ШИНЖЛЭХ УХААН ТЕХНОЛОГИЙН ИХ СУРГУУЛЬ
Мэдээлэл, Холбооны Технологийн Сургууль

F.CS213 Биоалгоритм

Searching Similar Sequences in Databases

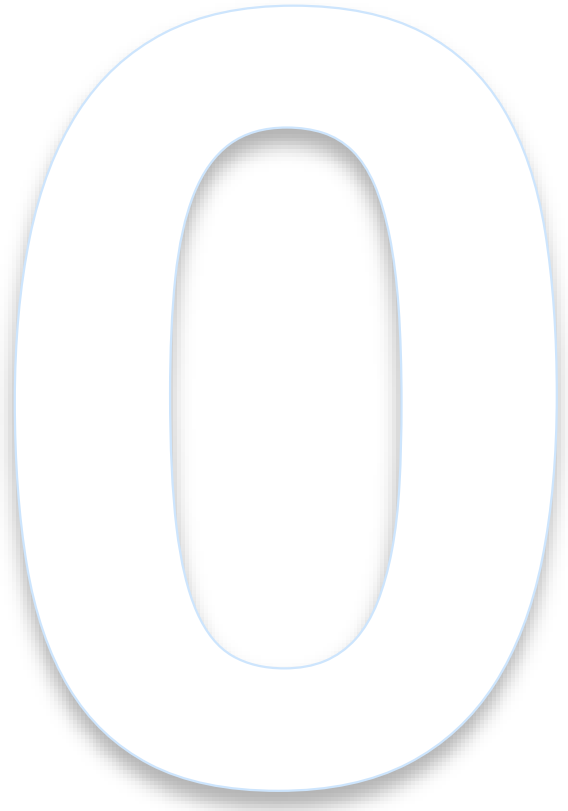
Өгөгдлийн сангаас ижил төстэй дарааллуудыг хайх

Лекц 6

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

Лекцийн агуулга

- Introduction
- Basic Local Alignment Search Tool
    - BLAST Algorithm
    - BLAST Programs
- Implementing Our Own BLAST
- Using BLAST Through BioPython



Удиртгал

Биологийн судалгаанд ямар ч мэдээлэлгүй байх ДНХ эсвэл уургийн дараалалтай элбэг тааралддаг (жнь, ДНХ-ийн дараалал тогтоох төслүүдээс ирдэг).

- Эдгээр дарааллын талаар таамаглал дэвшүүлж, тайлбар (аннотац) өгөхийг зорино
- Тухайн үүргээр өмнө нь тайлбарлагдсан дараалалтай ижил төстэй болохыг тогтооно.
- Ийм дарааллыг агуулсан байж болох ӨС-уудыг шүүнэ
  - Уг ӨС-ийн бүх дараалалтай өгөгдсөн дарааллыг Хос дараалал зэрэгцүүлэлтийн алгоритмуудаар харьцуулна
  - Ижил төстэй байдлын түвшинг аль болох өндөр байлгахыг зорино.
- Дараах параметрүүд нь эцсийн үр дүн (ижил төстэй байдлыг харуулсан дарааллууд)-д нөлөөлнө.
  - Зэрэгцүүлэлтийн төрөл (локал эсвэл глобал),
  - Зөрүүний торгуулийн загвар ба параметрүүд,
  - Тохиромжтой орлуулах матриц (эсвэл ижил/ялгаатайн оноо).

УДИРТГАЛ Тооцооллын хүндрэл

Дарааллаар зэрэгцүүлэлтийн DP алгоритмуудыг цөөн удаа ажиллуулахад дүнтэй гэж үзэж болох ч олон удаа (сая сая удаа) ажиллуулах шаардлагатай үед тохиромжгүй болдог.

- *Оролт*: Өгөгдсөн дараалал, Дарааллын жагсаалт, Орлуулах матриц, Зөрүүний торгуул (тогтмол байхаар),
- *Гаралт:* Өгөгдсөн дараалалын тухайн ӨС (ls) дахь хамгийн сайн локал зэрэгцүүлэлт болон дараалал

```
def align_query (query, ls, sm, g):
    bestScore = -1
    bestSeq = None
    bestAl = None
    for seq in ls:
        al = smith_Waterman(query, seq, sm, g)
        if al[2] > bestScore:
            bestScore = al[2]
            bestSeq = seq
            bestAl = al
    bestAlin = recover_align_local(bestAl[0], bestAl[1], query,
   bestSeq)
    return bestAlin, bestScore
```

- Квадрат зэргийн хүндрэл үүсдэгы.
  - 2 матриц: Оноо, trace-back
  - Тус бүр нь тухайн дарааллын уртаар хэмжигдэх 2 хэмжээст матрицууд
- ӨС-гийн дараалал бүрийн хувьд энэ алгоритмыг ажиллуулах тул нийт дарааллын тоогоор үржүүлнэ.

Практикт эдгээр харьцуулалтыг хийхэд ӨС нь ихэвчлэн том хэмжээтэй, хэдэн сая дараалал агуулсан байдаг

УДИРТГАЛ

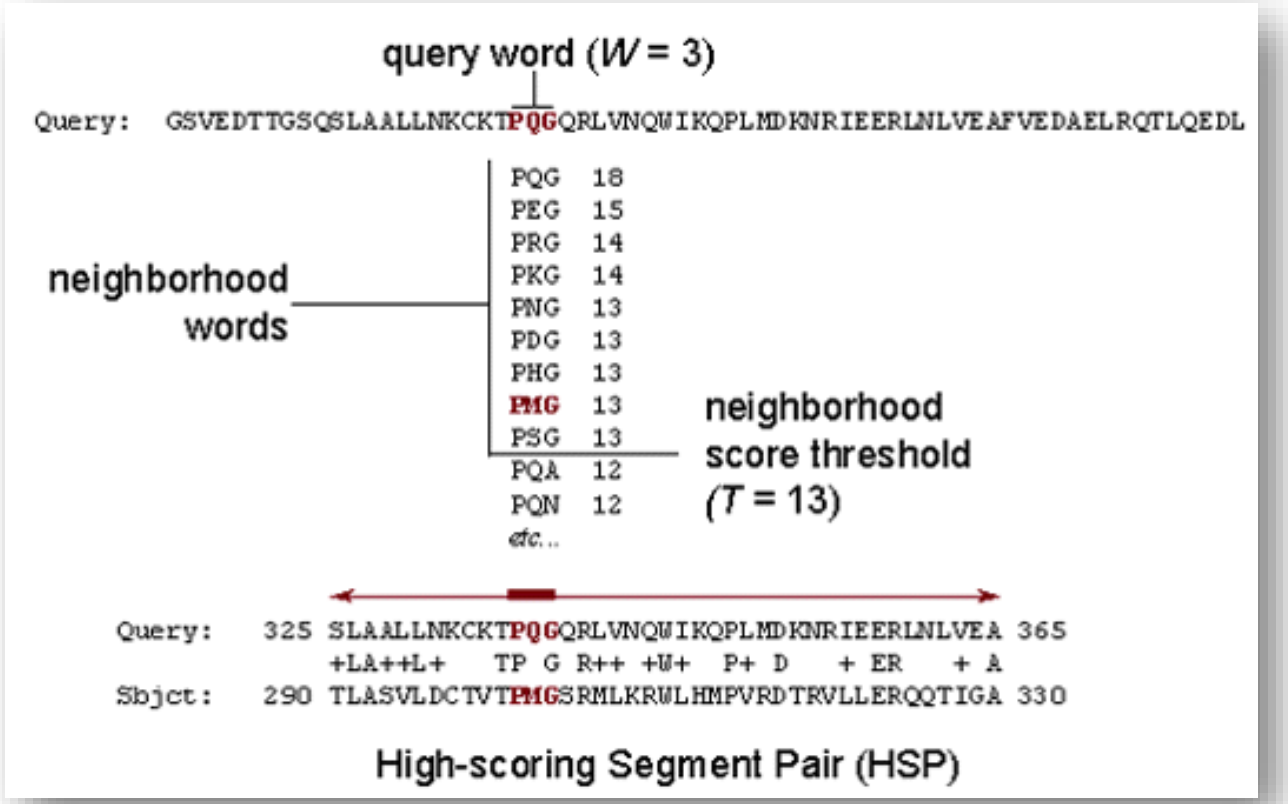
Хьюристик хандлагатай алгоритмууд

Тооцооллын хүндрэлийн асуудлыг ерөнхийдөө DP алгоритмаас 100 дахин хурдан байж болох хьюристик хандлагатай зарим алгоритм (программ)-ыг хөгжүүлэн шийддэг.

- Жишээ нь: Урьдчилсан боловсруулалттай буюу хайлт хийхэд тохируулсан бүтэцтэй мэдээллийг бүрдүүлэх боловсруулалтыг эхлээд бэлтгэдэг
- Биоинформатикийн нийгэмлэгээс хэд хэдэн хьюристик алгоритмуудыг санал болгосон.
  - FASTA: Хамгийн алдартай нь, Биоинформатикийн эхэн үед гарч ирсэн
  - BLAST (Basic Local Alignment Search Tool) програм
    - NCBI эсвэл EBI зэрэг үндсэн судалгааны хүрээлэнгүүдийг хамарсан олон сервер
    - Нээлттэй ашиглах боломжтой, маш өргөн хэрэглэгддэг.
- Дутагдалтай тал:
  - Алгоритмын зорилго сулардаг, ө.х эдгээр алгоритмууд нь хамгийн сайн шийдлийг (зэрэгцүүлэлт) олох баталгаа болохгүй.
  - Ялангуяа дараалын ижил төстэй байдал бага үед асуудал үүсч болно,
  - Өгөгдсөн өгөгдлийн санд ижил төстэй байдлын өндөр түвшний дараалал байхгүй.

Basic Local Alignment Search Tool - BLAST

BLAST алгоритм: Өгөгдсөн дараалал болон сонгосон ӨС-ийн дарааллуудын хооронд сайн зэрэгцүүлэлт олох зорилготой.



1. Зэрэгцүүлэлтийн чанарыг бүдгэрүүлж болзошгүй нөлөө багатай бүсүүдийг өгөгдсөн дарааллаас хасна;
2. Өгөгдсөн дараалал дээрх w урттай бүх *"үг (word)"* -ийг гарган авна;
3. Дээрх үг бүрийн хувьд w урттай бүх хөрш үгийн жагсаалт байгуулна (BLOSUM62, PAM);
   - Зэрэгцүүлэлтийн оноо нь T-ээс их
4. ӨС-ийн дараалал бүр дээр уг үгсийг хайна.
   - w уртаар таарч байгаа (*hits*);
5. Бүх hit-ийг хоёр тийш нь өргөтгөнө;
6. Хамгийн өндөр оноотой зэрэгцүүлэлтүүдийг сонгоно (high-scoring pairs-HSPs).

BLAST Программууд https://www.ncbi.nlm.nih.gov/BLAST

BLASTN: Нуклейтид хайлт

- RefSeq зэрэг Нуклейтидийн олон ӨС-аас ижил төстэй дарааллыг хайх, тухайн зүйлийн геномоор шүүх
- Их төстэй дарааллуудын хайлтыг оновчлох, урт дарааллыг хурдан хайх боломжийг олгодог (megablast программыг ашиглан).
- Параметерүүд
  - үгийн урт w, анхны утга 11
  - онооны функц, ижил/ялгаатай (анхны утга 2 ба -3)
  - нээх/өргөтгөх зай-торгууль (анхны утга -5 ба -2).

BLASTP: Уургийн хайлт

- RefSeq, UniProt, PDB зэрэг ӨС.
- Өөр өөр анхны утгатай параметрүүд. Жнь $w = 6$ бол:
  - онооны функц нь орлуулах матрицыг ашигладаг (анхны утга BLOSUM62)
  - нээх/өргөтгөх зай-торгууль (анхны утга -11 ба -1).
- Бусад : PSI-BLAST, PHI-BLAST, DELTA-BLAST.

BLASTX

- ДНХ-ийн дараалал авч уургийн дарааллууд дээр хайлт хийдэг.
- Нуклеотидын дарааланд кодлогдсон боломжит уургийг олоход ашиглаж болно;

TBLASTN

- Уургийн дараалал авч ДНХ-ийн ӨС-д хайлт хийдэг.
- Оролттой төстэй уургийг кодлох боломжтой дарааллуудыг ӨС-аас тодорхойлохыг оролддог;

TBLASTX

- ДНХ-ийн дараалал авч ДНХ-ийн ӨС-д хайлт хийдэг
- Уншилтын бүх 6 фрэймээр ДНХ дараалал хайхаас гадна Уургийн дарааллууд дээр хайлт хийдэг.
- Энэхүү кодчилолын боломжид суурилан оролттой төстэй нуклеотид дарааллыг тодорхойлж болно.

Зэрэгцүүлэлтийн үнэмшил (Significance)

Зэрэгцүүлэлтийн оролт ба дарааллын ижил төсөө нь статистикийн хувьд бодитой уу? Ө.х энэ ижил төстэй байдал нь санамсаргүй тохиолдлоор үүсэх магадлал хэр байх вэ?

- Орлуулах матриц / зай-торгууль дээр суурилсан BLAST болон DP-ын онооны функц нь өөр өөр урттай дарааллуудыг харьцуулахад ашиглах боломжгүй.
- Оролтын hit-үүдийг ялгаатай урттай дарааллуудтай харьцуулахдаа оноог нормалчлах шаардлагатай.
  - BLAST нь мэдээллийн онолын хэмжүүрүүдийг ашиглан нормалчилсан оноог тооцоолдог
  - Дарааллын урт руу нормалчилсан эдгээр оноо нь дурын төрлийн зэрэгцүүлэлтийг харьцуулахад хангалтгүй.
- BLAST дахь HSP тус бүрийн үр дүнгээр илрэх өгөгдсөн зэрэгцүүлэлтийн үнэмшлийг үнэлэхдээ E утгыг хэрэглэдэг.
  - Тухайн HSP-ийн өгч байгаагаас багагүй оноотой зэрэгцүүлэлтүүдийн тоо.
  - Зэрэгцүүлэлтийн оноо, ӨС-ийн хэмжээ, оролтын урт, зэрэгцүүлэлтийн параметрүүдийг харгалзан тооцоолно.
  - E-ийн утга бага байх тусам (0 рүү ойртох тусам) тохирох (match) илүү үнэмшилтэй болно.
- Зэрэгцүүлэлт нь гомолог болж байгаа эсэхийг шалгахдаа E утгаас гадна бусад үр дүнгүүд дээр анхаарах.
  - BLAST нь локал зэрэгцүүлэлтийг буцаадаг тул зэрэгцүүлэлтийн хамрах хүрээг оролт (түүнчлэн олдсон дараалал) дээр ажиглах.
  - Нийт дарааллын аль хэсгүүд нь байж болох зэрэгцүүлэлтийг багтааж болохыг анхаарах.
  - Дарааллуудын хэсгүүдэд (уургийн домайн эсвэл бусад локал паттерн) глобал гомолог байна уу эсвэл зөвхөн ижил төсөөтэй байна уу гэдгийг ойлгоход тусална.

BLAST

Энгийн жишээ - MyBlast

1. Мөр бүртээ дараалал агуулсан текст файлаас ӨС-г ачаална.
2. Урьдчилан боловсруулалт хийнэ.
   - Түлхүүрүүд нь дараалал дээр байж болох w урттай бүх үг, утга нь тухайн үгний илэрч байгаа байрлал байх dictionary үүсгэнэ (*hashing*).
   - Орлуулах матрицыг ашиглахгүй. Харин ижил/ялгаатайн оноо (1/0)-г авч үзнэ. Энд зай тооцохгүй. Тиймээс зөвхөн төгс hit-үүдыг авч үзнэ, ө.х босго оноо w-тэй тэнцүү.
3. Дарааллаас ороллтой тохирох үгүүдийн бүх илрэлийг хайна
   - Үр дүн нь hit-үүдийн жагсаалт (оролт дах индекс, дараалалд илрэх индекс).
4. Өмнөх функцээр олсон hit-үүдийг өргөтгөнө
   - Hit-ийг хоёр тийш сунгаж, онооны өсөлтөд оруулсан хувь нэмэр нь өргөтгөлийн байрлалын тал хувиас их буюу тэнцүү байх болно гэдгийг харгалзан процессийг хялбаршуулсан.
   - Үр дүн нь дараах талбаруудтай хослол хэлбэртэй: оролт дахь зэрэгцүүэлтийн эхлэл индекс, дараалал дээрх зэрэгцүүлэлтийн эхлэл индекс, зэрэгцүүлгийн хэмжээ, оноо (жнь ижил байгаа тэмдэгтүүдийн тоо).
5. ӨС-ийн бүх дараалалтай асуулгыг харьцуулахын тулд өмнөх функцуудыг ажиллуулна.

Энгийн жишээ - MyBlast

```
def build_map (query, w):
    res = {}
    for i in range(len(query)-w+1):
        subseq = query[i:i+w]
        if subseq in res:
            res[subseq].append(i)
        else:
            res[subseq] = [i]
    return res
```

1. Мөр бүртээ дараалал агуулсан текст файлаас ӨС-г ачаална.
2. Урьдчилан боловсруулалт хийнэ.
   - Түлхүүрүүд нь дараалал дээр байж болох w урттай бүх үг, утга нь тухайн үгний илэрч байгаа байрлал байх dictionary үүсгэнэ (*hashing*).
   - Орлуулах матрицыг ашиглахгүй. Харин ижил/ялгаатайн оноо (1/0)-г авч үзнэ. Энд зай тооцохгүй. Тиймээс зөвхөн төгс hit-үүдыг авч үзнэ, ө.х босго оноо w-тэй тэнцүү.
3. Дарааллаас ороллтой тохирох үгүүдийн бүх илрэлийг хайна
   - Үр дүн нь hit-үүдийн жагсаалт (ороллт дах индекс, дараалалд илрэх индекс).
4. Өмнөх функцээр олсон hit-үүдийг өргөтгөнө
   - Hit-ийг хоёр тийш сунгаж, онооны өсөлтөд оруулсан хувь нэмэр нь өргөтгөлийн байрлалын тал хувиас их буюу тэнцүү байх болно гэдгийг харгалзан процессийг хялбаршуулсан.
   - Үр дүн нь дараах талбаруудтай хослол хэлбэртэй: ороллт дахь зэрэгцүүлтийн эхлэл индекс, дараалал дээрх зэрэгцүүлэлтийн эхлэл индекс, зэрэгцүүлгийн хэмжээ, оноо (жнь ижил байгаа тэмдэгтүүдийн тоо).
5. ӨС-ийн бүх дараалалтай асуулгыг харьцуулахын тулд өмнөх функцуудыг ажиллуулна.

MyBlast

Dictionary үүсгэх, Оролт дээрх үгүүдийг тодорхойлох

```
def build_map (query, w):
    res = {}
    for i in range(len(query)-w+1):
        subseq = query[i:i+w]
        if subseq in res:
            res[subseq].append(i)
        else:
            res[subseq] = [i]
    return res
```

```
def get_hits (seq, m, w):
    res = []  # list of tuples
    for i in range(len(seq)-w+1):
        subseq = seq[i:i+w]
        if subseq in m:
            l = m[subseq]
            for ind in l:
                res.append( (ind,i) )
    return res
```

MyBlast

Hit-ийг өргөтгөх

```
def extends_hit (seq, hit, query, w):
    stq, sts = hit[0], hit[1]
    ## move forward
    matfw = 0
    k=0
    bestk = 0
    while 2*matfw >= k and stq+w+k < len(query) and sts+w+k < len(seq
   ):
        if query[stq+w+k] == seq[sts+w+k]:
            matfw+=1
            bestk = k+1
        k += 1
    size = w + bestk
    ## move backwards
    k = 0
    matbw = 0
    bestk = 0
    while 2*matbw >= k and stq > k and sts > k:
        if query[stq-k-1] == seq[sts-k-1]:
            matbw+=1
            bestk = k+1
        k+=1
    size += bestk

    return (stq-bestk, sts-bestk, size, w+matfw+matbw)
```

MyBlast

Хамгийн сайн оноо ба зэрэгцүүлэлт

```
def hit_best_score(seq, query, m, w):
    hits = get_hits(seq, m, w)
    bestScore = -1.0
    best = ()
    for h in hits:
        ext = extends_hit(seq, h, query, w)
        score = ext[3]
        if score > bestScore or (score== bestScore and ext[2] < best
    [2]):
            bestScore = score
            best = ext
    return best
```

```
def best_alignment (db, query, w):
    m = build_map(query, w)
    bestScore = -1.0
    res = (0,0,0,0,0)
    for k in range(0,len(db)):
        bestSeq = hit_best_score(db[k], query, m, w)
        if bestSeq != ():
            score = bestSeq[3]
            if score > bestScore or (score== bestScore and bestSeq[2]
    < res[2]):
                bestScore = score
                res = bestSeq[0], bestSeq[1], bestSeq[2], bestSeq[3],
    k
    if bestScore < 0: return ()
    else: return res
```

BioPython хэрэглэх

Модуль: Bio.Blast.NCBIWWW

- Цөм функц – **qblast.** Параметерууд:
  - **string Program** : Хэрэглэх программ (*"blastn"*, *"blastp"*, *"blastx"*, *"tblastn"* or *"tblastx"*)
  - **string Database** : ӨС (*"nr"*, *"nt"* or *"swissprot"*)
  - **string Query** : Оролтын дараалал(FASTA эсвэл **NCBI**-аас заасан форматтай дараалал)
  - Optional parameters: Гаралтын төрөл (xml г.м), E-ийн босго, Орлуулгын матриц, Зай-торгууль г.м

```
>>> from Bio.Blast import NCBIWWW
>>> from Bio import SeqIO
>>> record = SeqIO.read(open("example_blast.fasta"), format="fasta")
>>> result_handle = NCBIWWW.qblast("blastn", "nt", record.format("
    fasta"))
>>> save_file = open("my_blast.xml", "w")
>>> save_file.write(result_handle.read())
>>> save_file.close()
>>> result_handle.close()
```

BlastRecord class

BlastRecord ангийн объект нь BLAST хайлтын үр дүнгээс олж болох бүх мэдээлэл, мөн процесст ашигласан параметрүүдийг агуулдаг.

- Гурван түвшний шаталсан зохион байгуулалттай.
  - BlastRecord-ийн түвшин: ***alignments*** жагсаалт, мөн хайлтанд ашигласан ***matrix*** *(орлуулах матриц)*, ***gap_penalties***, ***database*** ерөнхий параметрүүд.
  - *Alignment* түвшин: ***hsps*** (**HSP**-ийн багц, өөр өөр HSP-ээр нэг дараалал дээр өөр өөр байршил олдоно), мөн *title, accession, hit_id, hit_def, length*.
  - **HSP** түвшин: *expect, score, query_start, sbjct_start, align_length, sbjct* локал зэрэгцүүлэлтийн мэдээллүүд, Мөн match (E утга), нормчилсон оноо, зэрэгцүүлэлт эхэлсэн оролт дээрх индекс, зэрэгцүүлэлт эхлэх дараалал дээрх индекс, HSP урт, HSP зэрэгцүүлэлтийн дэд хэсгүүд (оролтын болон тухайн дарааллын 2-ын ижил төстэй).

```
>>> from Bio.Blast import NCBIXML
>>> blast_record = NCBIXML.read(result_handle)

>>> blast_records = NCBIXML.parse(result_handle)
>>> for blast_record in blast_records:
```

АНХААРАЛ ТАВЬСАНД БАЯРЛАЛАА