DNA 데이터로 가족찾기

Hyungyong Kim, Insilicogen, Inc.
2014-08-30

- 김형용
- Twitter: @yong27
- Blog & Wiki: http://e.biohackers.net
- Bioinformatics concerned development in Insilicogen, Inc. (**We are hiring!**)

- XPer
- Django, Twisted, Biopython, Scipy, wxPython, pandas, scikit-learn

	3/8		
	7/12	—	not match
	3/11	—	
9	7/14	—	
9	5/5		
/8			

n fingerprint of the suspect Park Hyun Kyu does not
exactly to that semen sample found in the victim's body.
e said conclusively that the suspect is the murderer.
on e-1 to e-5 are returned herewith.

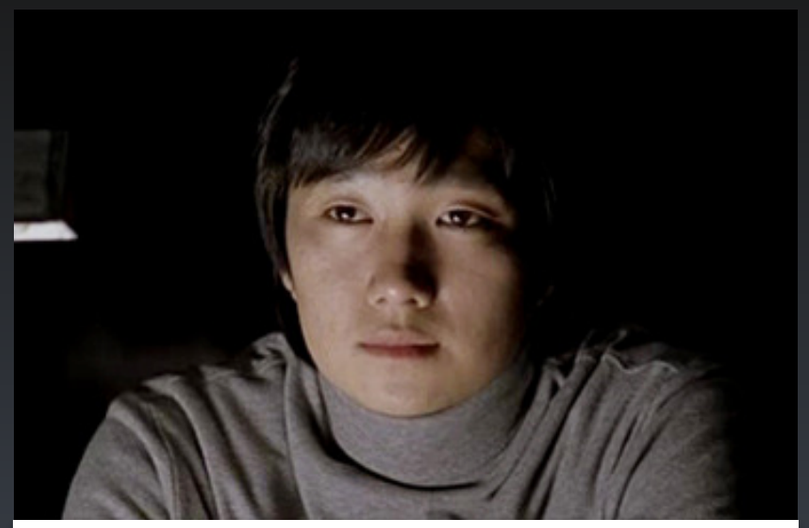
Director Clarence H. Kelly

Not for use in

출처: 동관K의 블로그 : 영화 “살인의 추억” DNA 검사 결과 번역
http://tcap.egloos.com/m/5809979


```
astr_evidence = {
    'CSF1PO': (12, 13),
    'D5S818': (13, 13),
    'D7S820': (8, 12),
    'D13S317': (8, 12),
    'TH01': (7, 7),
    'vWA': (18, 18),
    'D3S1358': (17, 18),
    'D8S1179': (15, 11),
    'D16S3252': (9, 9),
    'D18S51': (13, 16),
    'D21S11': (30, 32.2),
    'D21S11': (30, 32.2),
    'FGA': (21, 23),
    'PentaD': (11, 12),
    'PentaE': (10, 12),
}
```

≠



```
astr_suspect = {
    'CSF1PO': (11, 15),
    'D5S818': (8, 15),
    'D7S820': (11, 13),
    'D13S317': (12, 16),
    'TH01': (5, 7),
    'vWA': (18, 19),
    'D3S1358': (15, 19),
    'D8S1179': (12, 14),
    'D16S3252': (5, 11),
    'D18S51': (12, 14),
    'D21S11': (31, 35),
    'D21S11': (35, 36),
    'FGA': (15, 26),
    'PentaD': (13, 14),
    'PentaE': (8, 12),
}
```

DNA 법의학
유전자 감식(감정)

DNA Profiling
DNA fingerprinting
DNA testing
DNA typing

Alec Jeffreys

영국의 유전학자 (1950)

1985년 유전자지문 개발

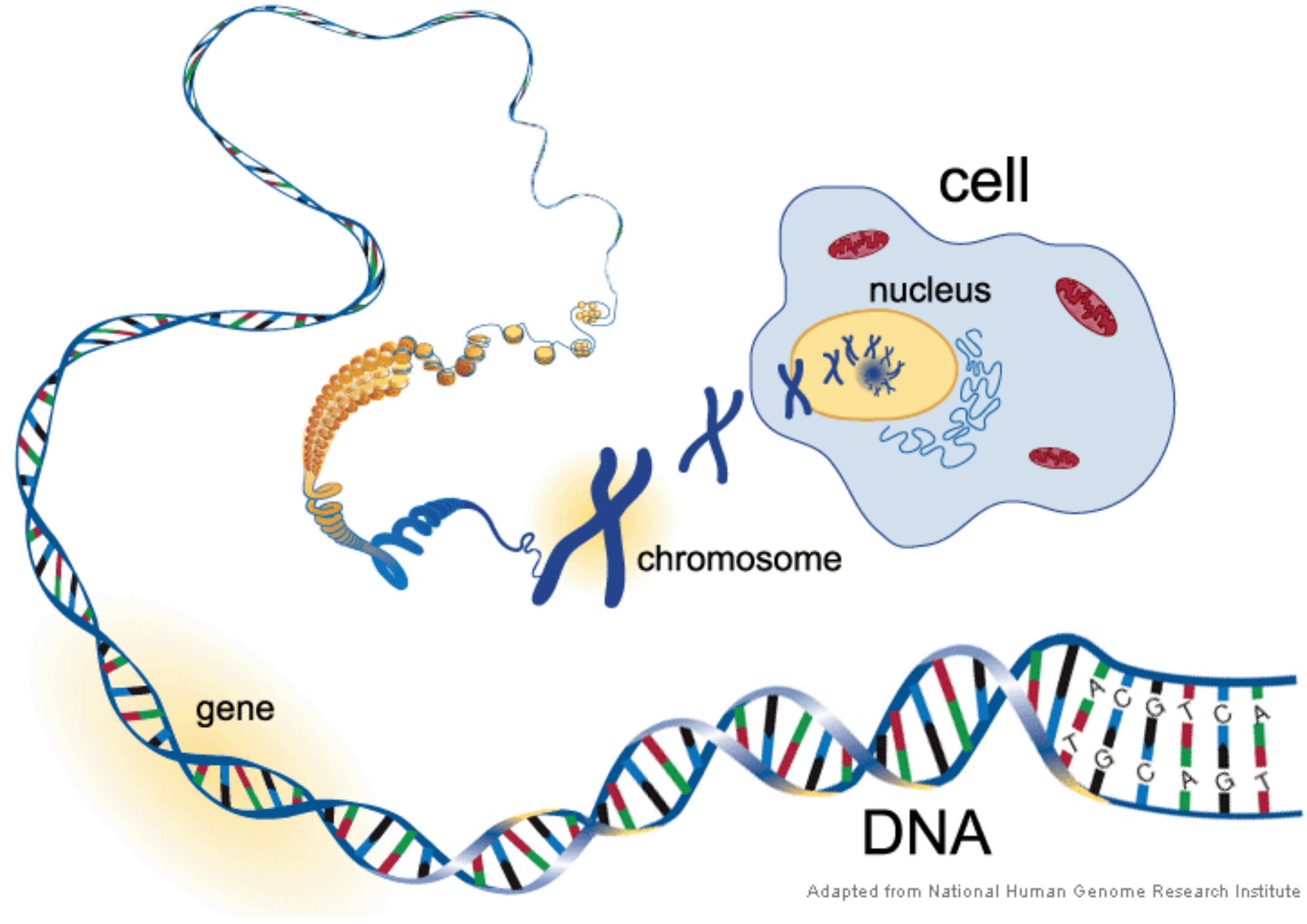
Individual specific “fingerprints” of human DNA

http://en.wikipedia.org/wiki/Alec_Jeffreys

Genetic variation (유전변이)

99 %

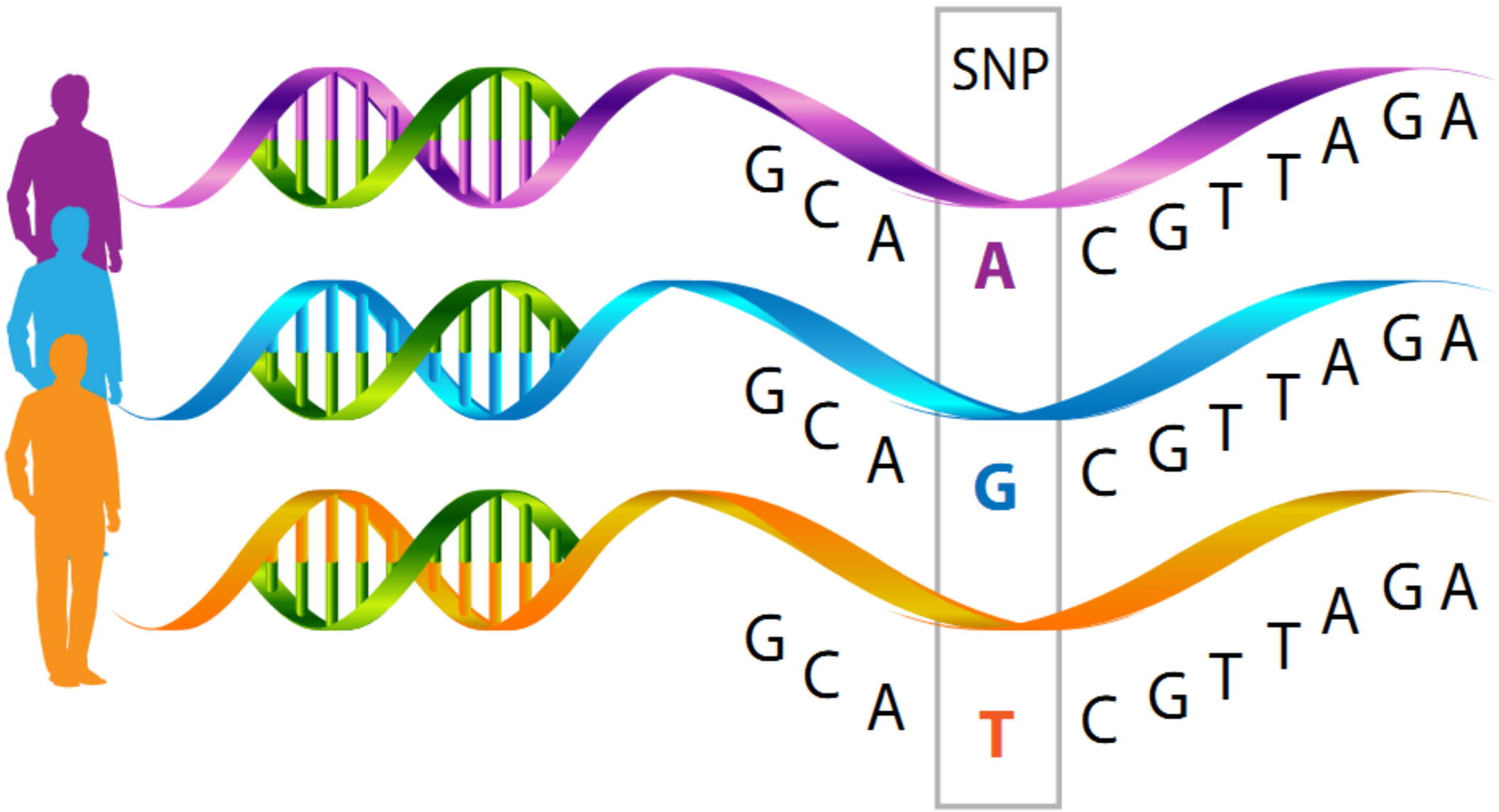
99.9 %



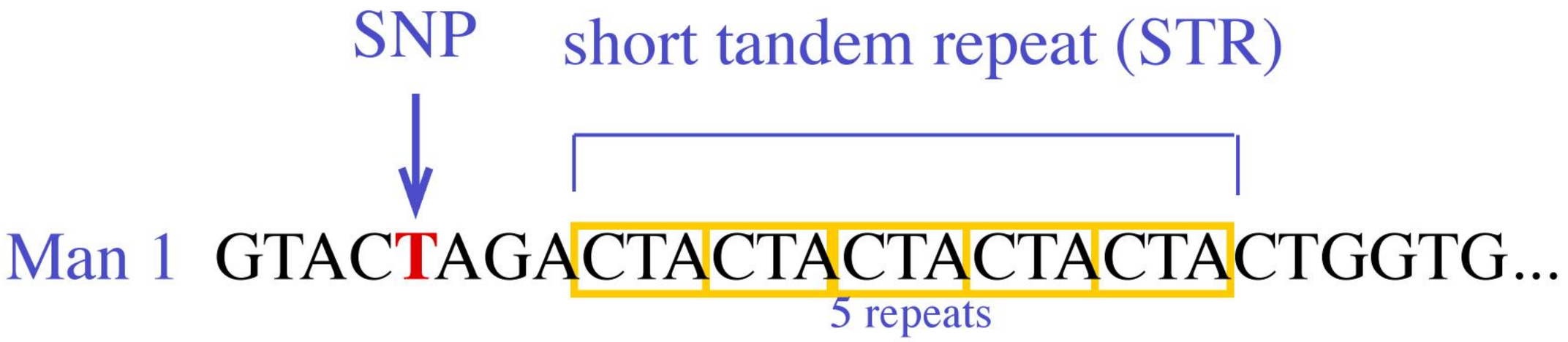
…AGCAAGTCAGATGACACCGTGACACGGATCAGT…

3Gbp = 3Gbyte

단일 염기 변이
(Single Nucleotide Polymorphism)

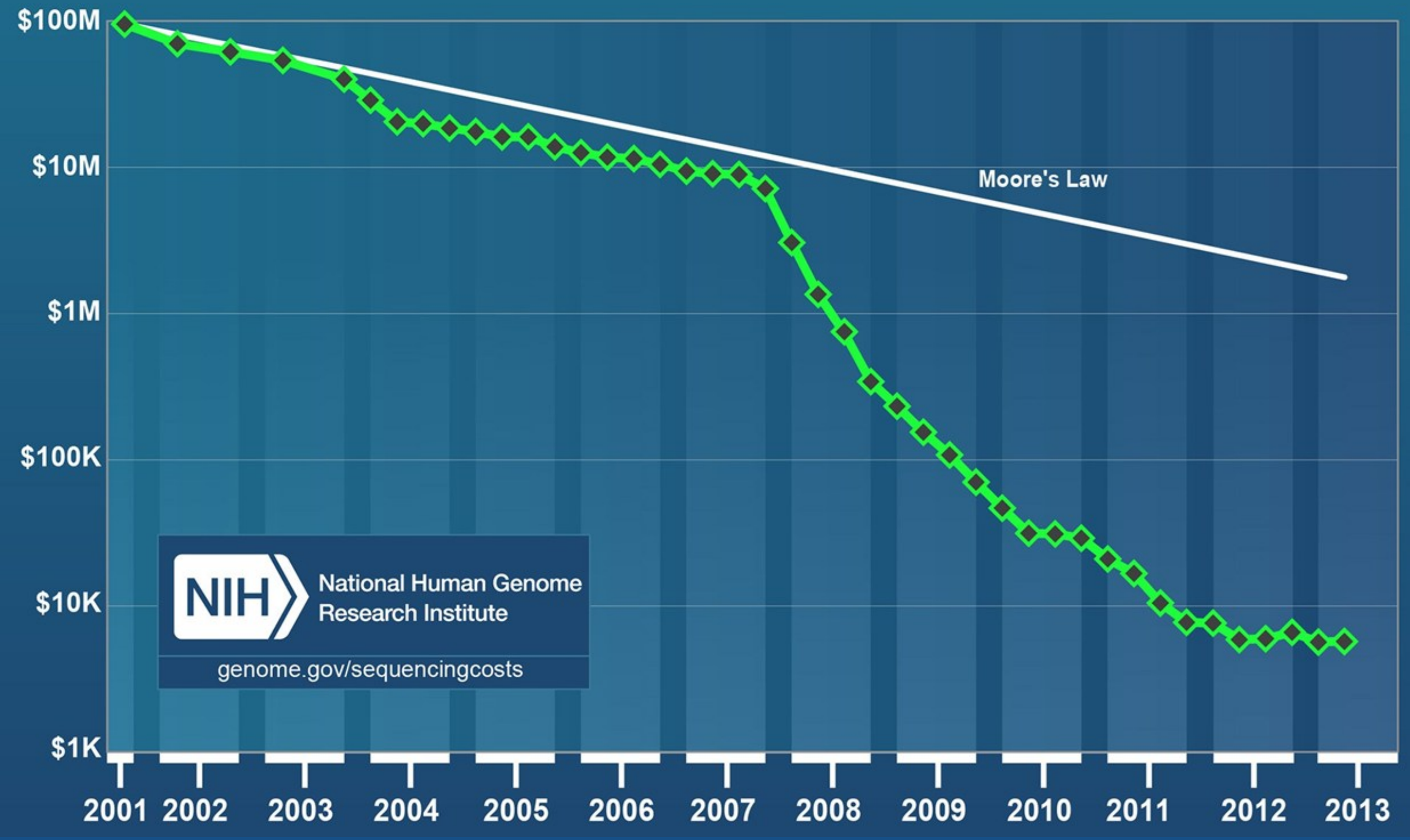


짧은 반복수 변이


SNP
short tandem repeat (STR)
Man 1 GTACTAGACTACTACTACTACTACTGGTG...
5 repeats
Man 2 GTACAAGACTACTACTACTACTACTACTGGTG...
6 repeats
Man 3 GTACAAGACTACTACTACTACTACTACTACTGGTG...
7 repeats

Now is 1000$ Genome era

Cost per Genome

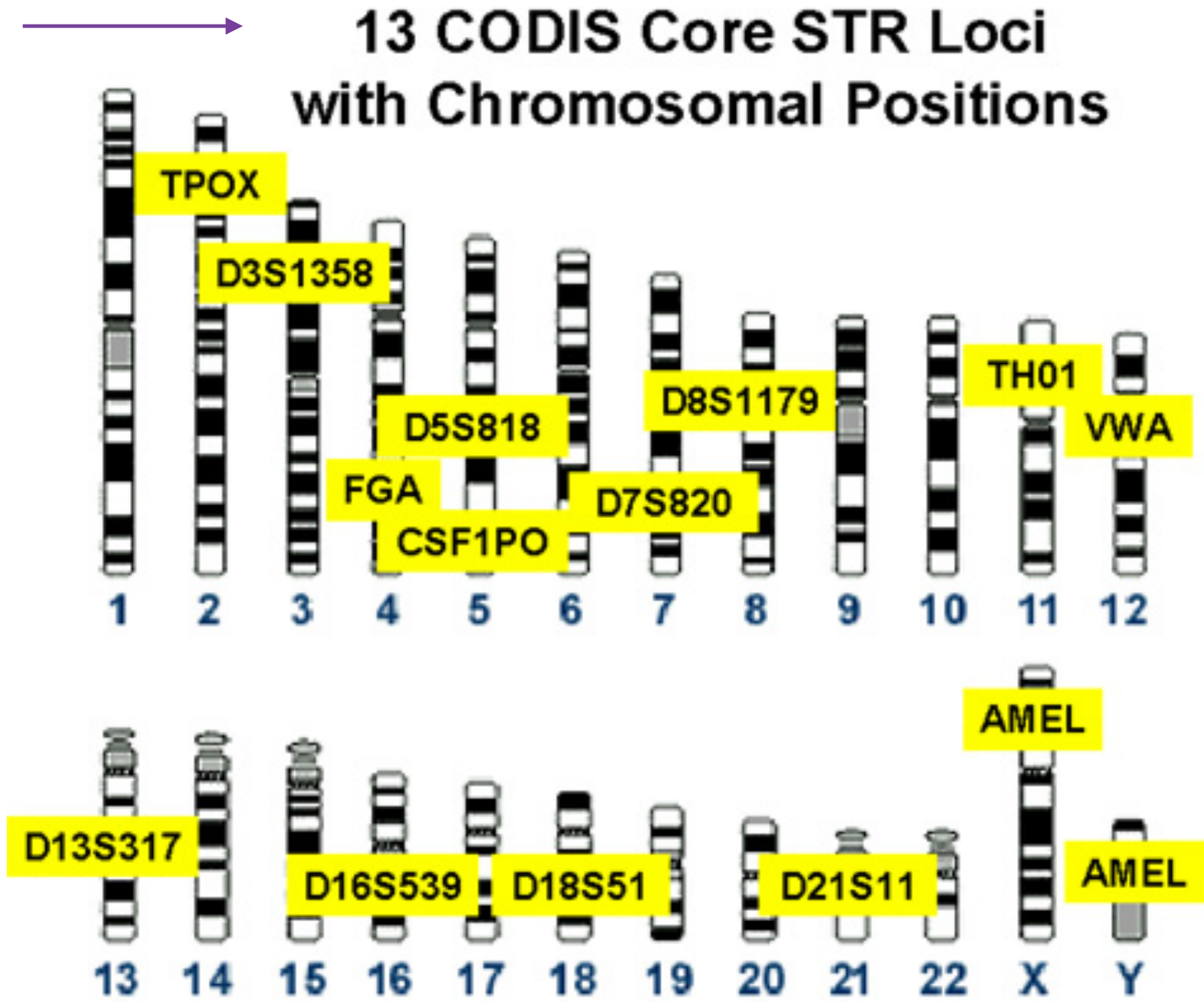


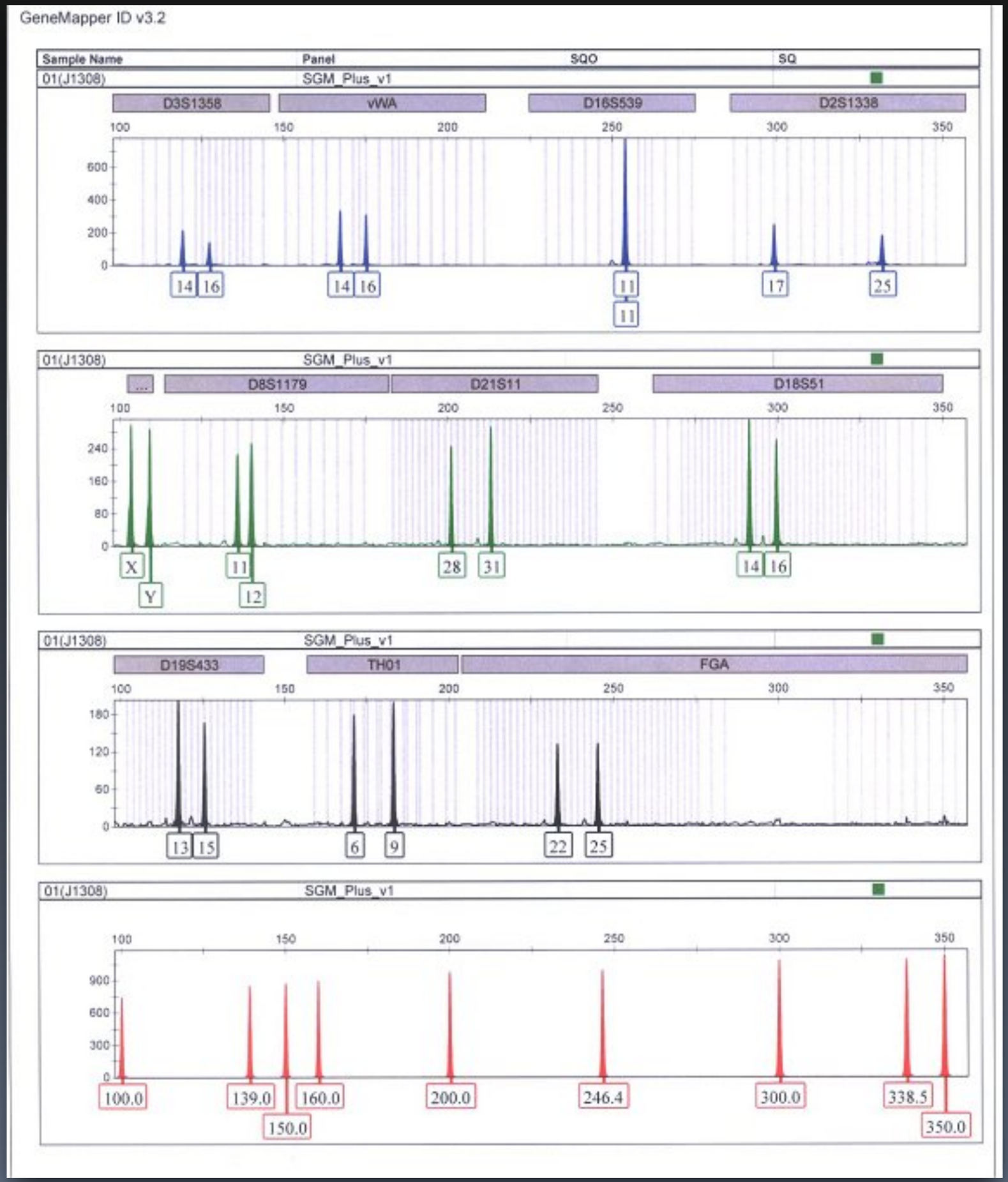
약 10여 좌위 STR

“검사 가지수를 높여서 우연히 맞을 확률을 낮춤”


DEPARTMENT of INVESTIGATION
FBI
THIS CERTIFIES THAT THE SIGNATURE AND PHOTOGRAPH
SPECIAL AGENT
OF FEDERAL BUREAU OF JUSTICE, UNITED STATES DEPARTMENT OF INVESTIGATION

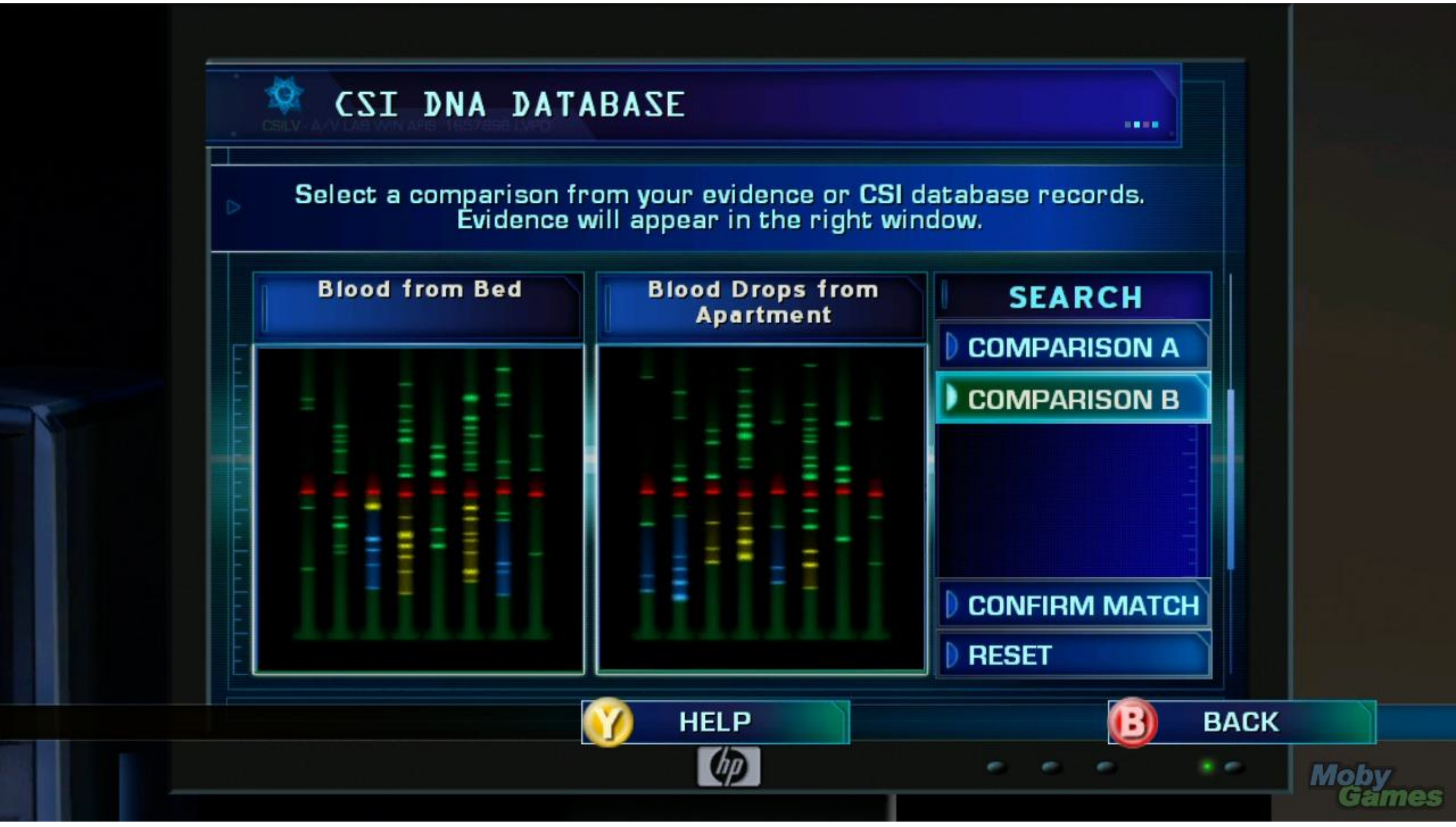
CODIS


13 CODIS Core STR Loci
with Chromosomal Positions
TPOX
D3S1358
D5S818
FGA
CSF1PO
D7S820
D8S1179
TH01
VWA
AMEL
D13S317
D16S539
D18S51
D21S11
AMEL
1 2 3 4 5 6 7 8 9 10 11 12
13 14 15 16 17 18 19 20 21 22 X Y



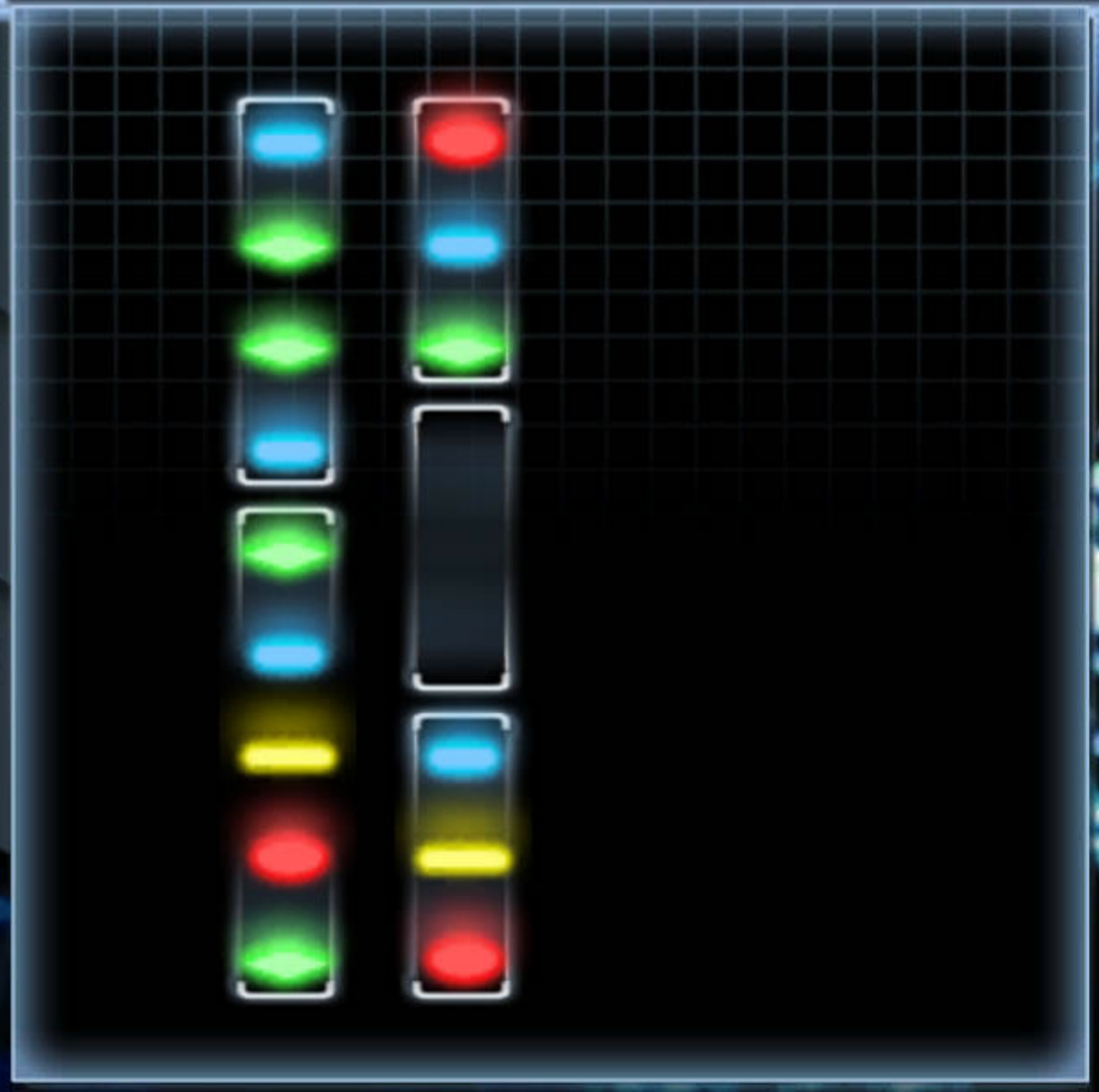
STR multiplex

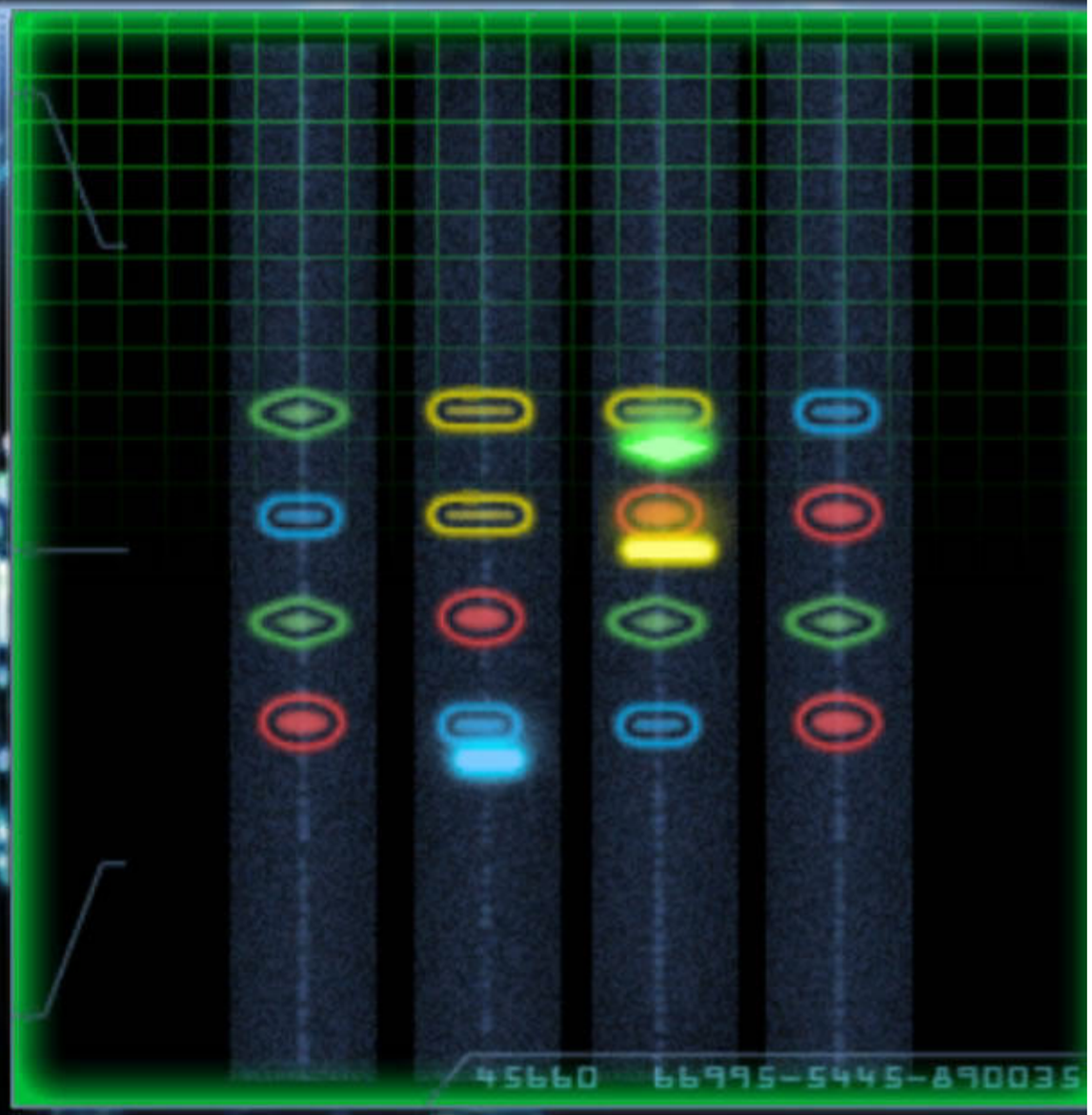
좌위	유전자형
D3S1358	14,16
vWA	14,16
D16S539	11,11
D2S1338	17,25
D8S1179	11,12
D21S11	28,31
D18S51	14,16
D19S433	13,15
TH01	6,9
FGA	22,25


CSI DNA DATABASE
Select a comparison from your evidence or CSI database records.
Evidence will appear in the right window.
Blood from Bed
Blood Drops from Apartment
SEARCH
COMPARISON A
COMPARISON B
CONFIRM MATCH
RESET
Y
HELP
B
BACK
Moby Games

Blood from carpet at crime scene

Victim's blood from crime scene





HELP

BACK

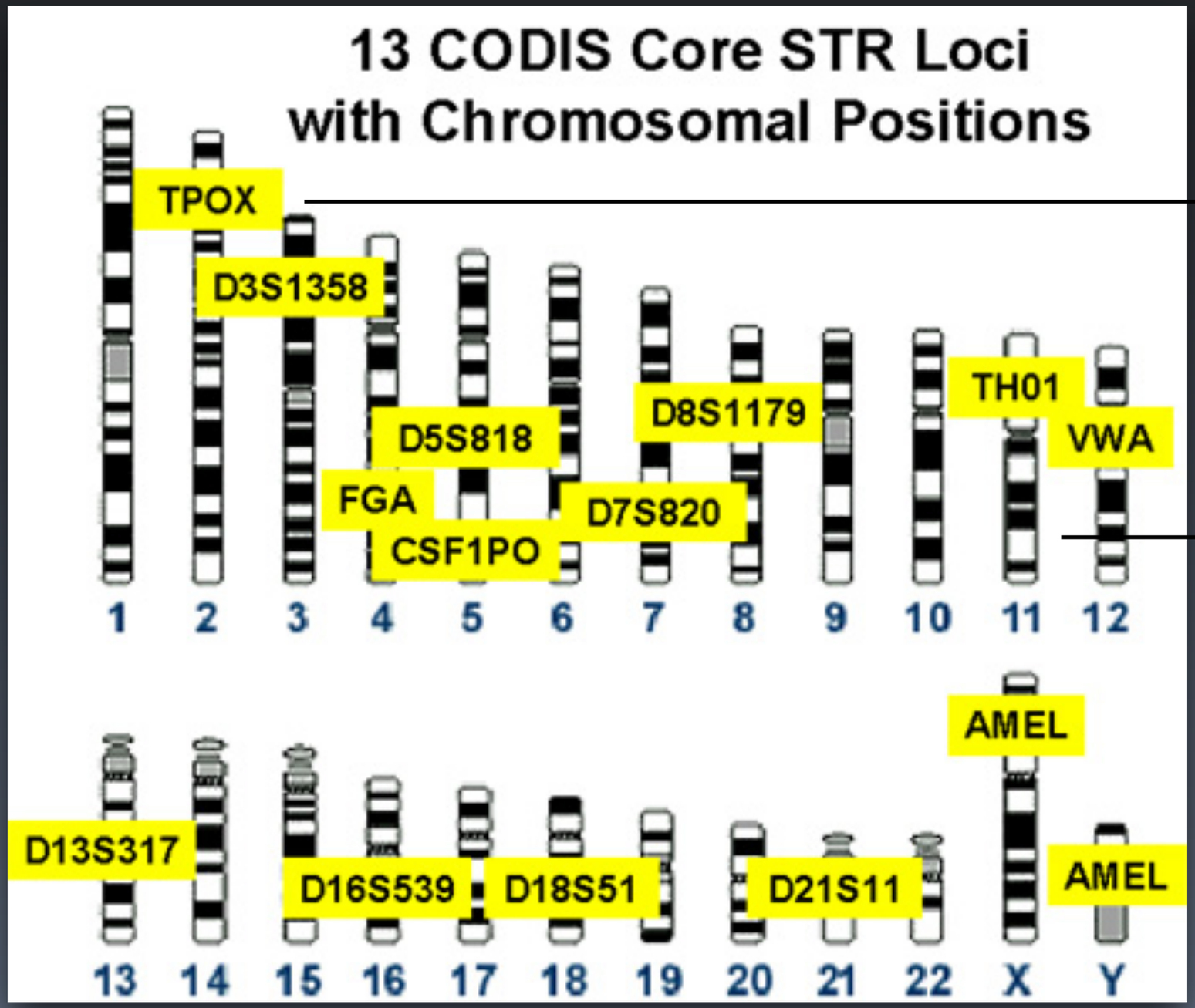
사실은

```
astr_evidence = {
    'CSF1PO': (12, 13),
    'D5S818': (13, 13),
    'D7S820': (12, 8),
    'D13S317': (12, 8),
    'TH01': (7, 7),
    'vWA': (18, 18),
    'D3S1358': (17, 18),
    'D8S1179': (15, 11),
    'D16S3252': (9, 9),
    'D18S51': (13, 16),
    'D21S11': (30, 32.2),
    'D21S11': (30, 32.2),
    'FGA': (21, 23),
    'PentaD': (11, 12),
    'PentaE': (10, 12),
}
```

```
astr_suspect = {
    'CSF1PO': (12, 13),
    'D5S818': (13, 13),
    'D7S820': (12, 8),
    'D13S317': (12, 8),
    'TH01': (7, 7),
    'vWA': (18, 18),
    'D3S1358': (17, 18),
    'D8S1179': (15, 11),
    'D16S3252': (9, 9),
    'D18S51': (13, 16),
    'D21S11': (30, 32.2),
    'D21S11': (30, 32.2),
    'FGA': (21, 23),
    'PentaD': (11, 12),
    'PentaE': (10, 12),
}
```

```
all(astr_suspect[locus] == genotype for locus, genotype in
        astr_evidence.items())
```

얼마나 유의하게 같은가?




```
TPOX = {
    7:     0.002,
    8:     0.517,
    9:     0.114,
    10:     0.024,
    11:     0.307,
    12:     0.032,
    13:     0.004,
    }
```

```
TH01 = {
    6: 0.157,
    7: 0.235,
    8: 0.05,
    9: 0.494,
    9.3: 0.045,
    10: 0.017,
    11: 0.002,
    }
```

한국인의
대립유전자빈도

evidence | suspect

STR marker	allele 1(p)	allele 2(q)	allele 1(p)	allele 2(q)	발생빈도 (frequency)
CSF1PO	13	12	13	12	0.056
D5S818	13	13	13	13	0.015
D7S820	12	8	12	8	0.052
D13S317	12	8	12	8	0.089
TH01	7	7	7	7	0.061
TPOX	12	8	12	8	0.025
vWA	18	18	18	18	0.033
D3S1358	18	17	18	17	0.033
D8S1179	15	11	15	11	0.024
D16S3253	9	9	9	9	0.004
D18S51	16	13	16	13	0.032
D21S11	32.2	30	32.2	30	0.057
FGA	23	21	23	21	0.063
PentaD	12	11	12	11	0.050
PentaE	12	10	12	10	0.006
D12S391	20	17	20	17	0.043
D14S608	12	11	12	11	0.066

← Korean population Allele frequencies

All alleles should be matched.

2.19×10^{-15} → Combined frequency P(x)

Likelihood Ratio (우도비)

= Combined frequency of all matched alleles

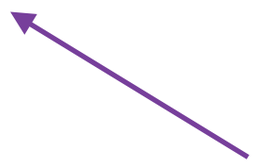
= 1 / P(x)

= **Identity Index**

"두 검체의 DNA typing이 동일할 경우, 증거물이 혐의자로 부터 유래했을 가능성과 그렇지 않은 가능성과의 비율"

```
korean_allele_frequencies = {
    'TH01': {
        6: 0.157,
        7: 0.235,
        8: 0.05,
        9: 0.494,
        9.3: 0.045,
        10: 0.017,
        11: 0.002,
    },
    'TPOX': {
        7: 0.002,
        8: 0.517,
        9: 0.114,
        10: 0.024,
        11: 0.307,
        12: 0.032,
        13: 0.004,
    },
    ...
}
```

```python
combined_frequency = 1
for locus, (a1, a2) in astr_evidence.items():
    pa1 = korean_allele_frequencies[locus][a1]
    pa2 = korean_allele_frequencies[locus][a2]
    if a1 == a2:
        combined_frequency *= pa1 * pa2
    else:
        combined_frequency *= 2 * pa1 * pa2

print(1 / combined_frequency)
```



얼마나 동일하냐 우도비(LR)

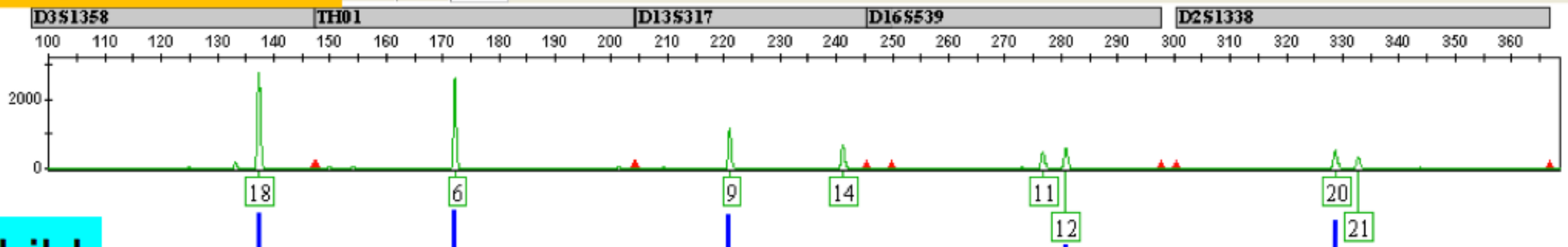
하지만

죽은 백광호의 아들이 있다!

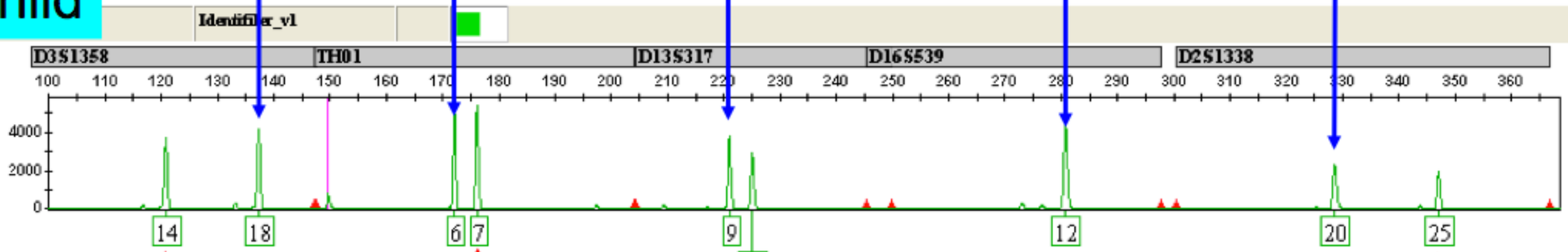
(물론 픽션입니다)

아들 DNA 로 백광호의 범죄 여부?

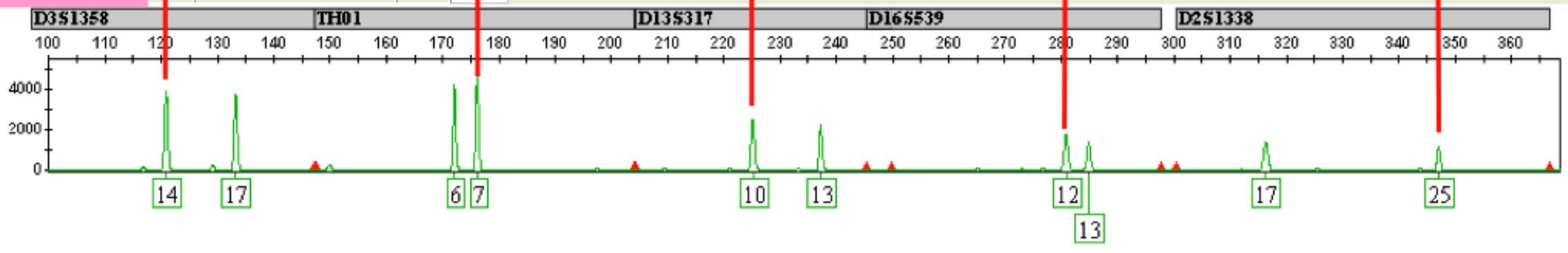
Alleged Father



Child



Mother



	Marker	Father		Mother	
1	CSF1PO	10	10	10	12
2	TPOX	8	8	8	10
3	TH01	6	6	6	9
4	vWA	17	18	17	20
5	D16S539	11	13	8	9
6	D7S820	9	9	8	12
7	D13S317	11	14	8	12
8	D5S818	12	13	11	13
9	FGA	21	22	21	25
10	D8S1179	12	14	13	14
11	D18S51	14	16	14	17
12	D21S11	28	30	31	32.2
13	D3S1358	16	17	17	17
14	D2S1338	22	23	23	25
15	D19S433	12	14	14	14

	Marker	Father		Child		Mother	
1	CSF1PO	10	10	10	10	10	12
2	TPOX	8	8	8	8	8	10
3	TH01	6	6	6	6	6	9
4	vWA	17	18	17	17	17	20
5	D16S539	11	13	9	13	8	9
6	D7S820	9	9	8	9	8	12
7	D13S317	11	14	8	14	8	12
8	D5S818	12	13	11	13	11	13
9	FGA	21	22	21	25	21	25
10	D8S1179	12	14	14	14	13	14
11	D18S51	14	16	14	17	14	17
12	D21S11	28	30	28	31	31	32.2
13	D3S1358	16	17	17	17	17	17
14	D2S1338	22	23	23	23	23	25
15	D19S433	12	14	12	14	14	14

```
astr_child = {
    'CSF1PO': (12, 13),
    'D5S818': (13, 13),
    'D7S820': (12, 8),
    'D13S317': (12, 8),
    'TH01': (7, 7),
    'vWA': (18, 18),
    'D3S1358': (17, 18),
    'D8S1179': (15, 11),
    'D16S539': (9, 9),
    'D18S51': (13, 16),
    'D21S11': (30, 32.2),
    'FGA': (21, 23),
    'PentaD': (11, 12),
    'PentaE': (10, 12),
  }
```

```
astr_father = {
    'CSF1PO': (11, 13),
    'D5S818': (13, 13),
    'D7S820': (12, 9),
    'D13S317': (12, 10),
    'TH01': (7, 7),
    'vWA': (15, 18),
    'D3S1358': (13, 18),
    'D8S1179': (11, 11),
    'D16S539': (9, 12),
    'D18S51': (13, 19),
    'D21S11': (30, 32.2),
    'FGA': (21, 21),
    'PentaD': (11, 11),
    'PentaE': (10, 15),
}
```

```
all(set(astr_child[locus]) & set(genotype) for locus, genotype
        in astr_father.items())
```

얼마나 유의하게 친자관계인가?

Genotype Combinations at One Marker		Likelihood Ratio
Child's Genotype	Alleged Father's Genotype	
AA	AA	$1/p_A$
AA	BB	0
AA	AB	$1/2p_A$
AA	BC	0
AB	AB	$(p_A + p_B)/4p_Ap_B$
AB	AC	$1/4p_A$
AB	CD	0

Multiply LR across all loci

$$LR = \frac{\text{Probability of data if the alleged father is the true father}}{\text{Probability of data if an unrelated man is the true father}}$$

"추정부와 자식이 실제 부모자식 관계에 있을 가능성과 그렇지 않은 가능성과의 비율"

```
formula = {
    1: (lambda pa, pb: 0.25 / pa),
    2: (lambda pa, pb: (pa + pb) / (4 * pa * pb)),
    3: (lambda pa, pb: 0.5 / pa),
    4: (lambda pa, pb: 0.5 / pa),
    5: (lambda pa, pb: 1 / pa),
}

def find_formula(a1, a2, b1, b2):
    if a1 != b1:
        if a1 == b2:
            b1, b2 = b2, b1
        if a2 == b1:
            a1, a2 = a2, a1
        if a2 == b2:
            a1, a2 = a2, a1
            b1, b2 = b2, b1
    if a1 == b1 and a2 != b2 and a1 != a2 and b1 != b2:
        f = 1; A = a1; B = a2
    elif a1 == b1 and a2 == b2 and a1 != a2:
        f = 2; A = a1; B = a2
    elif a1 == b1 and a2 != b2 and b1 == b2:
        f = 3; A = a1; B = a2
    elif a1 == b1 and a2 != b2 and a1 == a2:
        f = 4; A = a1; B = a2
    elif a1 == b1 and a2 == b2 and a1 == a2:
        f = 5; A = a1; B = a2
    return f, A, B
```

유전자형 조합에 따른 공식찾기

```
index = 1
for locus, (a1, a2) in astr_child.items():
    b1, b2 = astr_father[locus]
    f, A, B = find_formula(a1, a2, b1, b2)
    pa = korean_allele_frequencies[locus][A]
    pb = korean_allele_frequencies[locus][B]
    index *= formula[f](pa, pb)

print(index)
```

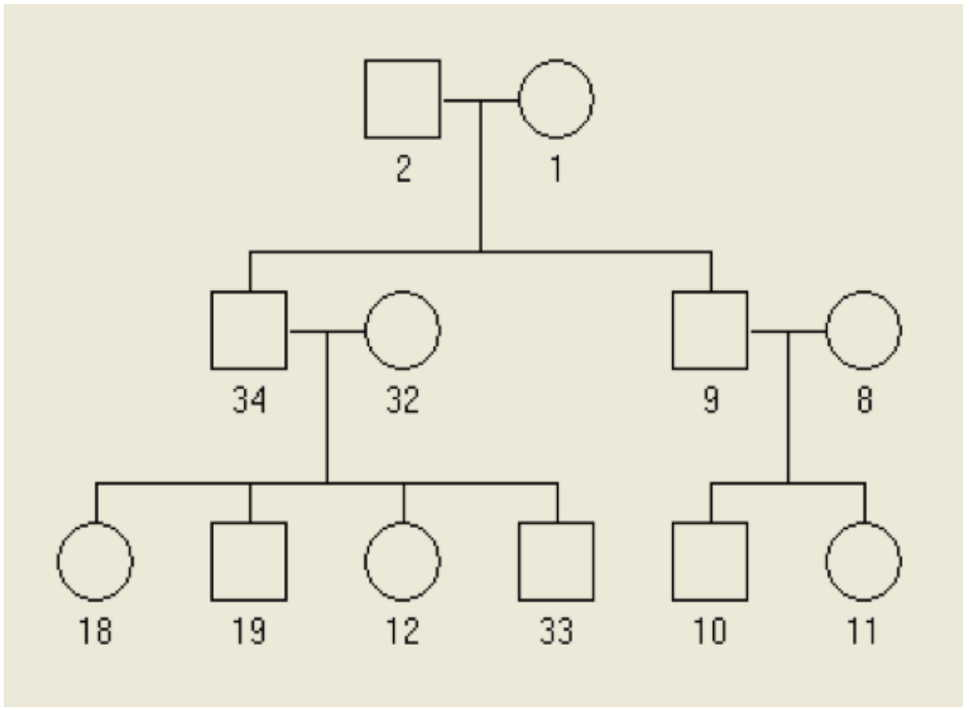
백광호는 역시 범인이 아님

박현규의 삼촌이
생존함!

삼촌 DNA로 박현규 범죄 여부?

	Marker	Brother		Sister	
1	CSF1PO	10	10	10	12
2	TPOX	8	8	8	10
3	TH01	6	6	6	9
4	vWA	17	17	17	20
5	D16S539	9	13	9	11
6	D7S820	8	9	9	12
7	D13S317	8	14	11	12
8	D5S818	11	13	11	12
9	FGA	21	25	21	25
10	D8S1179	14	14	13	14
11	D18S51	14	17	14	14
12	D21S11	28	31	30	32
13	D3S1358	17	17	16	17
14	D2S1338	23	23	23	25
15	D19S433	12	14	14	14

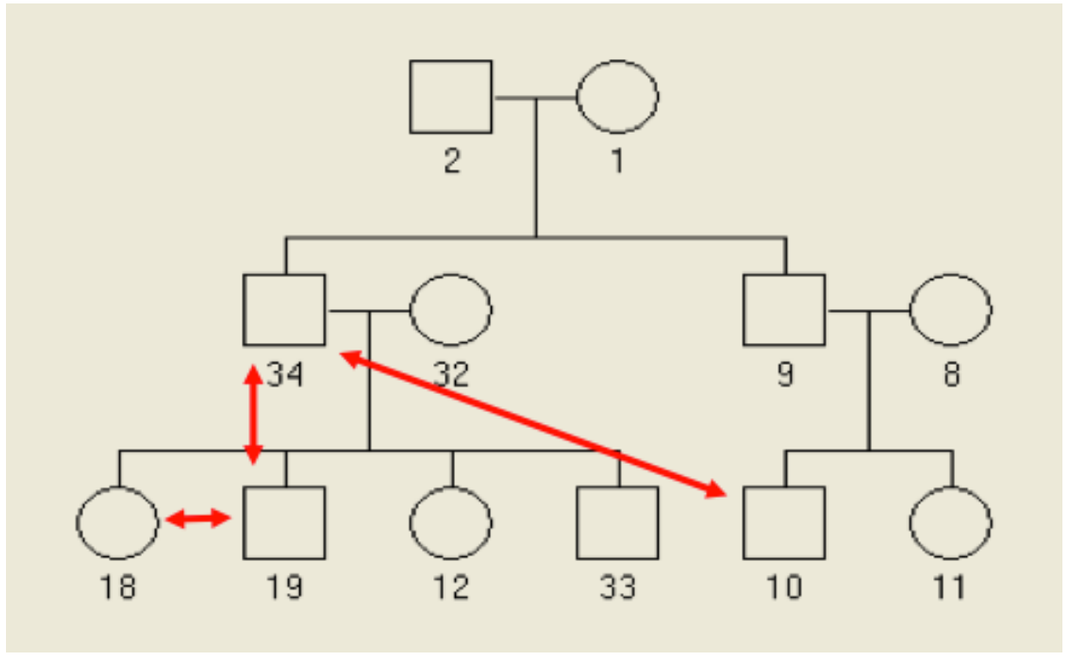
	Marker	Uncle		Nephew	
1	CSF1PO	10	10	11	12
2	TPOX	8	8	8	10
3	TH01	6	6	6	9.3
4	vWA	17	18	16	17
5	D16S539	11	13	11	13
6	D7S820	9	9	9	11
7	D13S317	11	14	11	12
8	D5S818	12	13	12	12
9	FGA	21	22	20	24
10	D8S1179	12	14	10	13
11	D18S51	14	16	13	13
12	D21S11	28	30	27	31
13	D3S1358	16	17	16	18
14	D2S1338	22	23	18	22
15	D19S433	12	14	14	14



가족관계별
대립유전자(allele) 공유 확률

Relationship	0 alleles	1 alleles	2 alleles
Parent-child	0	1	0
Full siblings	1/4	1/2	1/4
Half siblings	1/2	1/2	0
Cousins	3/4	1/4	0
Uncle-nephew	1/2	1/2	0
Grandparent-grandchild	1/2	1/2	0

Half siblings, uncle-nephew,
and grandparent-grandchild are genetically identical.



가족관계별 LR 계산 (15좌위, 40좌위)

	15	40	15	40	15	40
Comparison	**LR for 34 & 19**	**LR for 34 & 19**	**LR for 18 & 19**	**LR for 18 & 19**	**LR for 34 & 10**	**LR for 34 & 10**
Parent-Child	1.28E+06	6.68E+16	9.08E+05	0.00E+00	0.00E+00	0.00E+00
Full Siblings	3.22E+04	5.73E+12	2.76E+07	1.57E+19	6.07E-03	3.30E+03
Half Siblings	7.38E+03	8.63E+11	4.89E+04	4.99E+12	6.65E-01	8.98E+05
Cousins	1.95E+02	1.32E+08	8.96E+02	1.05E+09	1.52E+00	2.17E+04
Uncle-Nephew	7.38E+03	8.63E+11	4.89E+04	4.99E+12	**6.65E-01**	**8.98E+05**
Grandparent-Grandchild	7.38E+03	8.63E+11	4.89E+04	4.99E+12	6.65E-01	8.98E+05
	Parent/ Child		Full Sibs		Uncle/ Nephew	

즉, 가족관계별로 LR 계산

가장 큰 LR 찾기

Kinship Formulas:

$$[P_2(xy) \times \phi_2] + [P_1(xy) \times \phi_1] + [P_0(xy) \times \phi_0]$$

#1	#2	frequency
AB	AB	$\phi_2 + 0.5\phi_1(p_A + p_B) + 2\phi_0 p_A p_B$
AA	AA	$\phi_2 + \phi_1 p_A + \phi_0 p_A^2$
AA	AB	$\phi_1 p_B + 2\phi_0 p_A p_B$
AB	AC	$0.5\phi_1 p_C + 2\phi_0 p_A p_C$
AB	CD	$2\phi_0 p_C p_D$
AA	BB	$\phi_0 p_B^2$
AA	BC	$2\phi_0 p_B p_C$

IBD coefficient	Φ0	Φ1	Φ2
Parent-Child	*0*	*1*	*0*
Full-sib	*1/4*	*1/2*	*1/4*
Half-sib	*1/2*	*1/2*	*0*
4 chon	*3/4*	*1/4*	*0*
Unrelated	*1*	*0*	*0*
5 chon	*7/8*	*1/8*	*0*
6 chon	*15/16*	*1/16*	*0*
7 chon	*31/32*	*1/32*	*0*
8 chon	*63/64*	*1/64*	*0*

$$KI = \frac{\Pr(X, Y \mid Relationship)}{\Pr(X, Y \mid Unrelated)}$$

Kinship Index
= P(X, Y|related) / P(X, Y|unrelated)

"두 검체의 DNA typing (X, Y)이 특정 친척관계에 있을 가능성과 그렇지 않을 가능성과의 비율"

한국일보

경찰 "유병언 의심 사체 발견…형 DNA와 상당부분 일치"

여기서 잠깐!

형제관계 LR가 높으면
(10^5) 인정 가능

등록: 2014.07.22 00:49

유병언 전 세모그룹 회장에 대한 구속영장이 재발부된 21일 오후 서울 종로구청 민원실 입구에 유 전 회장과 아들 대균 씨의 수배전단이 붙어 있다. 연합뉴스

검찰 수사를 피해 달아난 '세월호 실소유주' 유병언(73) 전 세모그룹 회장(청해진해운 회장)으로 의심되는 사체가 발견됐다.

범죄현장 시료와 삼촌과
혈연관계 없음!

세월이 흘러 2014년

디엔에이신원확인정보의 이용 및 보호에 관한 법률

[시행 2014.1.7.] [법률 제12186호, 2014.1.7., 일부개정] 최종공포내용

2014.1.7 시행

대검찰청(디엔에이수사담당관), 02-3480-2465
법무부(형사법제과), 02-2110-3307~8
경찰청(과학수사센터), 02-3150-1750

판 **제1조(목적)** 이 법은 디엔에이신원확인정보의 수집·이용 및 보호에 필요한 사항을 정함으로써 범죄수사 및 범죄 예방에 이바지하고 국민의 권익을 보호함을 목적으로 한다.

데이터베이스

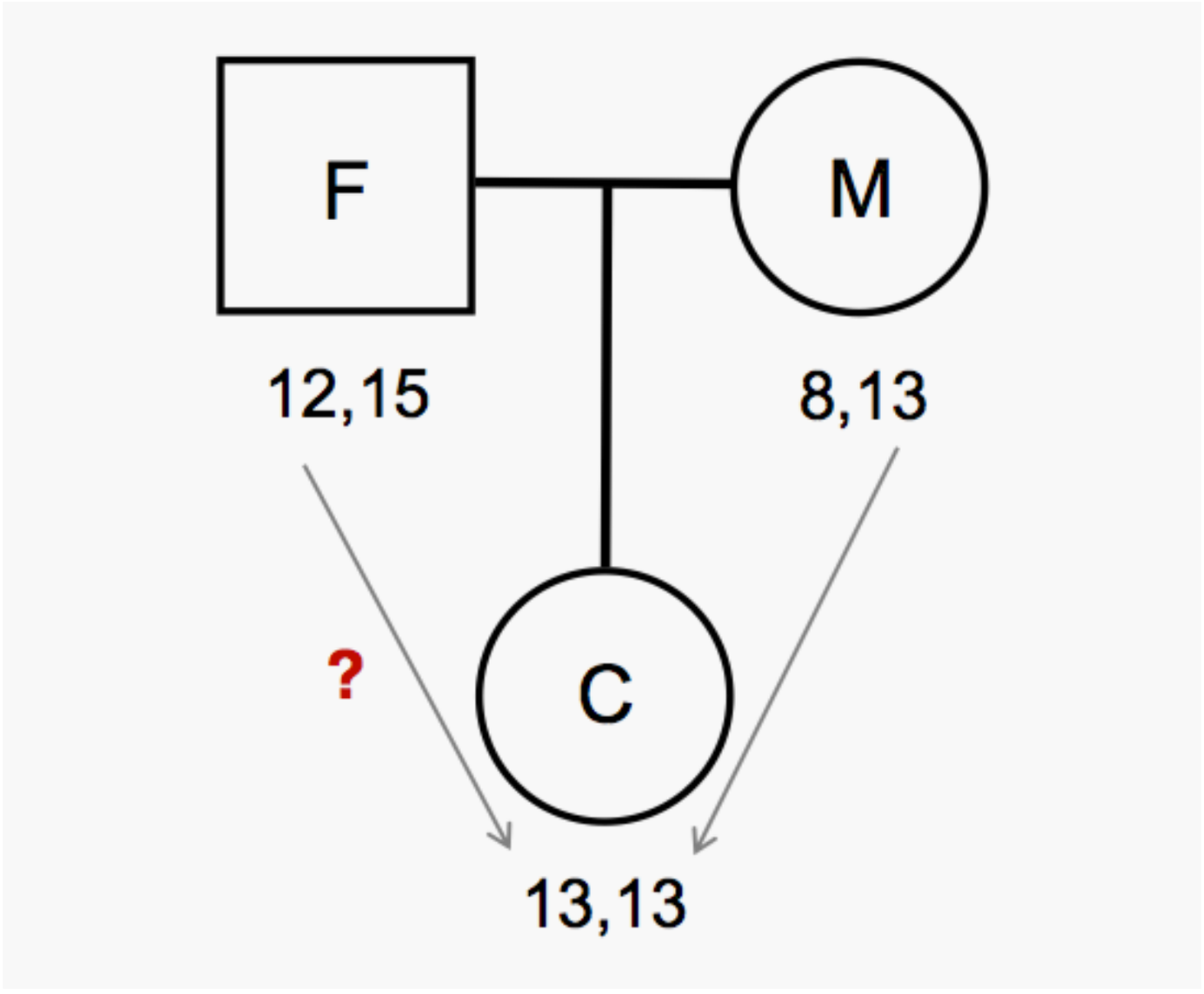
판 **제2조(정의)** 이 법에서 사용하는 용어의 뜻은 다음과 같다.

1. "디엔에이"란 생물의 생명현상에 대한 정보가 포함된 화학물질인 디옥시리보 핵산(Deoxyribonucleic acid, DNA)을 말한다.
2. "디엔에이감식시료"란 사람의 혈액, 타액, 모발, 구강점막 등 디엔에이감식의 대상이 되는 것을 말한다.
3. "디엔에이감식"이란 개인 식별을 목적으로 디엔에이 중 유전정보가 포함되어 있지 아니한 특정 염기서열 부분을 검사·분석하여 디엔에이신원확인정보를 취득하는 것을 말한다.
4. "디엔에이신원확인정보"란 개인 식별을 목적으로 디엔에이감식을 통하여 취득한 정보로서 일련의 숫자 또는 부호의 조합으로 표기된 것을 말한다.
5. "디엔에이신원확인정보데이터베이스"(이하 "데이터베이스"라 한다)란 이 법에 따라 취득한 디엔에이신원확인정보를 컴퓨터 등 저장매체에 체계적으로 수록한 집합체로서 개별적으로 그 정보에 접근하거나 검색할 수 있도록 한 것을 말한다.

범죄자 데이터베이스에서 검색?

비슷한 것도 찾아줘야 한다

– 근사 검색 (approximate search)



돌연변이
혹은
실험 오차?

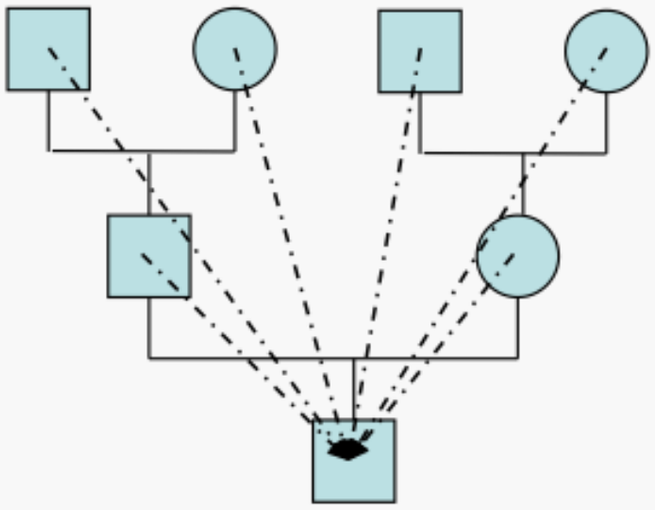
부분 검색

```python
astr_query = {
    'CSF1PO': (12, 13),
    'D5S818': (13, 13),
    'D7S820': (12, 8),
    'D13S317': (12, 8),
    'TH01': (7, 7),
    'vWA': (18, 18),
    'D3S1358': (17, 18),
    'D8S1179': (15, 11),
    'D16S539': (9, 9),
    'D18S51': (13, 16),
    'D21S11': (30, 32.2),
    'FGA': (21, 23),

}
```

```python
astr_target = {
    'CSF1PO': (11, 13),
    'D5S818': (13, 13),
    'D7S820': (12, 9),
    'D13S317': (12, 10),

    'D3S1358': (13, 18),
    'D8S1179': (11, 11),
    'D16S539': (9, 12),
    'D18S51': (13, 19),
    'D21S11': (30, 32.2),
    'FGA': (21, 21),
    'PentaD': (11, 11),
    'PentaE': (10, 15),
}
```

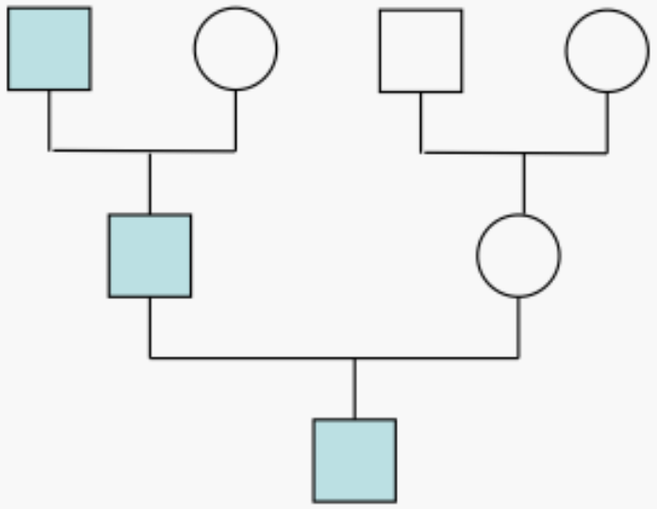
조합 검색

CODIS STR Loci

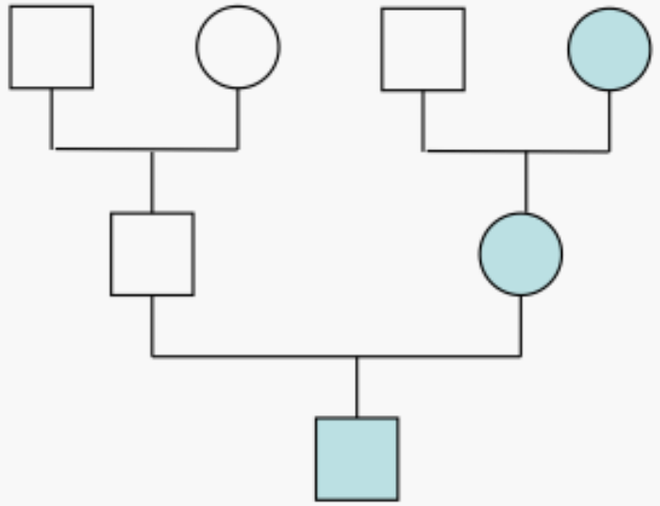


Autosomal
(passed on in part,
from all ancestors)

Lineage Markers



Y-Chromosome
(passed on complete,
but only by sons)



Mitochondrial
(passed on complete,
but only by daughters)

```python
database = [
    'sample1': {
        'CSF1PO': (11, 13),
        'D5S818': (13, 13),
        'D7S820': (8, 11),
        'D13S317': (7, 8),
        'TH01': (7, 7),
        'vWA': (18, 18),
        ...
    },
    'sample2': {
        'CSF1PO': (11, 16),
        'D5S818': (10, 18),
        'D7S820': (8, 11),
        'D13S317': (7, 9),
        'TH01': (7, 7),
        'vWA': (18, 18),
        ...
    },
    ...
]
```

```python
astr_query = {
    'CSF1PO': (12, 13),
    'D5S818': (13, 13),
    'D7S820': (8, 12),
    'D13S317': (8, 12),
    'TH01': (7, 7),
    'vWA': (18, 18),
    'D3S1358': (17, 18),
    'D8S1179': (15, 11),
    'D16S539': (9, 9),
    'D18S51': (13, 16),
    'D21S11': (30, 32.2),
    'FGA': (21, 23),
    'PentaD': (11, 12),
    'PentaE': (10, 12),
}
result = []
for target in database:
    mismatches = 0
    for locus, genotype in astr.query.items():
        if genotype != target[locus]:
            mismatches += 1
    if mismatches < 2:
        result.append(target)

print(result)
```

불일치수 2개 이하만 →

← 검색결과

데이터베이스가 커질수록 느려짐

RDB?

	locus1_1	locus1_2	locus2_1	locus2_2	locus3_1	locus3_2	…
sample1	12	14	11	15	11	14	
sample2	12	13	11	15	11	11	
sample3	14	15	11	13	11	12	
sample4	15	13	7	8	11	9	
…							

A-STR 동일성 검색 SQL (불일치수 포함)

```
SELECT * FROM astr WHERE (
    (
        (CASE WHEN
            astr.locus1_1 = query_locus1_a1 AND
            astr.locus1_2 = query_locus1_a2
        THEN 1 ELSE 0)
        + (CASE WHEN
            astr.locus2_1 = query_locus1_a1 AND
            astr.locus2_2 = query_locus2_a2
          THEN 1 ELSE 0)
        + …
    ) >= count_locus - unmatch_count
)
```

Django “extra” queryset modifier

```
match_wheres = []
for locus, (a1, a2) in query.items():
    match_wheres.append('''
        CASE WHEN astr.locus1_a1 = {} and astr.locus_a2 = {}
        THEN 1 ELSE 0
        '''.format(a1, a2))

ASTR.objects.extra(
    where=[
        '{} >= {}'.format('+'.join(match_wheres), matching_count),
    ],
)
```

빠른 동일성 검색, 친자검색
(30,000 records, PostgreSQL)

혹시 친척검색도
할 수 있나?

Kinship Formulas:

$$[P_2(xy) \times \phi_2] + [P_1(xy) \times \phi_1] + [P_0(xy) \times \phi_0]$$

#1	#2	frequency
AB	AB	$\phi_2 + 0.5\phi_1(p_A + p_B) + 2\phi_0 p_A p_B$
AA	AA	$\phi_2 + \phi_1 p_A + \phi_0 p_A^2$
AA	AB	$\phi_1 p_B + 2\phi_0 p_A p_B$
AB	AC	$0.5\phi_1 p_C + 2\phi_0 p_A p_C$
AB	CD	$2\phi_0 p_C p_D$
AA	BB	$\phi_0 p_B^2$
AA	BC	$2\phi_0 p_B p_C$

IBD coefficient	Φ0	Φ1	Φ2
Parent-Child	*0*	*1*	*0*
Full-sib	*1/4*	*1/2*	*1/4*
Half-sib	*1/2*	*1/2*	*0*
4 chon	*3/4*	*1/4*	*0*
Unrelated	*1*	*0*	*0*
5 chon	*7/8*	*1/8*	*0*
6 chon	*15/16*	*1/16*	*0*
7 chon	*31/32*	*1/32*	*0*
8 chon	*63/64*	*1/64*	*0*

$$KI = \frac{\Pr(X, Y \mid Relationship)}{\Pr(X, Y \mid Unrelated)}$$

이것을 SQL로 만들 수 있을까? 포기!

빠른 검색을 위한 캐시 데이터 구조

pandas DataFrame

```
n [14]: astrs
Out[14]:
            CSF1PO     D13S317     D16S539      D18S51     D19S433  \
sample1  [100, 120]    [80, 80]   [90, 110]  [140, 160]  [130, 152]
sample2  [100, 130]   [90, 110]  [100, 130]  [130, 140]  [130, 140]
sample3  [110, 110]  [110, 130]  [120, 120]  [160, 190]  [130, 140]
sample4  [110, 120]   [80, 110]  [110, 130]  [140, 180]        None
sample5  [100, 120]   [80, 100]   [90, 130]  [140, 140]  [120, 152]
sample6  [100, 120]   [90, 110]  [100, 130]  [130, 140]  [130, 130]
sample7  [100, 120]  [100, 110]   [90, 130]  [140, 140]  [120, 152]

            D21S11     D2S1338     D3S1358      D5S818      D7S820  \
sample1        NaN  [170, 210]  [150, 170]  [110, 110]   [80, 100]
sample2  [300, 300]  [200, 240]  [150, 160]  [110, 130]  [110, 120]
sample3  [290, 300]  [200, 230]  [150, 180]   [90, 110]   [90, 110]
sample4  [290, 322]  [170, 190]        None        None        None
sample5  [290, 322]  [230, 230]  [150, 170]   [90, 110]  [110, 130]
sample6  [300, 300]  [170, 200]  [150, 160]  [110, 110]  [100, 110]
sample7  [290, 290]  [230, 230]  [170, 180]   [90, 110]  [100, 130]

            D8S1179         FGA      TH01       TPOX        XY         vWA
sample1  [110, 120]  [200, 230]  [70, 93]   [80, 80]  [10, 10]  [170, 170]
sample2        NaN  [230, 280]  [60, 90]  [80, 110]  [10, 20]  [170, 200]
sample3        NaN  [190, 220]  [70, 80]  [80, 110]  [10, 20]  [170, 180]
sample4        NaN  [212, 230]      None        None  [10, 20]  [140, 140]
sample5        NaN  [220, 240]  [70, 90]  [80, 110]  [10, 20]  [140, 180]
sample6        NaN  [220, 280]  [70, 90]   [80, 80]  [10, 20]  [140, 170]
sample7        NaN  [240, 240]  [60, 90]  [80, 110]  [10, 20]  [150, 180]
```

for 루프 대신 apply 메쏘드!

pandas.DataFrame.apply

`DataFrame.`**`apply`**(*func*, *axis=0*, *broadcast=False*, *raw=False*, *reduce=None*, *args=()*, ***kwds*)

Applies function along input axis of DataFrame.

Objects passed to functions are Series objects having index either the DataFrame's index (axis=0) or the columns (axis=1). Return type depends on whether passed function aggregates, or the reduce argument if the DataFrame is empty.

Examples

```
>>> df.apply(numpy.sqrt) # returns DataFrame
>>> df.apply(numpy.sum, axis=0) # equiv to df.sum(0)
>>> df.apply(numpy.sum, axis=1) # equiv to df.sum(1)
```

동일성 검색

불일치 좌위 갯수

```python
def count_mismatches(row, query):
    columns = row.notnull() & query.notnull()
    return (row[columns] != query[columns]).sum()

query = astrs.loc['sample1']
target = astrs.copy()

target['mismatches'] = target.apply(count_mismatches,
                                    query=query, axis=1)
result = target.loc[target['mismatches'] <= 2]

print(result)
```

DataFrame apply 함수로 일괄계산

친자 검색

동일성은 유전자형 일치여부 파악

```
def count_mismatches_identity(row, query):
    columns = row.notnull() & query.notnull()
    return (row[columns] != query[columns]).sum()
```

```
def count_mismatches_paternity(row, query):
    columns = row.notnull() & query.notnull()
    return sum(not (set(row[c]) & set(query[c])) for c in columns)
```

친자관계는 유전자형 공유여부 파악

친척 검색

가족관계별로 우도비 계산

Kinship Formulas:

$$[P_2(xy) \times \phi_2] + [P_1(xy) \times \phi_1] + [P_0(xy) \times \phi_0]$$

#1	#2	frequency
AB	AB	$\phi_2 + 0.5\phi_1(p_A + p_B) + 2\phi_0 p_A p_B$
AA	AA	$\phi_2 + \phi_1 p_A + \phi_0 p_A^2$
AA	AB	$\phi_1 p_B + 2\phi_0 p_A p_B$
AB	AC	$0.5\phi_1 p_C + 2\phi_0 p_A p_C$
AB	CD	$2\phi_0 p_C p_D$
AA	BB	$\phi_0 p_B^2$
AA	BC	$2\phi_0 p_B p_C$

IBD coefficient	Φ0	Φ1	Φ2
Parent-Child	*0*	*1*	*0*
Full-sib	*1/4*	*1/2*	*1/4*
Half-sib	*1/2*	*1/2*	*0*
4 chon	*3/4*	*1/4*	*0*
Unrelated	*1*	*0*	*0*
5 chon	*7/8*	*1/8*	*0*
6 chon	*15/16*	*1/16*	*0*
7 chon	*31/32*	*1/32*	*0*
8 chon	*63/64*	*1/64*	*0*

$$KI = \frac{\Pr(X, Y \mid Relationship)}{\Pr(X, Y \mid Unrelated)}$$

```
coefficients = {
    'parent-child': (0, 1, 0),
    'full-sib': (1./4, 1./2, 1./4),
    'half-sib': (1./2, 1./2, 0),
    'first cousin': (3./4, 1./4, 0),
    'unrelated': (1, 0, 0),
    '5 chon': (7./8, 1./8, 0),
    '6 chon': (15./16, 1./16, 0),
    '7 chon': (31./32, 1./32, 0),
    '8 chon': (63./64, 1./64, 0),
}
relationships = list(sorted(coefficients.keys()))
formulas = {
    1: (lambda phi, pa, pb, pc, pd:
            phi[2] + 0.5*phi[1]*(pa + pb) + 2*phi[0]*pa*pb ),
    2: (lambda phi, pa, pb, pc, pd: phi[2] + phi[1]*pa + phi[0]*pa*pa ),
    3: (lambda phi, pa, pb, pc, pd: phi[1]*pb + 2*phi[0]*pa*pb ),
    4: (lambda phi, pa, pb, pc, pd: 0.5*phi[1]*pa + phi[0]*pa*pa ),
    5: (lambda phi, pa, pb, pc, pd: 0.5*phi[1]*pc + 2*phi[0]*pa*pc ),
    6: (lambda phi, pa, pb, pc, pd: 2*phi[0]*pc*pd ),
    7: (lambda phi, pa, pb, pc, pd: phi[0]*pb*pb ),
    8: (lambda phi, pa, pb, pc, pd: 2*phi[0]*pb*pc ),
}
```

가족관계별 대립유전자 공유확률

유전자형 조합에 따른 공식

유전자형 조합에 따른 공식 찾기

```
def find_formula(query_alleles, target_alleles):
    a1, a2 = query_alleles
    b1, b2 = target_alleles
    if a1 == b1 and a2 == b2 and a1 != a2:
        f = 1; A = a1; B = a2; C = None; D = None
    elif a1 == b2 and a2 == b1 and a1 != a2:
        f = 1; A = a1; B = a2; C = None; D = None
elif a1 == a2 == b1 == b2:
    f = 2; A = a1; B = None; C = None; D = None
elif a1 == a2 and a1 == b1 and b1 != b2:
    f = 3; A = a1; B = b2; C = None; D = None
...
return f, A, B, C, D
```

```
def get_kinship_index(row, query, relationship):
    cpi = 1
    for locus in query[query.notnull()].index:
        f, A, B, C, D = find_formula(query[locus], row[locus])
        pa = korean_allele_frequencies[locus][A]
        pb = korean_allele_frequencies[locus][B]
        pc = korean_allele_frequencies[locus][C]
        pd = korean_allele_frequencies[locus][D]
        phi = coefficients[relationship]
        cpi *= formulas[f](phi, pa, pb, pc, pd)
    return cpi

def get_kinship_indices(row, query):
    return pd.Series([get_kinship_index(row, query, relationship)
            for relationship in relationships], index=relationships)

result = target.apply(get_kinship_indices, query=query, axis=1)
print(result)
```

가족관계별로 우도비(LR) 계산

가족관계별 우도비(LR) 계산 결과

	5 chon	6 chon	7 chon	8 chon	first cousin	full-sib	half-sib	parent-child	unrelated	
sample1	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000e+00	1.000000	1	1.000000	
sample2	0.000015	0.000020	0.000022	0.000024	0.000008	9.966914e-08	0.000002	0	0.000026	
sample3	0.000016	0.000020	0.000022	0.000023	0.000010	3.831698e-07	0.000003	0	0.000025	
sample4	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000e+00	1.000000	1	1.000000	
sample5	0.000930	0.001067	0.001139	0.001177	0.000683	7.588633e-05	0.000304	0	0.001214	
sample6	0.000019	0.000021	0.000023	0.000023	0.000014	1.514376e-06	0.000006	0	0.000024	
sample7	0.135726	0.145421	0.150269	0.152692	0.116337	3.877900e-02	0.077558	0	0.155116	

```
max_index = result.max(axis=1) / result['unrelated']
max_relative = result.idxmax(axis=1)
result['max_index'] = max_index
result['max_relative'] = max_relative
result = result[result['max_index'] > index_threshold]
result = result.sort(['max_index', 'max_relative'], ascending=False)
```

우도비최대값 필터링

우도비최대값으로 정렬

뿐만 아니라, 친척여부까지도 알 수 있는 시대

좀 더 빠르게

마침 고객 컴퓨터가

파이썬 멀티프로세싱!

```python
import multiprocessing
import numpy as np
import pandas as pd

def _apply_df(args):
    df, func, kwargs = args
    return df.apply(func, **kwargs)

def apply_by_multiprocessing(df, func, **kwargs):
    workers = multiprocessing.cpu_count()
    pool = multiprocessing.Pool(processes=workers)
    result = pool.map(_apply_df, [(d, func, kwargs)
              for d in np.array_split(df, workers)])
    pool.close()
    return pd.concat(list(result))

result = apply_by_multiprocessing(target, get_kinship_indices,
                                  query=query, axis=1)
```

DataFrame 쪼개고

각각 동시에
apply 실행하고

합치기

저장은 어떻게?

HDF5?

```
In [13]: astrs.to_hdf('astrs.h5', 'table')

In [14]: !ls -la astrs.h5
-rw-r--r--  1 yong27  staff  1069344  8 28 04:46 astrs.h5

In [15]:
```

- HDF5은 on-disk 데이터 포맷
- PyTables Query로 해결할 수 있음
- 그러나, 잦은 데이터의 변경이 어려움

pymongo

```
from pymongo import MongoClient
import pandas as pd

client = MongoClient('mongodb://localhost:27017/')
db = client['dbname']

## DB insert
for identifier, str_ in pd.read_csv(infile, index_col=0).iterrows():
    db.genotypes.insert({
                'identifier': identifier,
                'A-STR': str_.to_dict(),
                })

## DataFrame from mongodb
cursor = db.genotypes.find(
        {'A-STR': {'$exists': 1}})
astrs = pd.DataFrame.from_items(
        g['identifier'], g['A-STR'] for g in cursor).T
```

명령행 프로그램으로

16.4. `argparse` — Parser for command-line options, arguments and sub-commands

New in version 3.2.

Source code: Lib/argparse.py

The `argparse` module makes it easy to write user-friendly command-line interfaces. The program defines what arguments it requires, and `argparse` will figure out how to parse those out of `sys.argv`. The `argparse` module also automatically generates help and usage messages and issues errors when users give the program invalid arguments.

Tutorial

This page contains the API reference information. For a more gentle introduction to Python command-line parsing, have a look at the *argparse tutorial*.

16.4.1. Example ¶

The following code is a Python program that takes a list of integers and produces either the sum or the max:

```
import argparse

parser = argparse.ArgumentParser(description='Process some integers.')
parser.add_argument('integers', metavar='N', type=int, nargs='+',
                    help='an integer for the accumulator')
parser.add_argument('--sum', dest='accumulate', action='store_const',
                    const=sum, default=max,
                    help='sum the integers (default: find the max)')

args = parser.parse_args()
print(args.accumulate(args.integers))
```

```
./kinc -h: kinc [-h]
           {add-allele-frequencies,list-allele-frequencies,delete-allele-frequencies,show-allele-
frequencies,update-allele-frequencies,add-genotypes,search,search-multiple,check-between,report-hml}
           ...
commandline interface
arguments:
 -h, --help            show this help message and exit

subcommands:
 kinc has a few sub-commands

 {add-allele-frequencies,list-allele-frequencies,delete-allele-frequencies,show-allele-
frequencies,update-allele-frequencies,add-genotypes,search,search-multiple,check-between,report-hml}
                       -h for additional help
   add-allele-frequencies
                       Add allele frequencies data to DB
   list-allele-frequencies
                       List all available allele frequencies
   delete-allele-frequencies
                       Delete allele frequencies by the name
   show-allele-frequencies
                       Show the allele frequencies table
   update-allele-frequencies
                       Update allele frequencies by using current DB
   add-genotypes       Add many genotype data in batch file (generated by
                       GeneMark)
   search              Search relationship.
   search-multiple     Search relationship between groups.
   check-between       Check relationship between two queries.
   report-hml          Report checking result to HML format ('-a' option for
                       customer, '-b' for goldstar !!).
2013, Insilicogen, Inc.
```

```
./kinc add-genotypes -h: kinc add-genotypes [-h] -i INFILE [-f {GeneMark,JSON,CSV,XLSX}]
                         [-p {A-STR,Y-STR,SNP,mtDNA}] [-c COLUMNS] [-w] [-t]
                         [-g {goldstar,customer,anonymous-staff}]
arguments:
 -h, --help            show this help message and exit
 -i INFILE, --infile INFILE
                       Input file name, tab seperated machine generated txt
                       file
 -f {GeneMark,JSON,CSV,XLSX}, --format {GeneMark,JSON,CSV,XLSX}
                       Input file format (default is GeneMark text export)
 -p {A-STR,Y-STR,SNP,mtDNA}, --type {A-STR,Y-STR,SNP,mtDNA}
                       Type of genotypes (A-STR or Y-STR)
-c COLUMNS, --columns COLUMNS
                       Specify the column number (ID,Marker,Allele1,Allele2)
                       of GeneMark text export file (default: 0,4,6,7)
-w, --overwrite       Overwrite previous genotypes by identifier
-t, --initial         Initial setting. It deletes all previous data.
-g {goldstar,customer,anonymous-staff}, --group {goldstar,customer,anonymous-staff}
                       Group name of individuals.
```

```
./kinc search -h: kinc search [-h] -r {identity,paternity,kinship} -g
                  {goldstar,customer,anonymous-staff,all} -q QUERY -n NAME
                  [-u] [-ma PERMIT_MISMATCHES_ASTR] [-pa]
                  [-my PERMIT_MISMATCHES_YSTR] [-py]
                  [-mm PERMIT_MISMATCHES_MTDNA] [-pm] [-em]
arguments:
 -h, --help            show this help message and exit
 -r {identity,paternity,kinship}, --relationship-type {identity,paternity,kinship}
                       Relationship type
 -g {goldstar,customer,anonymous-staff,all}, --group {goldstar,customer,anonymous-staff,all}
                       Group name of individuals.
 -q QUERY, --query QUERY
                       Query identifier.
 -n NAME, --name NAME  The name of this allele frequencies data.
 -u, --partial         If this option is stated, then NA value will not be
                       filtered out.
 -ma PERMIT_MISMATCHES_ASTR, --permit-mismatches-astr PERMIT_MISMATCHES_ASTR
                       Permitted mismatch counts in genotype 'A-STR'. If this
                       option is stated, then filtering will be applied.
                       (cannot use this option in relationship_type
                       'kinship')
-pa, --partial-astr   Partial option in genotype 'A-STR'. (cannot use this
                       option in relationship_type 'kinship')
-my PERMIT_MISMATCHES_YSTR, --permit-mismatches-ystr PERMIT_MISMATCHES_YSTR
                       Permitted mismatch counts in genotype 'Y-STR'. If this
                       option is stated, then filtering will be applied.
-py, --partial-ystr   Partial option in genotype 'Y-STR'.
-mm PERMIT_MISMATCHES_MTDNA, --permit-mismatches-mtdna PERMIT_MISMATCHES_MTDNA
                       Permitted mismatch counts in genotype 'mtDNA'. If this
                       option is stated, then filtering will be applied.
-pm, --partial-mtdna  Partial option in genotype 'mtDNA'
-em, --exclude-cstretch-mtdna
                       Partial option in genotype 'mtDNA' this option require
                       'mm' option.
```

서버 프로그램으로

요청마다 메모리 로드할 필요 없음

django-rest-framework

- Resource
  - Genotype (유전자형)
  - Allele frequency (대립유전자빈도)
  - Group (유전자형 그룹)
  - Task (작업)

- Search API
  - Check (1:1 검사)
  - Search (1:다 검색)
  - Search multiple (다:다 검색)

RESTful API - Genotype 목록

List

URL: /genotypes/?{{ type=(A-STR|Y-STR|mtDNA) }}&{{ format=(csv|tsv|json) }}&{{ identifier=1212 }}&{{ }}&{{ desc=(true|false) }}&{{ filter-astr=(true|false) }}&{{ filter-ystr=(true|false) }}&{{ filter-mtdna=(tru

- 입력 type=(A-STR|Y-STR|mtDNA), format=(csv|tsv|json)
- 출력 genotypes 데이터 요약 or type 데이터
- pagination
    - ex) identifier=12121&page-no=1&paginate-by=10&sort=Y-STR&desc=true
    - page-no : page 번호
    - paginate-by : page당 나오는 item 개수
    - sort : 정렬 대상
    - desc=(true|false) : 차순
    - filter-astr=(true|false)
    - filter-ystr=(true|false)
    - filter-mtdna=(true|false)

```
identifier, groups, a-str 개수, y-str 개수, mtdna 개수
```

or

```
{
type : {}
}
```

- json일 경우, 실행 결과와 실행 결과 데이터를 출력
- csv, tsv는 데이터 요약 또는 type의 genotypes 데이터 출력

RESTful API - Genotype 상세

Detail

URL: /genotypes/{{ identifier=(identifier) }}/?{{ type=(A-STR|Y-STR|mtDNA) }}&{{ format=(csv|tsv|json)

- 입력 identifier=(identifier), type=(A-STR|Y-STR|mtDNA), format=(csv|tsv|json), kit=(all|ab|promega)
- 출력 identifier, type의 genotypes 데이터 출력

```
{
'group':[],
'A-STR':dataframe,
'Y-STR':dataframe,
}
```

or

```
{
'A-STR':dataframe
}
```

- json일 경우, 실행 결과 정보와 실행 결과 데이터를 출력
- csv, tsv일 경우, type의 genotypes 데이터 (format 이 csv, tsv 일 경우 type 이 명시 되어야 함)

RESTful API - Genotype 입력

Post

URL: /genotypes/

유전자형 자원을 입력한다.입력은 문서이다. 다음 옵션으로 보낸다. save-by=(new)

- new : 새로운 마커 추가
- replacement : 마커에 해당하는 값을 모두 대체
- merge-with-overwrite : 같은 마커는 덮어쓰고, 없는 마커는 새로 추가
- merge-without-overwrite : 같은 마커는 덮어쓰지 않고, 없는 마커는 새로 추가

- 입력 : txt(GeneMark?), csv, xlsx 형식의 genotypes 데이터 파일
    - 헤더 정보
        - type=(A-STR|Y-STR|mtDNA) : 유전자 종류
        - informat=(GeneMark?|XLSX|CSV)
        - group
        - save-by=(new)
        - infile : 파일

- 출력 : 실행 결과 출력
- json : 실행 결과 출력

RESTful API – Genotype 변경

Put

URL: /genotypes/

유전자형 자원을 갱신한다. 갱신은 문서이거나 개별 genotype 데이터이다.

개별 genotype 데이터을 추가할 때 다음과 같은 옵션이 있고 매개변수를 선택해야 한다.

save-by=(new|replace|merge-overwrite|merge-no-overwrite)

- new : 새로운 마커 추가
- replacement : 마커에 해당하는 값을 모두 대체
- merge-with-overwrite : 같은 마커는 덮어쓰고, 없는 마커는 새로 추가
- merge-without-overwrite : 같은 마커는 덮어쓰지 않고, 없는 마커는 새로 추가

- 입력 : txt(GeneMark?), csv, xlsx 형식의 genotypes 데이터 파일
    - 헤더 정보
        - type=(A-STR|Y-STR|mtDNA) : 유전자 종류
        - informat=(GeneMark?|XLSX|CSV)
        - group
        - save-by=(replacement|merge-with-overwrite|merge-without-overwrite)
        - infile : 파일

- 출력 : 실행 결과 출력
- json : 실행 결과 출력

웹어플리케이션으로

유전자형 : [redacted]

좌위정렬 All

A-STR

좌위	유전자형
Amelogenin	X,Y
D3S1358	17,18
D1S1656	-
D2S441	-
D10S1248	-
D13S317	10,11
Penta E	-
D16S539	9,13
D18S51	14,14
D2S1338	23,23
CSF1PO	10,12
Penta D	-
TH01	6,9
vWA	15,18
D21S11	29,29
D7S820	10,13
D5S818	9,11
TPOX	8,11
D8S1179	-
D12S391	-
D19S433	12,15.2
FGA	24,24
D22S1045	-
SE33	-

변경

Y-STR

좌위	유전자형
DYS576	-
DYS389I	11
DYS448	20
DYS389II	27
DYS19	14
DYS391	10
DYS481	-
DYS549	-
DYS533	-
DYS438	11
DYS437	15
DYS570	-
DYS635	20
DYS390	23
DYS439	12
DYS392	14
DYS643	-
DYS393	12
DYS458	19
DYS385	15,19
DYS456	15
GATAH4.1	12

변경

mtDNA

구분	위치	유전자형
HV1 16024~16365	16319	A
	16290	T
	16362	C
	16213	A
	16223	T
HV2 73~340	152	C
	263	G
	200	G
	73	G
	235	G
HV3 438~574	표시할 자료가 없습니다.	
ETC	표시할 자료가 없습니다.	

변경

검색

검색 > 검색

식별번호	
검사옵션	검색종류 친척관계 대상그룹 부분검색
A-STR	A-STR 검사 허용불일치수 부분검색 최소혈연지수 대립유전자빈도표 Korean
Y-STR	Y-STR 검사 허용불일치수 부분검색
mtDNA	mtDNA 검사 허용불일치수 부분검색 C-stretch 영역 제외

조회

Total: 0

10개씩 보기

식별번호	비교분석	A-STR		Y-STR		mtDNA	
		대립유전자	불일치수	대립유전자	불일치수	대립유전자	불일치수
표시할 게시물이 없습니다.							

1

- 웹 어플리케이션 제작 방식의 변화
    1. 명령행 프로그램
    2. RESTful API
    3. 웹어플리케이션
    - 코어는 파이썬으로 구현됨. 웹 UI는 원하는 언어로
    - 유전자 검사 관련 다양한 프로젝트에 통합 가능

에필로그

개선 희망 사항

- 메모리 로드 방식 대신 mongodb 쿼리
- DataFrame 셀에 “리스트” 대신 “정수형”

- 오후 BoF에서 “좋은 아이디어” 받습니다.

DNA 프로파일 정보는

- 이것만으로 “표현형질” 확인 불가
- 본인 확인 뿐 아니라 친척여부도 확인할 수 있음
- 매우 중요한 개인정보임

INSILICOGEN

www.insilicogen.com

|주| 인실리코젠

DSC 데이터 사이언스 센터 (부설 R&D 센터) / 전문연구요원(병역특례) 인가

데이터 사이언스 시대! Design Your Value from Big Data!

- Thanks for your attention

김형용
(주)인실리코젠 (**We are hiring!**)
Twitter: @yong27
Blog & Wiki: http://e.biohackers.net