 or −2 утга..
  - Уургийн хувьд 20 амин хүчлийг биохимийн шинж чанараар нь бүлэглэдэг.
  - Ялгаатай тэмдэгттэй байхад хоёр амин хүчил нь шинж чанарын хувьд эрс ялгаатай байдаг учир тохиолдол бүрт өөр өөр оноо өгөх шаардлагатай болдог.

- Био дараалал зэрэгцүүлэлтийн зорилгын функц нь ихэвчлэн илүү ерөнхий, уян хатан байх үүднээс хоёр төрөлтэй.
  - *Орлуулгын матриц (substitution matrix):* Баганад нь зай ороогүй байх бүх тохиолдлын онооны утгыг агуулдаг.
    - Цагаан толгойн тэмдэгтүүдийн боломжит хос бүрт утга онооно
    - Жишиг зэрэгцүүлэлт (good alignment)-ийн ӨС ашиглан орлуулах матрицыг тооцоолдог
  - *Зайн торгуулийн (gap penalty) функц:* Цоорхой (эсвэл загвараас хамаарсан зайн дараалал)-г хэрхэн торгохыг тодорхойлдог.

Орлуулгын матриц

$$s(a,b) = round\left(2 \times \log_2 \frac{P_{(a,b)}}{p_a p_b}\right)$$

- s нь оноо
- a, b нь амин хүчлүүд
- $P_{(a,b)}$ нь a, b хосын тохиолдох магадлал
- p_a ба p_b нь a, b тохиолдох магадлал

- Жнь: 1000 хос амин хүчил агуулсан ӨС. SS, SL нь 40 ба 9 удаа тохиолддог, Давтамж: $S - 10\%$ ба $L - 15\%$
  - $score(S,S) = round\left(2 \times \log_2 \frac{40/1000}{0.1 \times 0.1}\right) = 4$
  - $score(S,L) = round\left(2 \times \log_2 \frac{9/1000}{0.1 \times 0.15}\right) = -1$
- *BLOSUM (BLOcks of Amino Acid SUbstitution Matrix) family*
  - *Blocks* ӨС-ийн локал зэрэгцүүлэлт дээр суурилдаг.
  - Суурилсан локал зэрэгцүүлэлтийн ижил төстэйн түвшингээс хамаарсан хэд хэдэн матрицтай.
  - Жнь, BLOSUM62 матрицад ижил төсөө нь 62%-иас дээш байх зэрэгцүүлэлт ашигласан.
- *PAM (Percent Accepted Mutations)*
  - Удамшлын зай (evolutionary distance) нь мэдэгдэж буй хамаарал бүхий уургийн бүлгээс гаргаж авсан.
  - Ойрын (илүү төстэй) дарааллыг харьцуулахын тулд жижиг тоог хэрэглэнэ.

	A	C	D	E	F	G	H	I	K	L	M	N	P	Q	R	S	T	V	W	Y
A	**4**																			
C	0	**9**																		
D	-2	-3	**6**																	
E	-1	-4	2	**5**																
F	-2	-2	-3	-3	**6**															
G	0	-3	-1	-2	-3	**6**														
H	-2	-3	1	0	-1	-2	**8**													
I	-1	-1	-3	-3	0	-4	-3	**4**												
K	-1	-3	-1	1	-3	-2	-1	-3	**5**											
L	-1	-1	-4	-3	0	-4	-3	2	-2	**4**										
M	-1	-1	-3	-2	0	-3	-2	1	-1	2	**5**									
N	-2	-3	1	0	-3	0	-1	-3	0	-3	-2	**6**								
P	-1	-3	-1	-1	-4	-2	-2	-3	-1	-3	-2	-1	**7**							
Q	-1	-3	0	2	-3	-2	0	-3	1	-2	0	0	-1	**5**						
R	-1	-3	-2	0	-3	-2	0	-3	2	-2	-1	0	-2	1	**5**					
S	1	-1	0	0	-2	0	-1	-2	0	-2	-1	1	-1	0	-1	**4**				
T	-1	-1	1	0	-2	1	0	-2	0	-2	-1	0	1	0	-1	1	**4**			
V	0	-1	-3	-2	-1	-3	-3	3	-2	1	1	-3	-2	-2	-3	-2	-2	**4**		
W	-3	-2	-4	-3	1	-2	-2	-3	-3	-2	-1	-4	-4	-2	-3	-3	-3	-3	**11**	
Y	-2	-2	-3	-2	3	-3	2	-1	-2	-1	-1	-2	-3	-1	-2	-2	-2	-1	2	**7**

BLOSUM62 орлуулах матриц

Зай торгуулийн (gap penalty) функц

- Зай гарсан багана бүрийн тогтмол торгуулийг тодорхойлсон g параметрийг ашиглана.
- Уургийн дараалал зэрэгцүүлэлтийн g-ийн ердийн (matched байх) утгууд нь DNA дараалал зэрэгцүүлэлт дээрх −7-оос −12, эсвэл −2-оос −3 хооронд хэлбэлзэж болно.
- *Affine gap penalty* загвар
  - Практикт өргөн хэрэглэгдэж байгаа илүү боловсронгуй хувилбар
  - Зай эхлэхэд маш их торгууль ногдуулдаг (зай эхллүүлэх торгууль g)
  - Зайг сунган нэмэх торгууль бага. (торгууль нь ихэвчлэн r = -1 эсвэл -2 байж болно)
- Жнь, BLOSUM62 орлуулгын матриц болон g = −8 ба r = −2 бүхий *affine gap penalty* загварыг харгалзан уураг дээрх глобал зэрэгцүүлэлтийн оноог тооцоолъё.

DNA

```
-CAGTGCATG-ACATA
 |||  ||  |   |||  |
TCAG-GC-TCTACAGA
```

$$score = -1+6+7+4-8+6-8-2+4-8+4+11+1+5+4+0 = 25$$

- Хэрэв тогтмол торгуултай энгийн загварыг ашиглсан бол 8 дахь баганын оноо −8 болж, нийт 19 оноотой болно.

Дараалал зэрэгцүүлэлтэд зориулсан динамик программчлал

Динамик программчлал (DP) бол **divide-and-conquer** аргад суурилсан оновчлолын алгоритмуудын ***ерөнхий зориулалтын (general-purpose)*** анги бөгөөд том хэмжээний асуудлыг шийдвэрлэхдээ дэд асуудлууд болон тэдгээрийн онооны шийдүүдийг дахин ашигладаг (дахин тооцоолдоггүй).

- Биоинформатикийн эхэн үед судлаачид биологийн дарааллын уялдаа холбоог шийдвэрлэхийн тулд DP алгоритмуудыг ашиглахыг санал болгосон.
- Дарааллууд дэд дарааллаас бүрдэх боломжтой гэж үзэн энэ санааг Хос дараалал зэрэгцүүлэлт хийхэд ашигладаг.
- Динамик программчлалын алгоритмууд
  - Глобал зэрэгцүүлэлт: The Needleman-Wunsch Algorithm
  - Локал зэрэгцүүлэлт: The Smith-Waterman Algorithm

Affine gap penalty загварын төстэй алгоритмууд зохиоход илүү хүндрэлтэй тул энгийн загварыг авч үзнэ.

Needleman-Wunsch алгоритм

		Sequence B					
			b_1	b_2	b_3	b_4	b_5
		gap	H	G	W	A	G
	gap	0	-8	-16	-24	-32	-40
a_1	P	-8	-2	-10	-18	-25	-33
a_2	H	-16	0	-4	-12	-20	-27
a_3	S	-24	-8	0	-7	-11	-19
a_4	W	-32	-16	-8	11	3	-5
a_5	G	-40	-24	-10	3	11	9

Sequence A

$A = PHSWG$ ба $B = HGWAG$.

- n ба m хэмжээтэй $A = a_1 a_2 \dots a_n$ ба $B = b_1 b_2 \dots b_m$ хоёр дарааллыг байг.
- A - мөр, B - баганатай S матрицыг байгуулна
  - Зай агуулсан зэрэгцүүлэлтийг илэрхийлэхийн тулд эхэнд нь нэмэлт мөр, багана оруулна.
  - $S_{i,j}$ элемент нь A-ийн $a_1 \dots a_i$ дэд дарааллыг, B-ийн $b_1 \dots b_j$ дэд дарааллыг зэрэгцүүлэх оновчтой оноо.

$$S_{i,j} = \max\left(S_{i-1,j-1} + sm(a_i, b_j), S_{i-1,j} + g, S_{i,j-1} + g\right)$$

- $sm(c_1, c_2)$ нь орлуулгын матрицын утга
- g нь торгуулийн утга, байрлал бүрт тогтмол
- Анхны утгууд нь $S_{i,0} = i * g$ ба $S_{0,j} = j * g$ байна.

Example:

$S_{1,1} = max\,(\,S_{0,0} + sm("H","P"),\ S_{0,1} + g,\ S_{1,0} + g\,) = max(\,0\text{-}2,\ -8\text{-}8,\ -8\text{-}8) = -2$

trace-back матриц.

	gap	H	G	W	A	G
gap	0	-8	-16	-24	-32	-40
P	-8	-2	-10	-18	-25	-33
H	-16	0	-4	-12	-20	-27
S	-24	-8	0	-7	-11	-19
W	-32	-16	-8	11	3	-5
G	-40	-24	-10	3	11	9

Best alignment:

P H S W - G
- H G W A G

Score of the best alignment:

-8 + 8 + 0 + 11 - 8 + 6 = 9

Смит-Уотерманы алгоритм

- Өмхөн алгоритмыг Локал зэрэгцүүлэлтэд тохируулсан
- Дарааллуудын дэд хэсгүүдийн хувьд хамгийн их байхаар буюу Зорилгын функц (оноо)-г их байхаар харгалзуулахыг хичээнэ.
- Өмнөх алгоритмтай харьцуулахад:
  - Оноо тооцоолох аргад ямар ч өөрчлөлт орохгүй
  - Сөрөг утга гарч ирэх хэсэг дээр нь зэрэгцүүлэлтийг эхлүүлнэ.
  - Ингэхдээ рекуррент харьцаанд 0 утгыг нэмж оруулна.

	gap	H	G	W	A	G
gap	0	0	0	0	0	0
P	0	0	0	0	0	0
H	0	8	0	0	0	0
S	0	0	8	0	1	0
W	0	0	0	19	11	3
G	0	0	6	11	19	17

$A = PHSWG$ ба $B = HGWAG.$

$$S_{i,j} = \max(S_{i-1,j-1} + sm(a_i, b_j), S_{i-1,j} + g, S_{i,j-1} + g, 0)$$

- $sm(c_1, c_2)$ нь орлуулгын матрицын утга
- g нь торгуулийн утга, байрлал бүрт тогтмол
- Анхны утгууд нь $S_{i,0} = i * g$ ба $S_{0,j} = j * g$ байна.

Examples:

$S_{1,1} = max(\ S_{0,0} + sm("H","P"),\ S_{0,1} + g,\ S_{1,0} + g,\ 0\ ) = max(\ 0\text{-}2,\ \text{-}8\text{-}8,\ \text{-}8\text{-}8,\ 0\ ) = 0$

$S_{4,3} = max(\ S_{3,2} + sm("H","P"),\ S_{3,3} + g,\ S_{4,2} + g,\ 0\ ) = max(\ 8\text{+}11,\ 0\text{-}8,\ 0\text{-}8,\ 0\ ) = 19$

trace-back матриц.

	gap	H	G	W	A	G
gap	0	0	0	0	0	0
P	0	0	0	0	0	0
H	0	8	0	0	0	0
S	0	0	8	0	1	0
W	0	0	0	19	11	3
G	0	0	6	11	19	17

Best alignments:

H S W
H G W

H S W G
H G W A

АНХААРАЛ ТАВЬСАНД БАЯРЛАЛАА