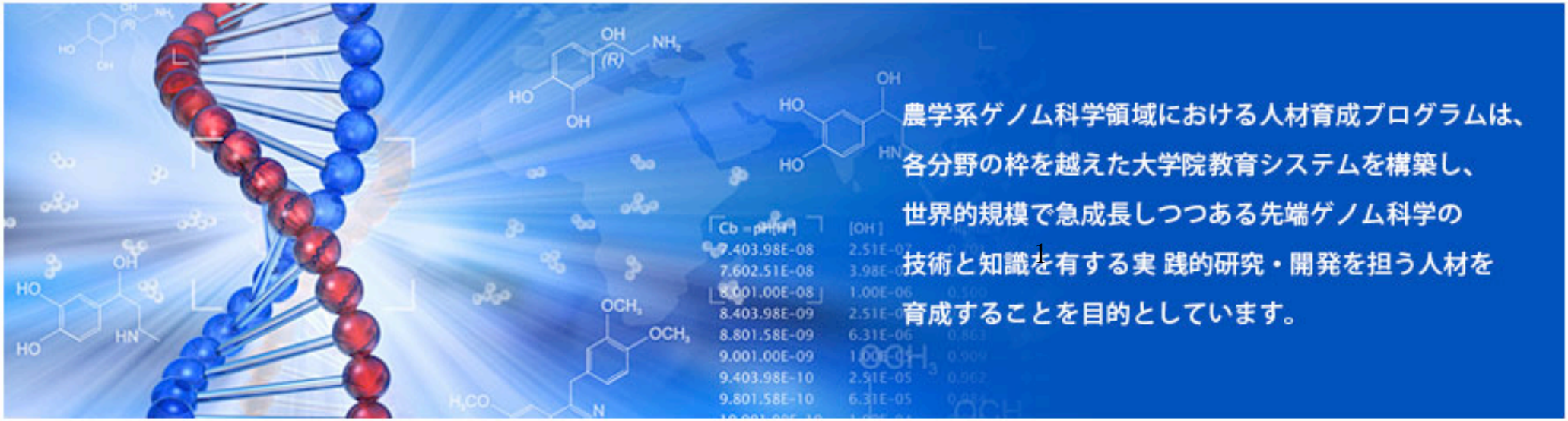
医療、農業、環境分野における ビッグデータ分析技術の研究開発

石井一夫　東京農工大学

2014年1月24日
総務省四国総合通信局
四国情報通信懇談会ICT研究交流フォーラムセミナー

徳島市ホテル千秋閣



本日の内容

１、概要
２、事例１　DNA自動解析装置の品質管理
３、事例２　進化系統樹の作成の最適化と
　　生物種の識別
４、事例３　精神神経系疾患の診断系確立
５、事例４　人材育成｜情報処理学会
　ビッグデータ活用実務フォーラム

自己紹介

東京農工大学農学府農学部特任教授

徳島市生、
徳島大学大学院医学研究科博士課程修了後
東京大学医科学研究所、理化学研究所、
フランス国立遺伝子多型解析センターCNG、
米国ノースウエスタン大学Feinberg医学部
などを経て現職。
専門：ゲノム科学、バイオインフォマティクス、
計算機統計学。

我々のビッグデータ処理の新しい産業応用
広告やゲーム、レコメンだけではない

個別化医療（ライフサイエンス）：
　精神神経系疾患の網羅的ゲノム診断法の開発

ゲノム育種（グリーンサイエンス）：
　イネなどの新品種の開発

環境アセスメント（エコサイエンス）：
　環境微生物の分布、分類、生態調査

ゲノム科学におけるビッグデータ分析

本研究室では、大規模データ解析に対し以下の4方法で対応している。
HPCも、モンテカルロも、クラウドも使えるものは何でも使う主義。

1. モンテカルロシミュレーション：大量データから無作為に
サンプルを抽出し、元のデータをシミュレーション

2. Big iron (大容量メモリサーバ)による大量並列処理（HPC）
HP社の協力により、4TBメモリ、CPU: Xeon E7 (80 コア、
160スレッド)の大容量メモリ解析サーバを使用

3. Hadoop による分散処理システム（クラウドを利用）
Amazon Elastic MapReduce（Amazon EMR）プラットフォーム
（後日発表）

4. Hadoop によらない分散処理システム
シェルスクリプトベースの分散処理。usp-BOA （USP研究所）を
利用。大量データのクオリティチェック（後日発表）

本日は、上記の1,2,の方法の適用例を紹介する。

次世代シーケンサーの例
(超高性能DNA自動解読装置)

イルミナ社　GAIIx, MiSeq,
ロシュダイアグノスティクス社　454
ライフテクノロジーズ社　IonProton
など

GAIIx

MiSeq

東京農工大に設置されている機器

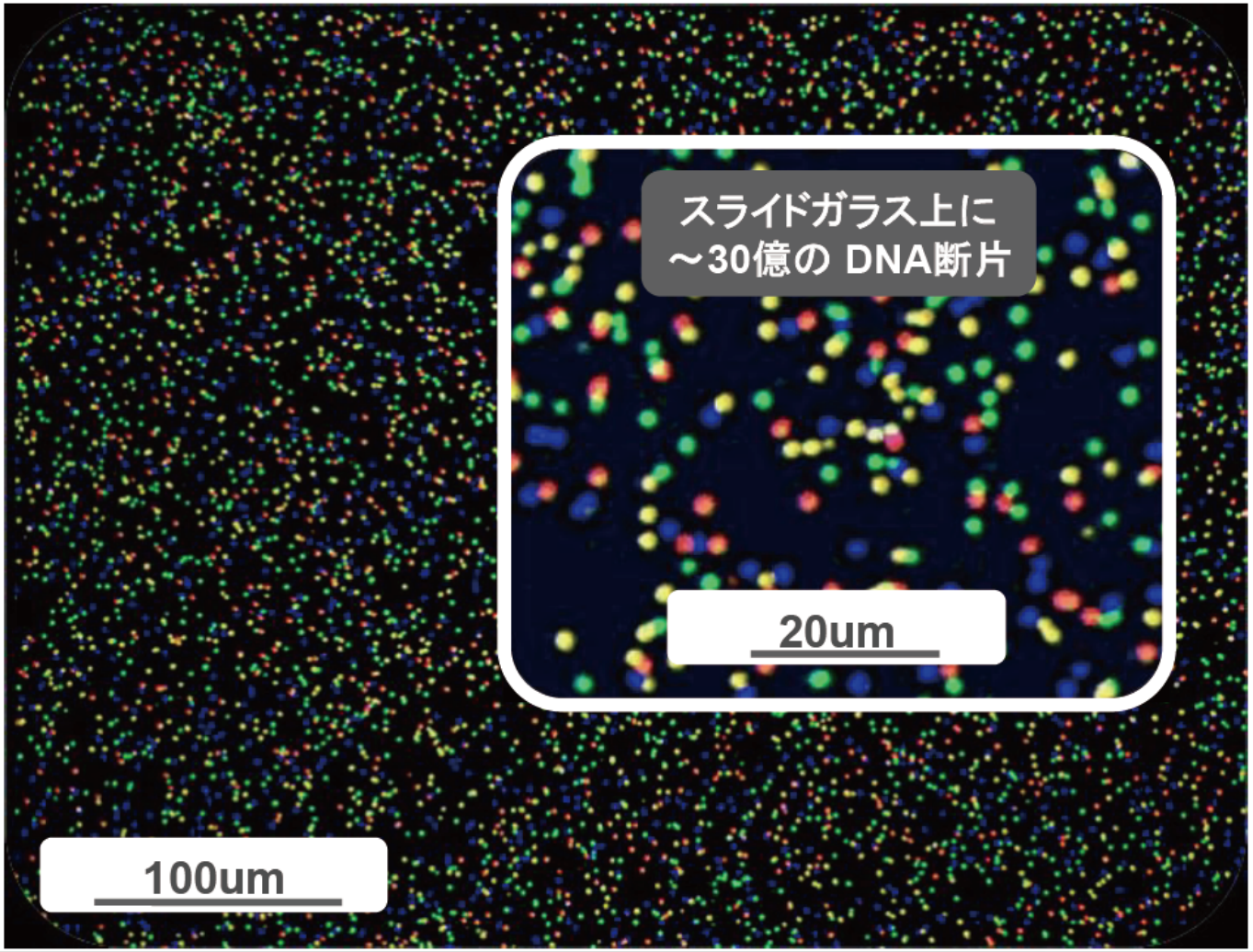
次世代シーケンサー

固相基板上に、DNA断片を固相化し、これを蛍光色素+酵素反応などを用いて、同時並列的に解読。CCD カメラで撮影+コンピュータで処理。

一度に、約数十～数千塩基のDNA断片を数十万から数億のDNA断片同時に解読できます。

数日から、数時間でヒトゲノム DNAを30億塩基解読することも可能。

次世代シーケンサーのDNA解読画像

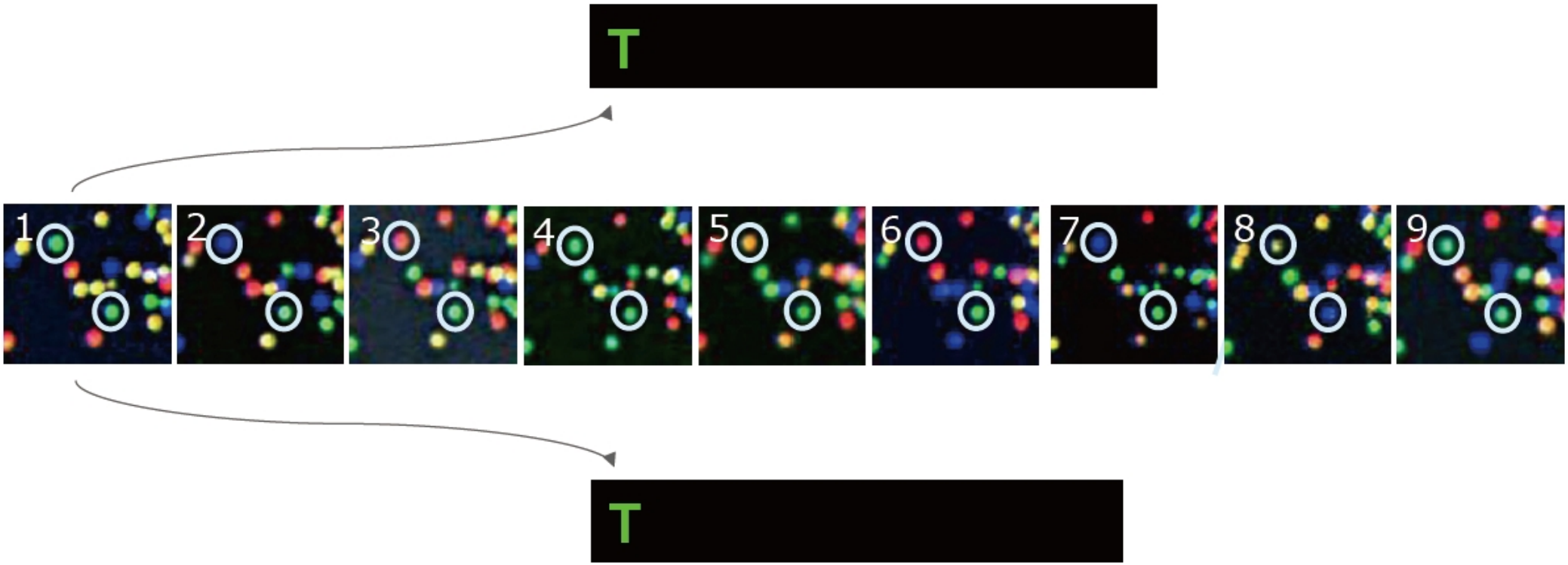


イルミナ社資料より

次世代シーケンサーのDNA解読画像

色のイメージからDNAの配列を決定

T C A T G A C T T



イルミナ社資料より

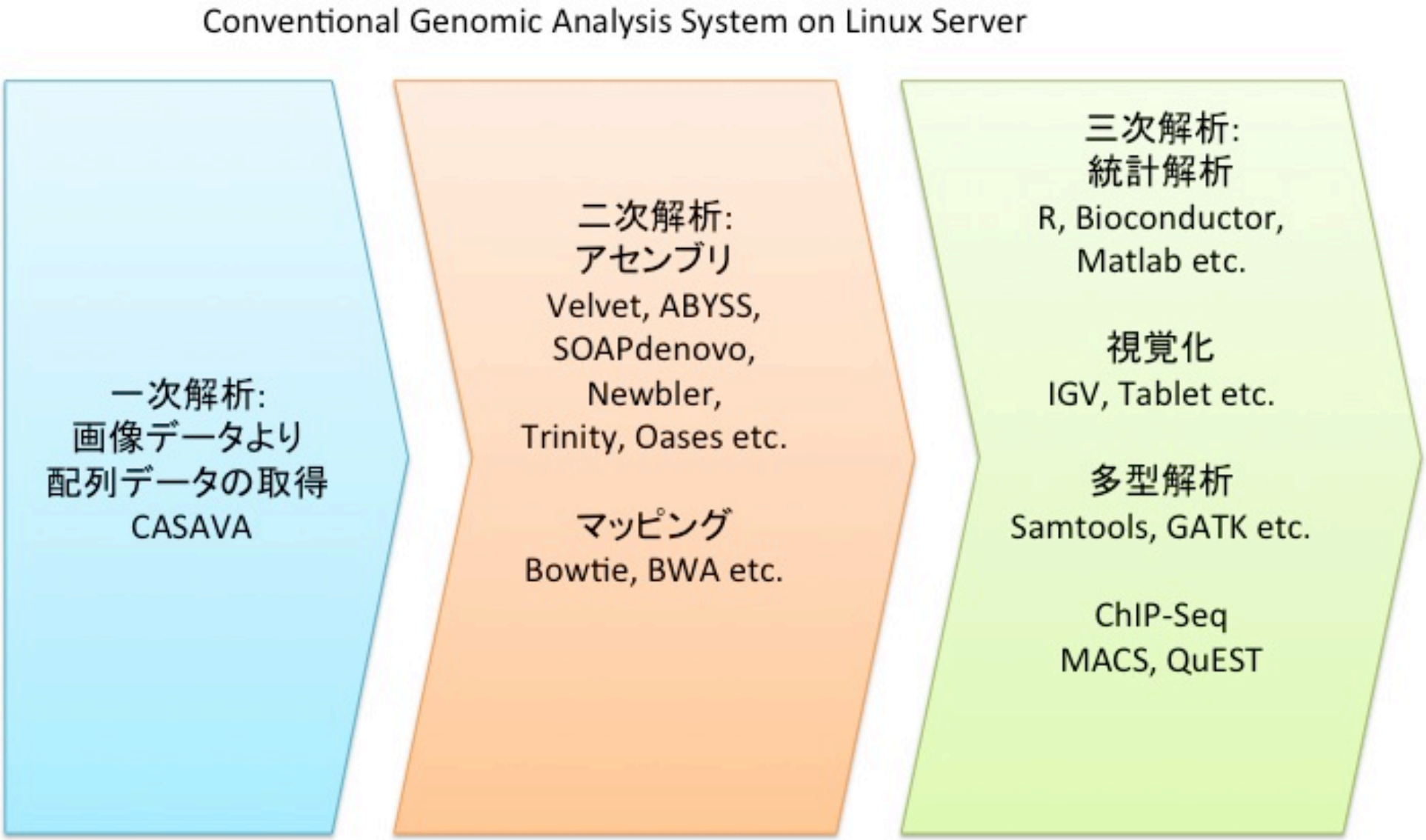
次世代シーケンサーのデータ解析

データ解析システムは主に、Linux をベースとしたフリーソフトウェアを用いて構築。

テキスト処理と数値計算を駆使し、awk, grep, cut, cat, sedなどのコマンドと、Perl/Python/Ruby などのスクリプト言語、S-PLUS/R、Octave(Matlab) などの統計解析ソフトを活用する。

専用解析ソフトも多く開発、C、C++、Java などで、開発。

データ解析は３段階



使用するソフトウェア群

3段階のデータ解析

1段階　画像処理　→　DNA塩基配列取得

２段階　DNA塩基配列を連結、整列、編集
　アセンブリ　配列を相互に連結
　マッピング　参照配列に整列

3段階　統計処理、視覚化、
　　データマイニング

解析ソフト群

1段階　画像処理　→　DNA塩基配列取得

CASAVA

２段階　配列を集計、編集

アセンブリ　配列を相互に連結

Velvet、SOAPdenovo、Trinity、、

マッピング　参照配列に整列

Bowtie、BWA、、、

3段階　統計処理、視覚化、

データマイニング　S-PLUS、BLAST、、

Hadoopを用いた
次世代シーケンサーのデータ解析

コンピュータクラスタや、クラウドの活用。

Hadoop MapReduce などのビッグデータフレームワークの活用例も増えてきている。

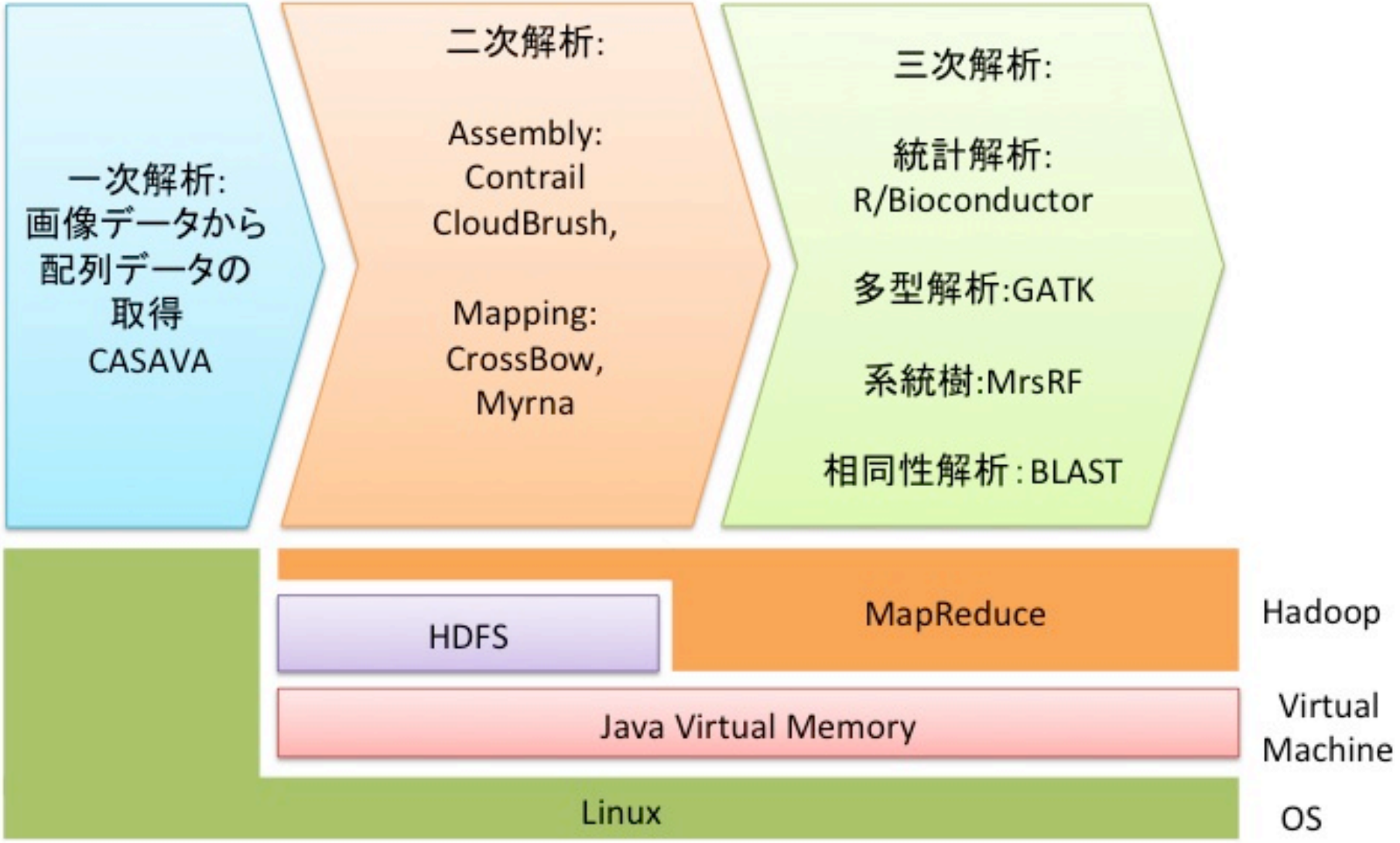
なぜ クラウド??

個人レベル、1研究室レベルで、コンピュータクラスタを構築することは、コスト的に無理。

クラウドでしか、コンピュータクラスタを使えない。

Hadoop を個人レベルで構築するひとつの方法として、クラウドサービスの利用を検討。

Hadoop 上で動作する解析システム

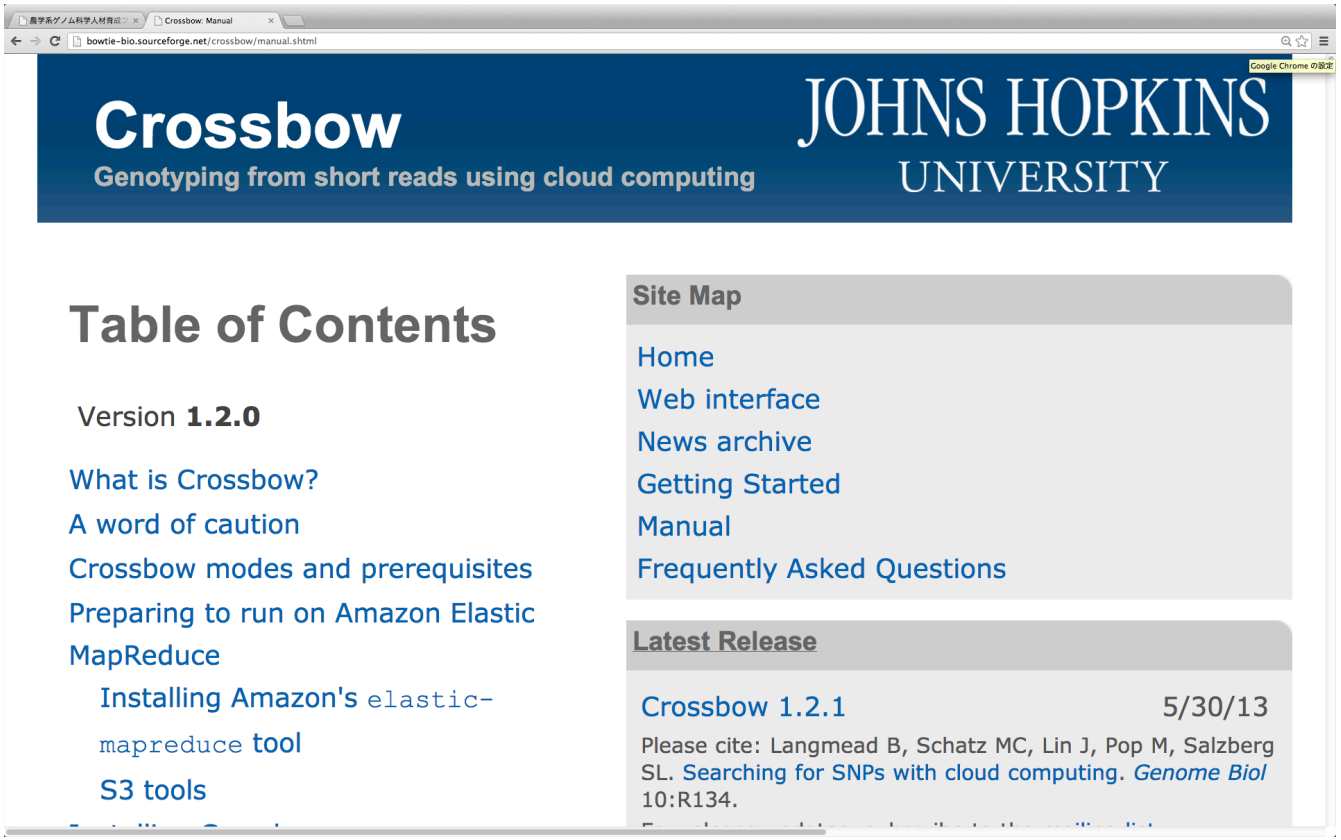


使用するソフトウェア群

16

Crossbow

Hadoopを使用して次世代シーケンサーデータをマッピングし、多型を検出するソフトウェア。

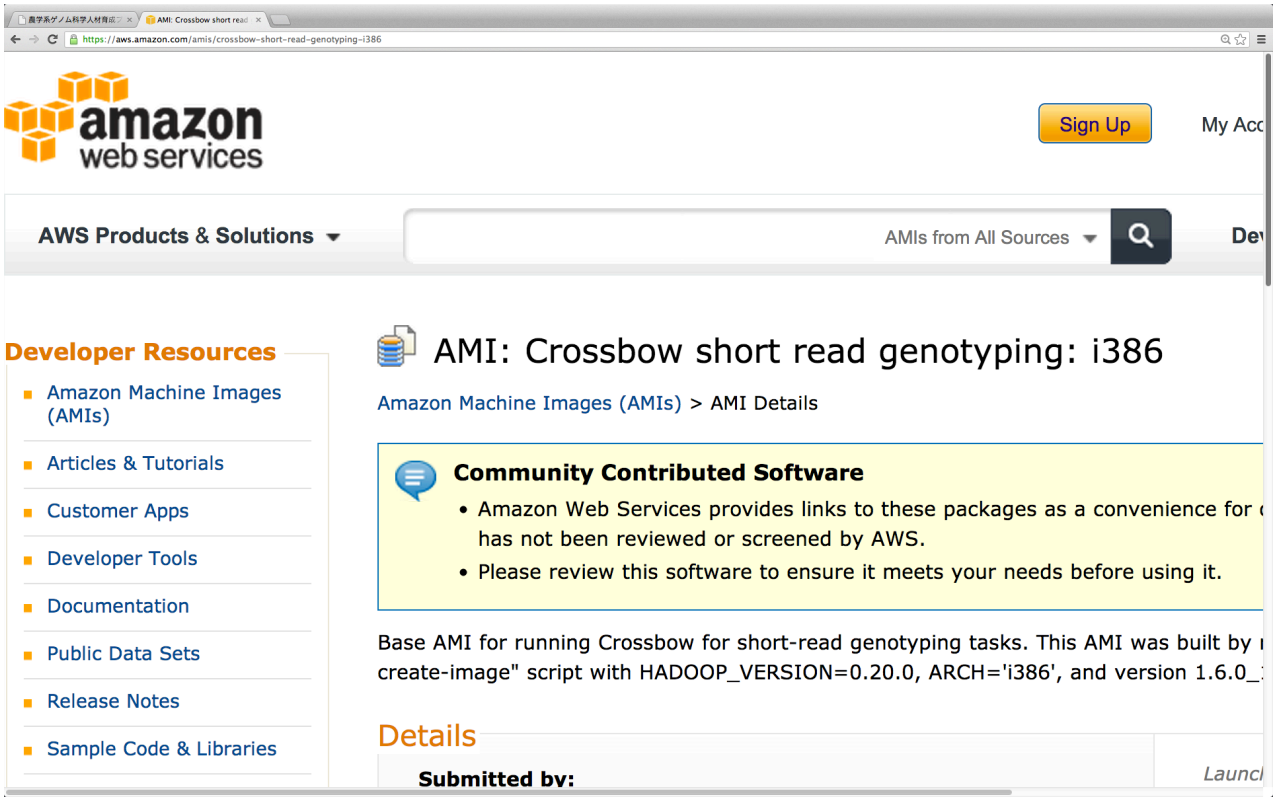


http://bowtie-bio.sourceforge.net/crossbow/index.shtmlより

AWSでのCrossbowの利用

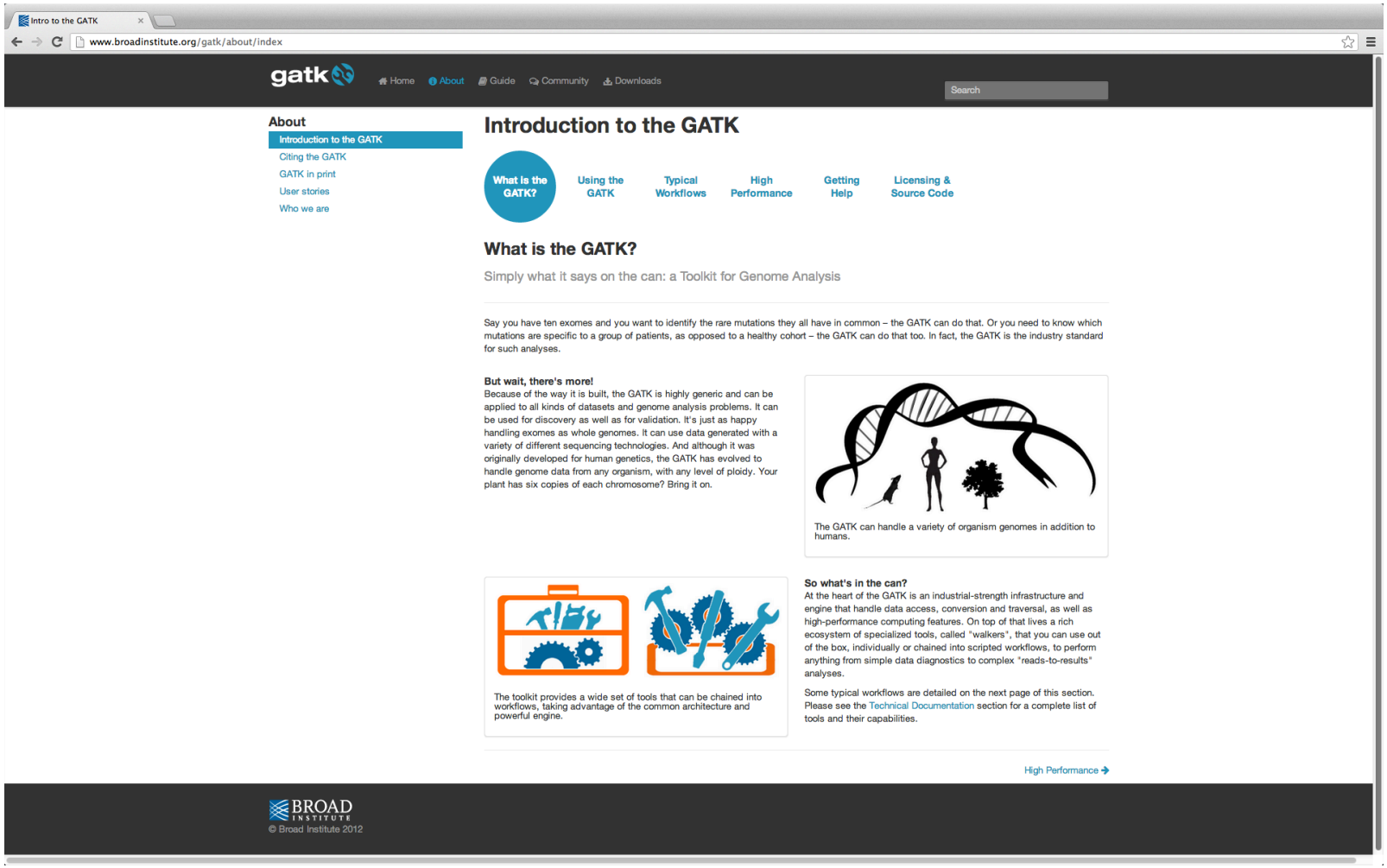
Hadoopを使用して次世代シーケンサーデータをマッピングし、多型を検出するソフトウェア。

Crossbow



GATK (Genome Analysis Toolkit)

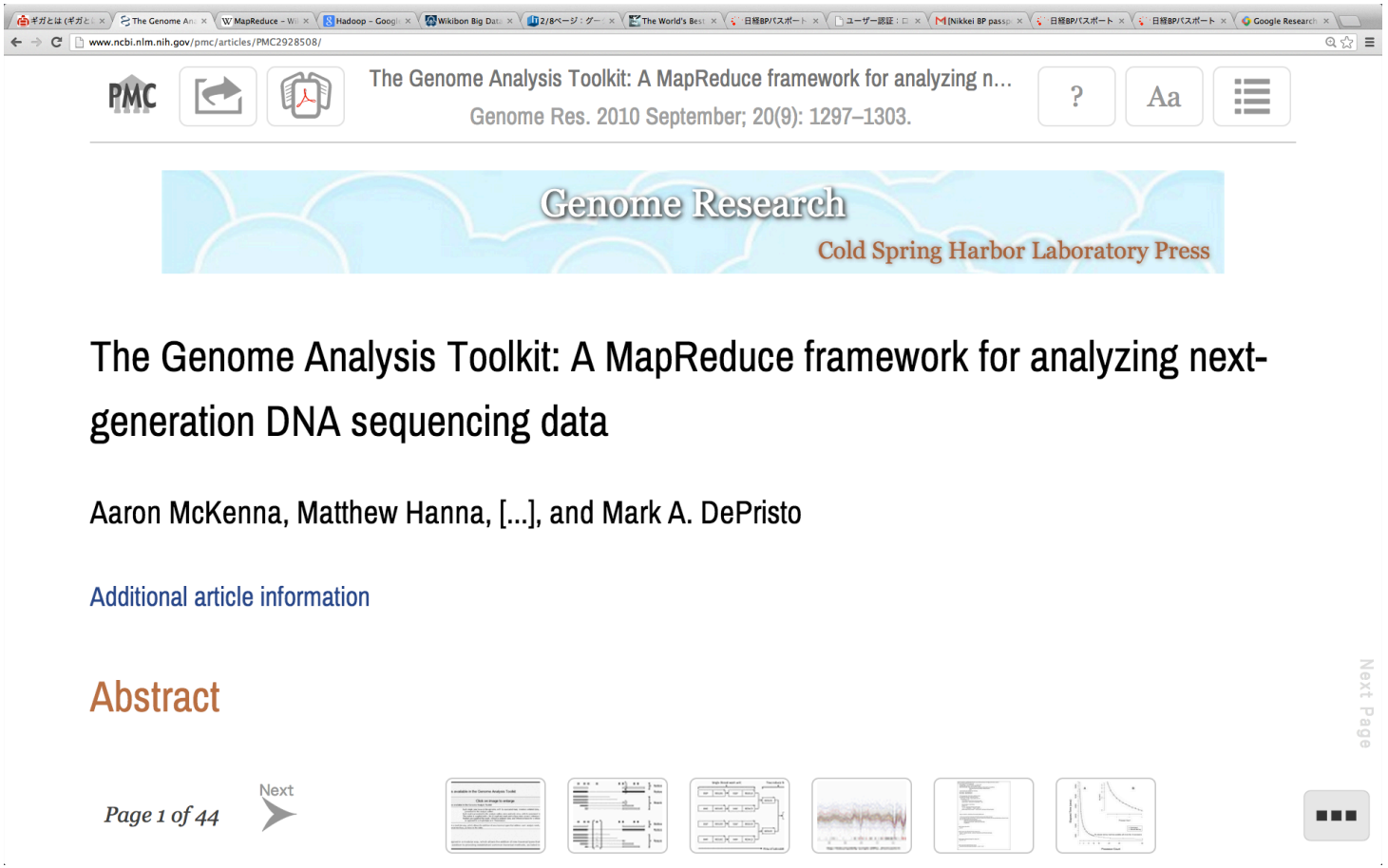
Javaベースで動作するMapreduceのフレームワークを取り入れた遺伝子多型解析用ソフトウェア



http://www.broadinstitute.org/gatk/ より引用

GATK の論文

Javaベースで動作するMapreduceのフレームワークを取り入れた遺伝子多型解析用ソフトウェア



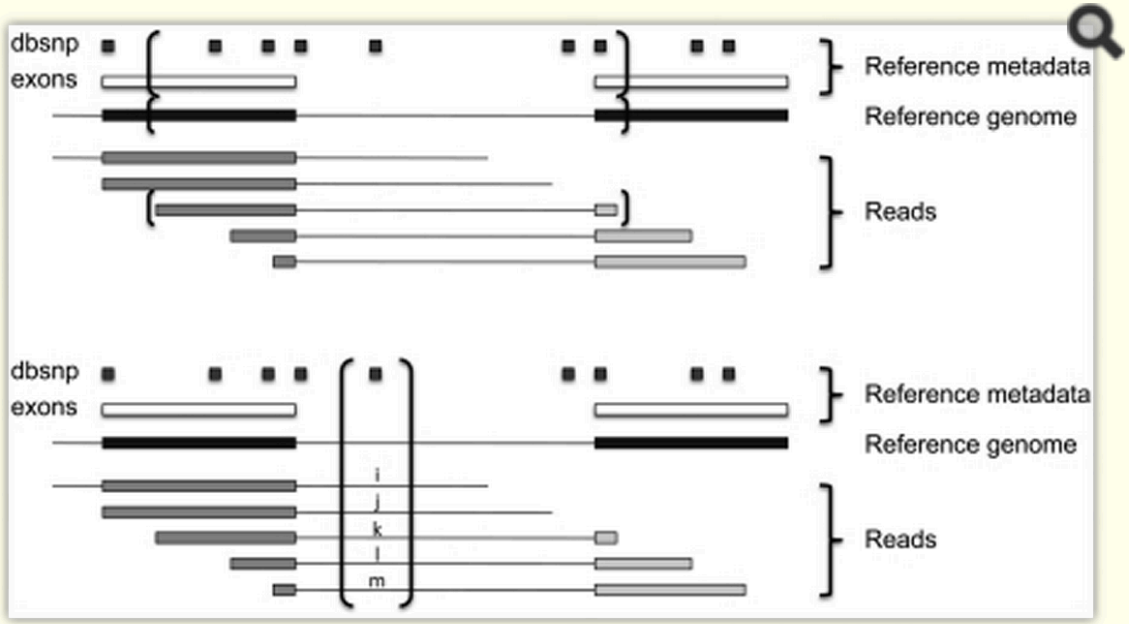
Genome Res. 2010 Sep;20(9):1297-303. doi: 10.1101/gr.107524.110. Epub 2010 Jul 19. より

GATKによるSNPコーリング（GATK の論文より）

www.ncbi.nlm.nih.gov/pmc/articles/PMC2928508/#!po=3.40909

The Genome Analysis Toolkit: A MapReduce framework for analyzing n...

Figure 1.



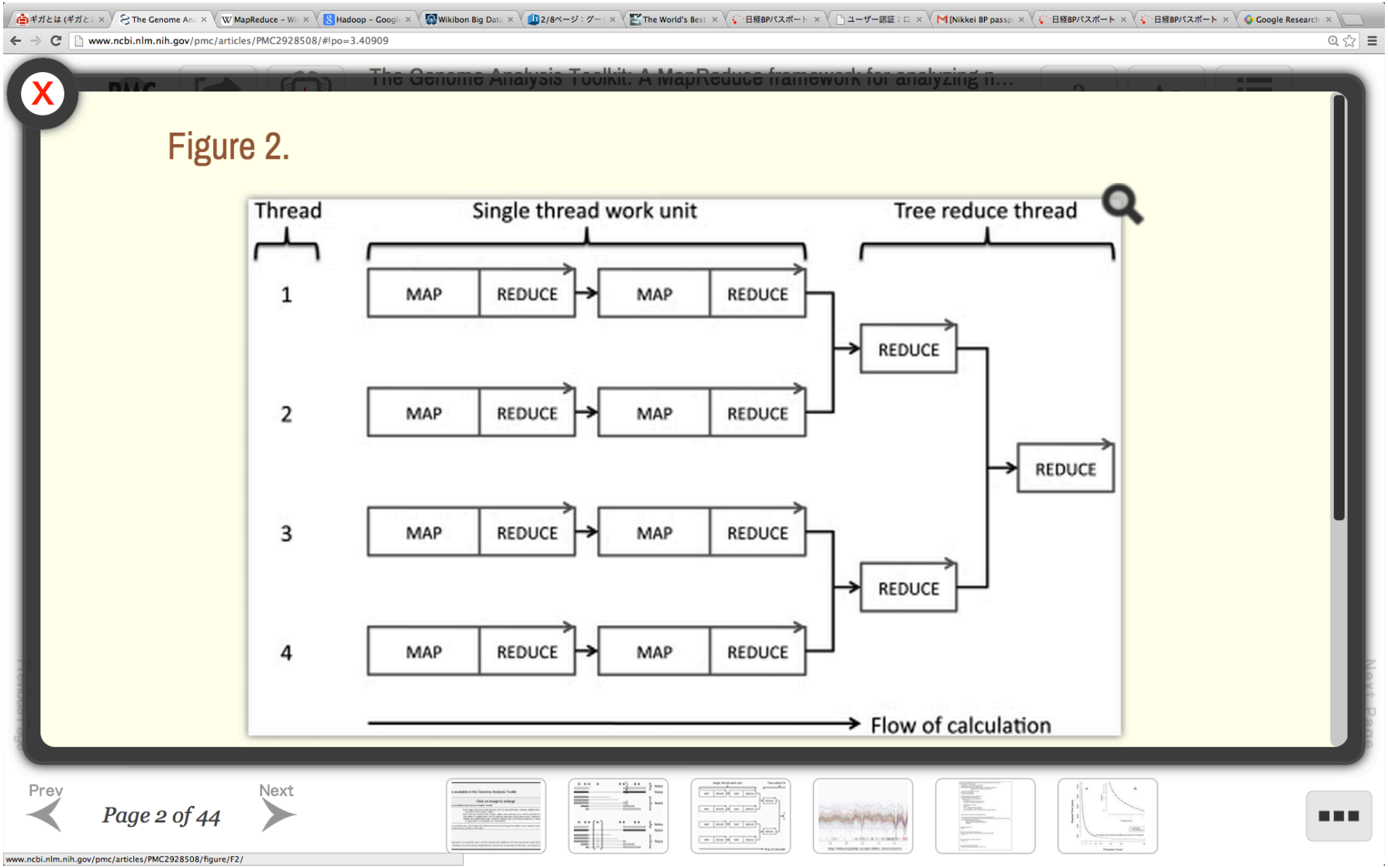
Read-based and locus-based traversals. Read-based traversals provide a

Prev Page 2 of 44 Next

www.ncbi.nlm.nih.gov/pmc/articles/PMC2928508/figure/F1/

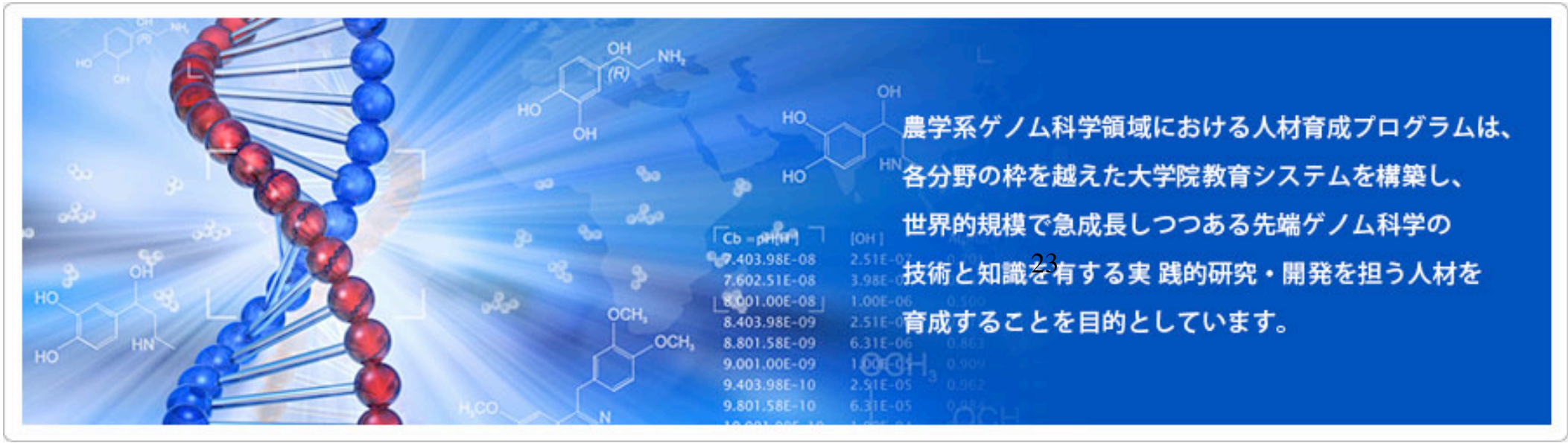
Genome Res. 2010 Sep;20(9):1297-303. doi: 10.1101/gr.107524.110. Epub 2010 Jul 19.より

MapReduceをモデルにしたGATKのプロセス（GATK の論文より）



Genome Res. 2010 Sep;20(9):1297-303. doi: 10.1101/gr.107524.110. Epub 2010 Jul 19.より

解析例　I
次世代シーケンサーデータの品質管理



次世代シーケンサーデータの品質管理

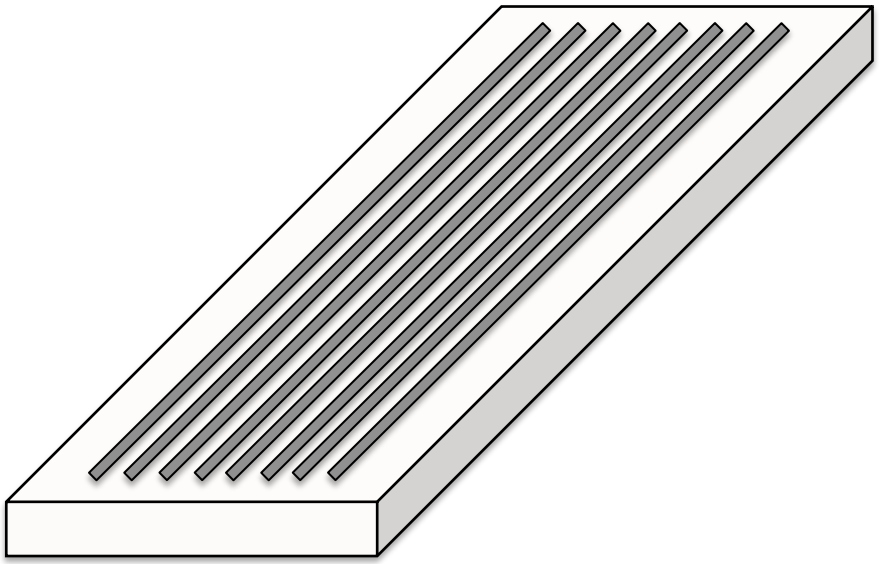
次世代シーケンサーデータのクォリティチェックに関する問題点。

イルミナ社の場合
フローセル上に DNA 反応クラスタを作らせる。
（1）サンプル濃度の間違い、（2）試薬濃度の間違い、（3）操作の荒さ、（4）電圧の不適、（5）データ転送のコマ落ち

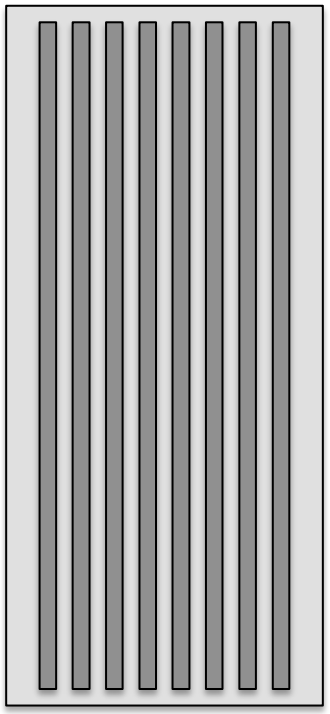
様々な原因により、クオリティ悪化が発生。

次世代シーケンサーデータの品質管理

イルミナ社の場合
フローセル上に DNA 反応クラスタを作らせる。



フローセルの図
（斜め上から見た図）



フローセルの図
（上から見た図）

次世代シーケンサーデータの品質管理

ランニングコストが高いために、詳細なチェックを行ない、クオリティのいいものだけを使用する工夫が必要。

数十～数百塩基のDNA 断片を数十万～数億断片のクオリティチェック
→　計算機的にかなり高い負荷

効率のよいチェックを行ない悪品質データの除去を行なうことが必要
→　データのサンプリングなどの工夫。

次世代シークエンサーから出力される配列データ

●fastqファイル

```
@HWI-ST977:153:D0J0KACXX:4:1101:1636:1851 1:N:0:TAGCTT
NGGTCCGGCTTTGAACCCCTGACAGGAAGGTATTATGCTGATCACGATG
CAACATGACAGATCGGCTCATGAAGCTTGGACTTGCTGTTCTCCTCTTTA
CG
+
#4=BDFFFGHHHHHJJIJJIJHIJJJGGIIFGIIIIJJIIJJJJJJIJJJJJJJJGIJHGHEFF
FDDCCEC@ACACDDDDCCCDDAACCDDCACDCDCC?
@HWI-ST977:153:D0J0KACXX:4:1101:1705:1877 1:N:0:TAGCTT
NTATCTTGACAGATTTTCTAGACTCATCCCAAGTTCTTGACCTAGCGCTG
ACAGAATTTGCTAAAATATGCTTATTCCGGTGCCAACTCCGTGGTATGCC
A
+
#1=DFFFFHHHHHJJJJJJJJJJJJJJJJJJIIEGHIHGGIJJJIIJJIJJJIJIIGJJJJIJGI
JJJJIJJIFHEFHGFFDDEEDDDDDDDDDDDDDDD
```

4行で１組

配列名

DNA配列

記号

クオリティデータ

4行１組の配列データが並び、全体で数百万～数億行に及ぶ

従来のクォリティチェック(FASTQC)

FASTQ ファイルのPhred quality score をもとに評価

Phred score = - 10 * log10(error probability)

0~40段階：40が最良、0が最悪

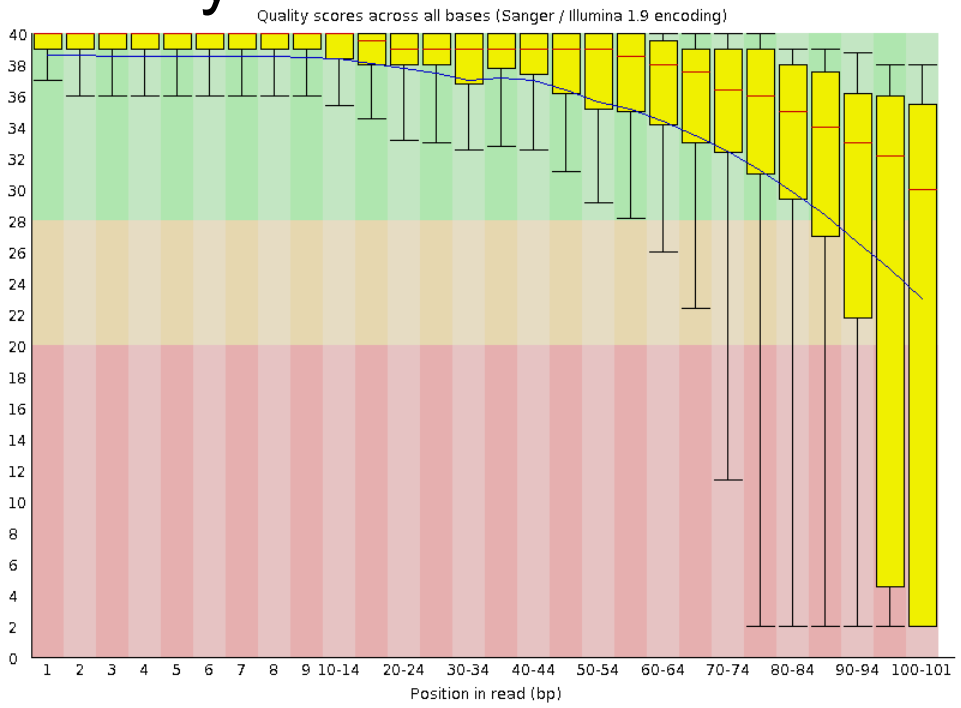
40字のアルファベット＋数字＋記号で表現

最初の20万リードの平均値でデータを評価

配列データのクオリティチェック

FastQC　→　配列データのクオリティチェックを行うフリーソフトフェア

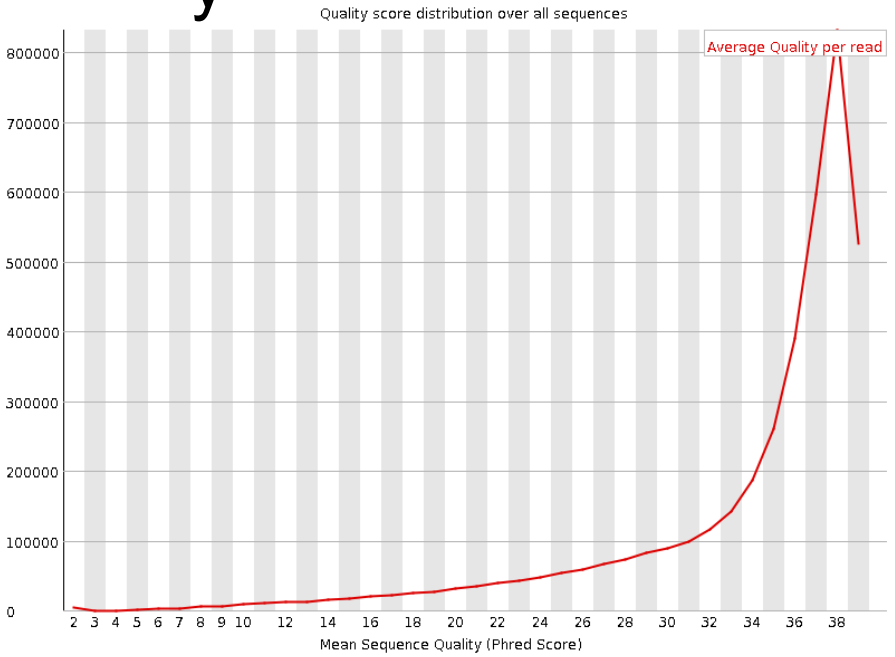
per_base_quality



1文字目

DNA配列

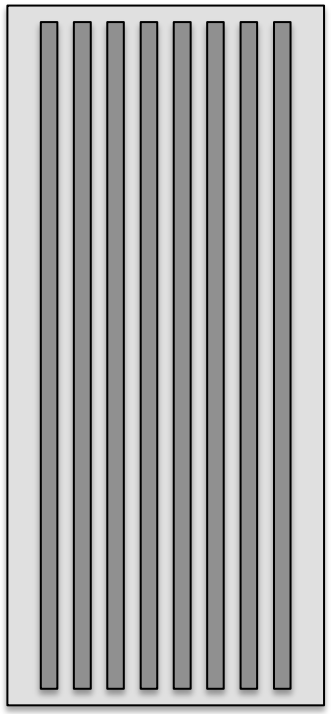
per_base_quality



Quality Score

クオリティチェックの問題点

GAIIxの場合には、フローセル上でDNA反応クラスタを作る



フローセルの図
（上から見た図）

（１）サンプル濃度の間違い

（２）試薬濃度の間違い

（３）操作の荒さ

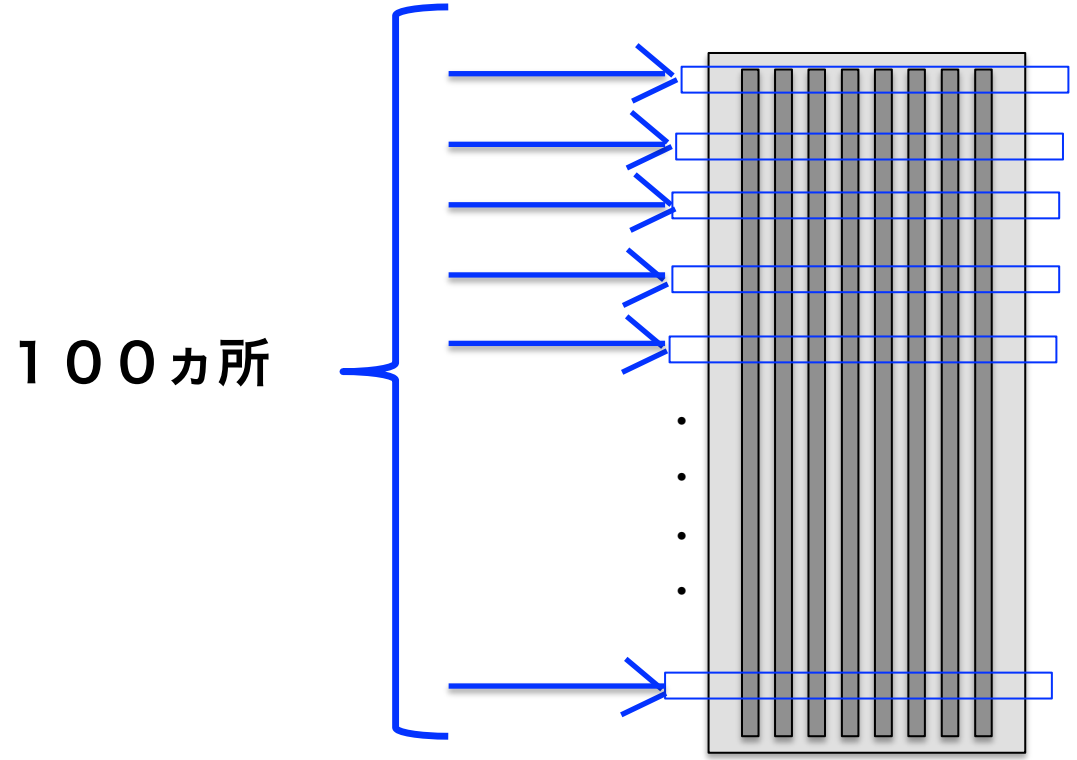
などの原因によりクオリティが悪化することがある。

研究目的

配列データが数千万個に及ぶ場合には、クオリティデータ全体を正確に把握することが困難

ファイル全体のクオリティデータを評価する方法を確立する

モンテカルロ法によるクオリティチェック①

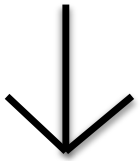


モンテカルロ法

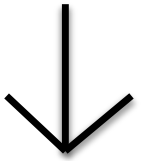
全体を100等分し、100カ所から1000個ずつリードを抽出

モンテカルロ法によるクオリティチェック②

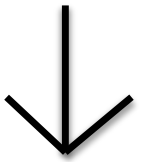
統計解析ソフト「R」を使用し乱数を発生



乱数に基づき、fastqファイルから無作為に
クオリティデータを抽出



クオリティデータを数値データに変換

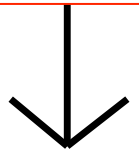


Rの関数hist（）、density（）、heatmap（）などを使用して全体のクオリティを評価

S-PLUS/Rによる乱数の発生

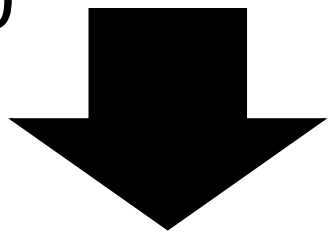
使用サンプル：L197 （Read1：15057415 reads、Read2：15057415 reads)

15057415read



100区間に等分

1 ～ 150574、150575～311148、・・・・、14906872～15057400



各区間において、Rの関数runif()を使用して、それぞれ1000個の乱数を発生させる（一様乱数）。
さらに、発生させた100区間分の乱数のデータをcbind()によっての結合し、100行×1000列の乱数表としてまとめた。

乱数表によるクオリティデータの抽出

- Fastqデータ

@HWI-ST977:153:D0J0KACXX:4:1101:1636:1851 1:N:0:TAGCTT
NGGTCCGGCTTTGAACCCCTGACAGGAAGGTATTATGCTGATCACGATG
CAACATGACAGATCGGCTCATGAAGCTTGGACTTGCTGTTCTCCTCTTTA
CG
+
#4=BDFFFGHHHHHJJIJJIJHIJJJGGIIFGIIIIJJIIJJJJJJIJJJJJJJJGIJHGHEFF
FDDCCEC@ACACDDDDCCCDDAACCDDCACDCDCC?
@HWI-ST977:153:D0J0KACXX:4:1101:1705:1877 1:N:0:TAGCTT
NTATCTTGACAGATTTTCTAGACTCATCCCAAGTTCTTGACCTAGCGCTG
ACAGAATTTGCTAAAATATGCTTATTCCGGTGCCAACTCCGTGGTATGCC
A
+
#1=DFFFFHHHHHJJJJJJJJJJJJJJJJJJIIEGHIHGGIJJJIIJJIJJJIJIIGJJJJIJGI
JJJJIJJIFHEFHGFFDDEEDDDDDDDDDDDDDDDD

4列で１組

リード名

DNA配列

記号

クオリティデータ

⟶ Fastqからクオリティデータのみを抽出する

⟶ 作製した乱数表の数字に当たる行のクオリティデータを抽出

100区間で1000個のクオリティデータを、無作為に抽出

クオリティデータの数値変換

無作為に抽出したデータをperlのスクリプトを使用して数値データに変換

クオリティデータ

#4=BDFFFGHHHHHHJJIJJIJHIJJJGGIIFGIIIIJJIIJJJJJJIJJJJJJJJGIJHGHEFFFDD
CCEC@ACACDDDDCCCDDAACCDDCACDCDCC?

Table 1 ASCII Characters Encoding Q-scores 0–40

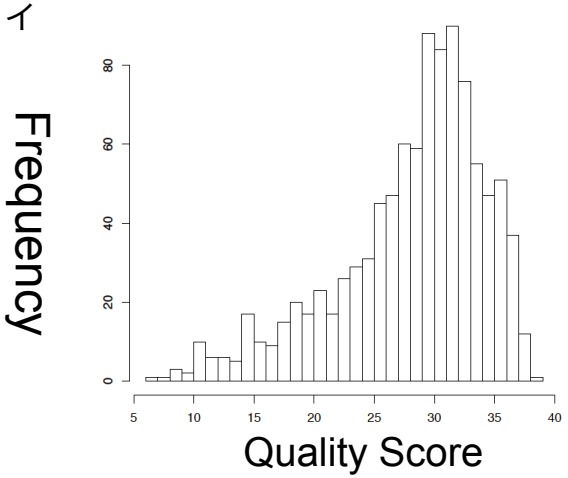
Symbol	ASCII Code	Q-Score	Symbol	ASCII Code	Q-Score	Symbol	ASCII Code	Q-Score
!	33	0	/	47	14	=	61	28
"	34	1	0	48	15	>	62	29
#	35	2	1	49	16	?	63	30
$	36	3	2	50	17	@	64	31
%	37	4	3	51	18	A	65	32
&	38	5	4	52	19	B	66	33
'	39	6	5	53	20	C	67	34
(	40	7	6	54	21	D	68	35
)	41	8	7	55	22	E	69	36
*	42	9	8	56	23	F	70	37
+	43	10	9	57	24	G	71	38
,	44	11	:	58	25	H	72	39
-	45	12	;	59	26	I	73	40
.	46	13	<	60	27			

得られたクオリティの数値データから各リードのクオリティの平均値を計算し、それを基にヒストグラム、密度推定、ヒートマップを作製し、クオリティデータを評価した。

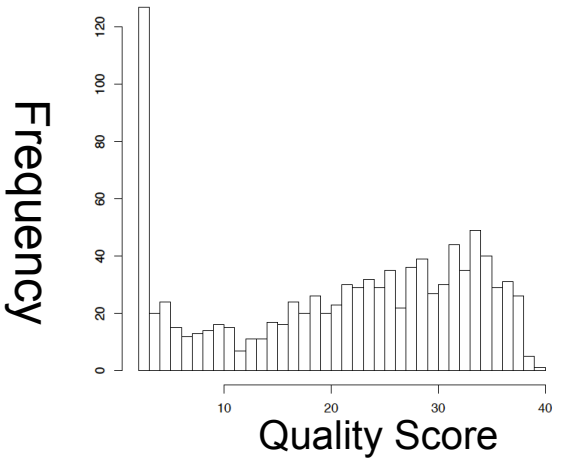
ヒストグラムによる評価

Rの関数hist()を使用して各区間でのヒストグラムを作製した。

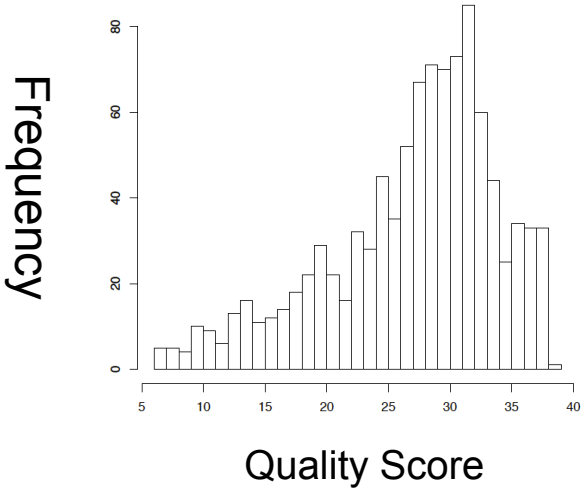
R1の1番目の区間（1～150574）のクオリティ



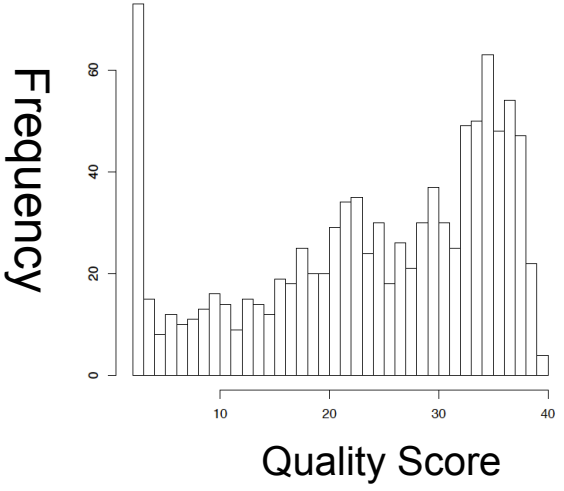
R2の1番目の区間（1～150574）のクオリティ



R1の41番目の区間（12045921 ～12196494 ）のクオリティ



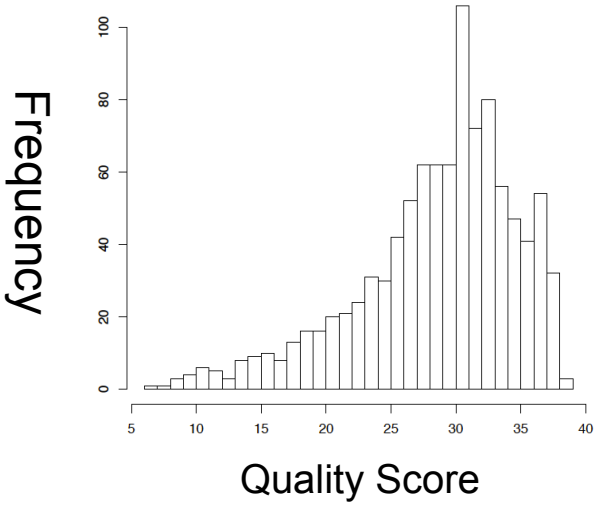
R2の81番目の区間（12045921 ～12196494 ）のクオリティ



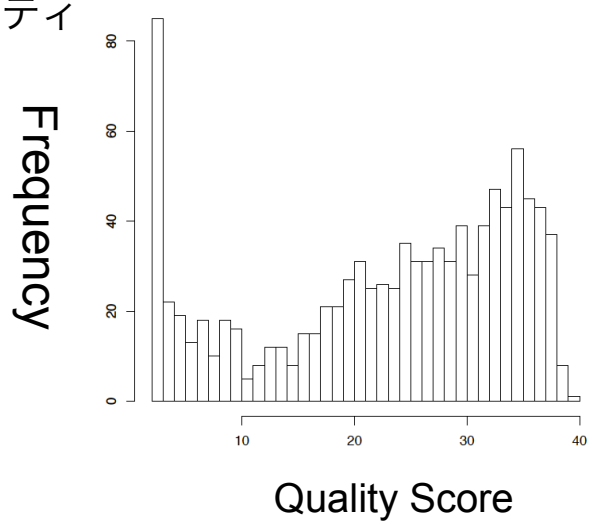
密度推定による評価

Rの関数hist()及びdensity()を使用して各区間でのクオリティチェックを行った。

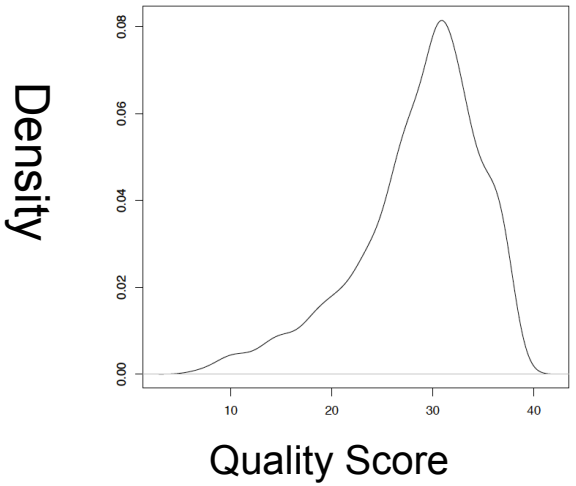
R1の41番目の区間（12045921 ～12196494 ）のクオリティ



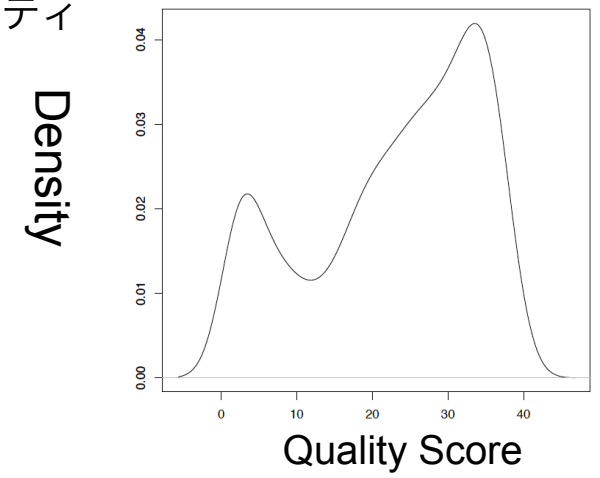
R2の81番目の区間（12045921 ～12196494 ）のクオリティ



R1の41番目の区間（12045921 ～12196494 ）のクオリティ



R2の81番目の区間（12045921 ～12196494 ）のクオリティ

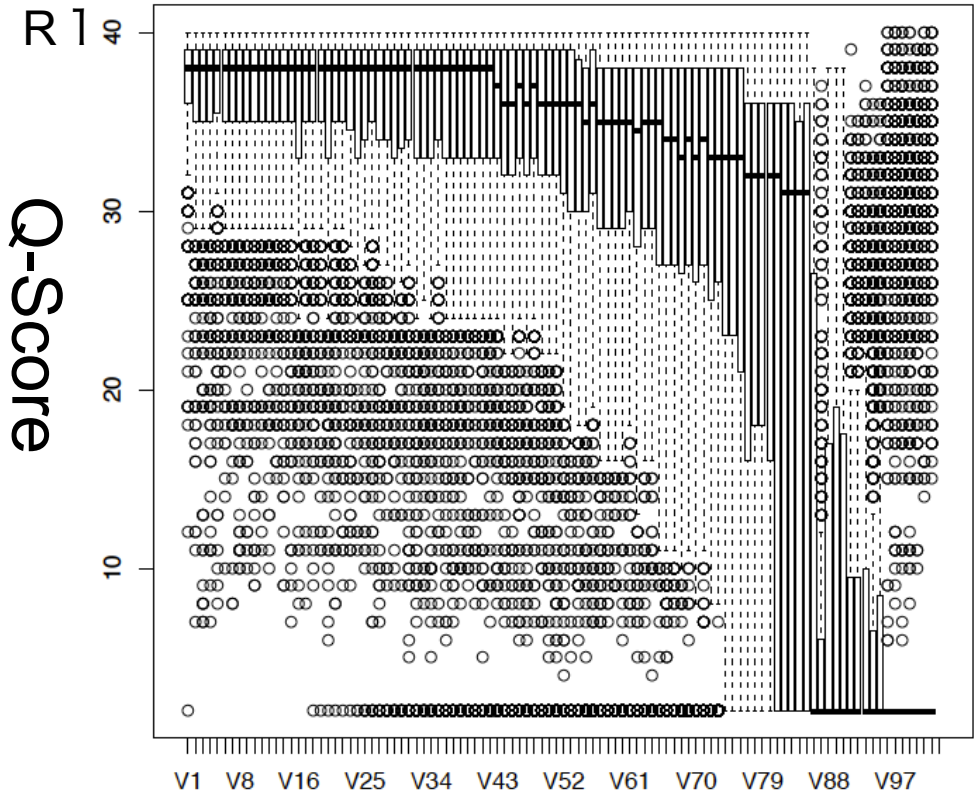


箱ひげ図による評価

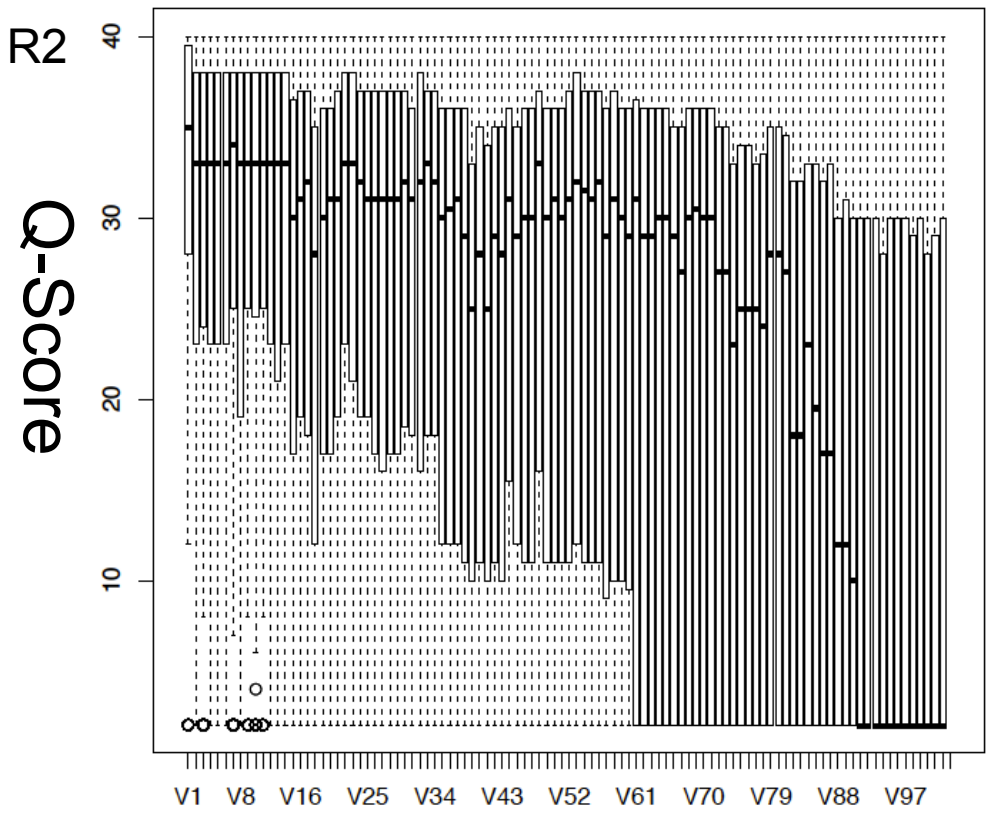
Rの関数boxplot()を使用して各区間での各塩基長ごとの
を行った。

DNA配列

NTATCTTGACAGATTTTCTAGACTCATCCCAAGTTCTTGACCTAGCGCTGAC
AGAATTTGCTAAAATATGCTTATTCCGGTGCCAACTCCGTGGTATGCCA

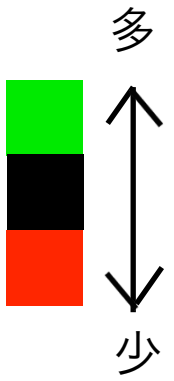


1文字目 → DNA配列



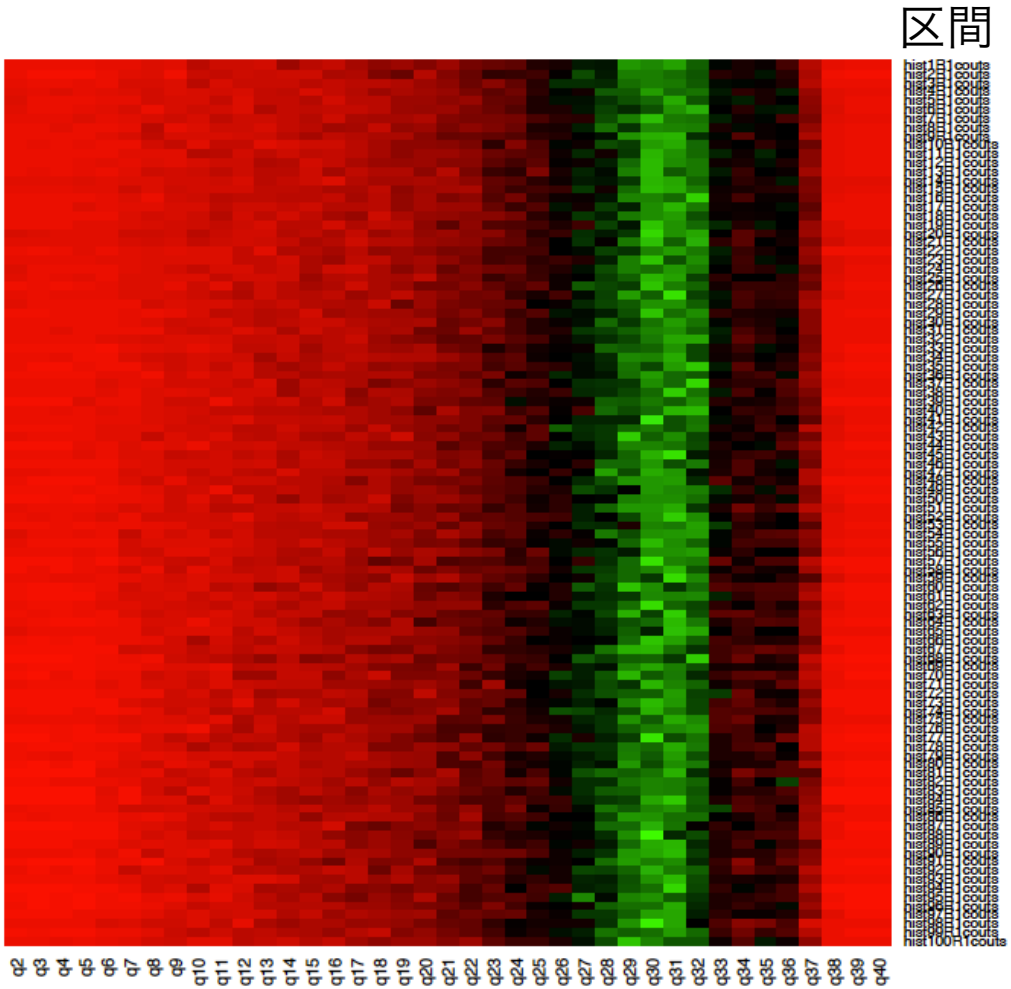
1文字目 → DNA配列

ヒートマップによる評価

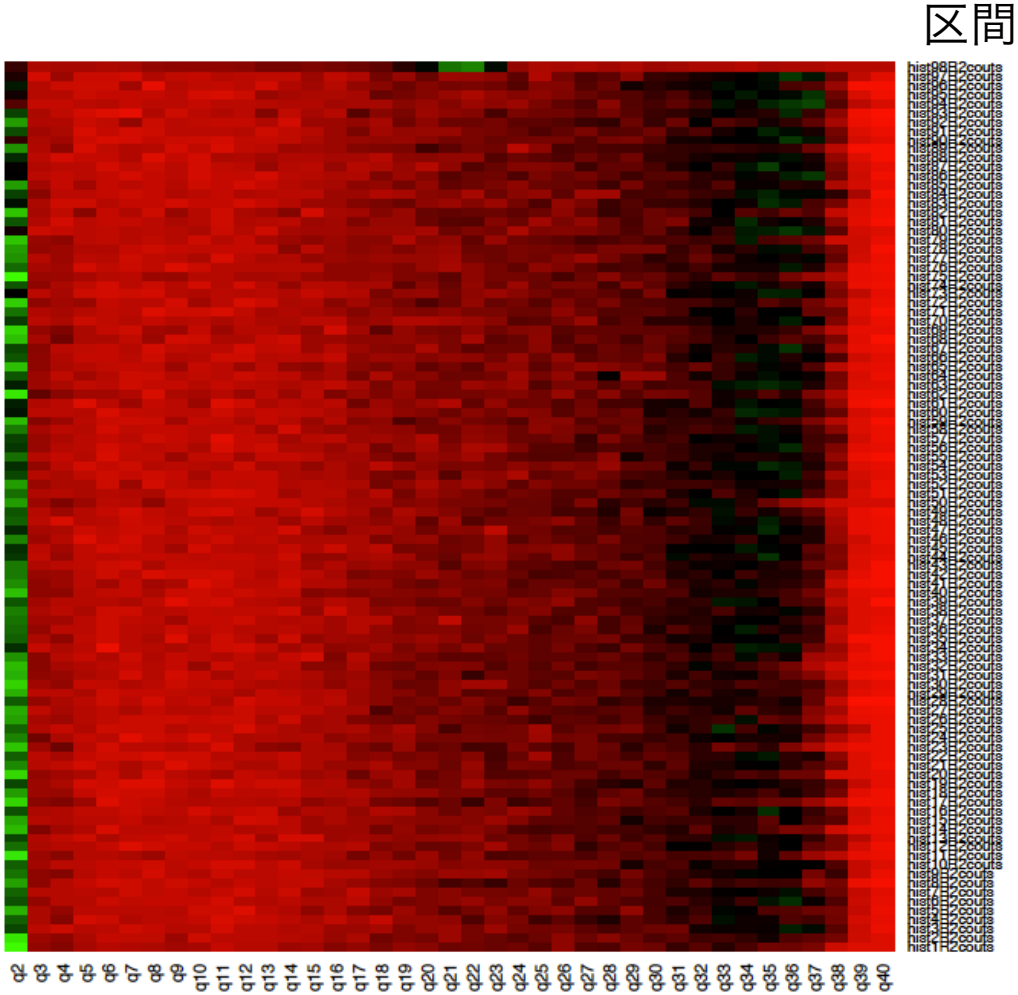


heatmap()関数を使用して全体のヒートマップを作製した。

R1のクオリティデータ



R2のクオリティデータ



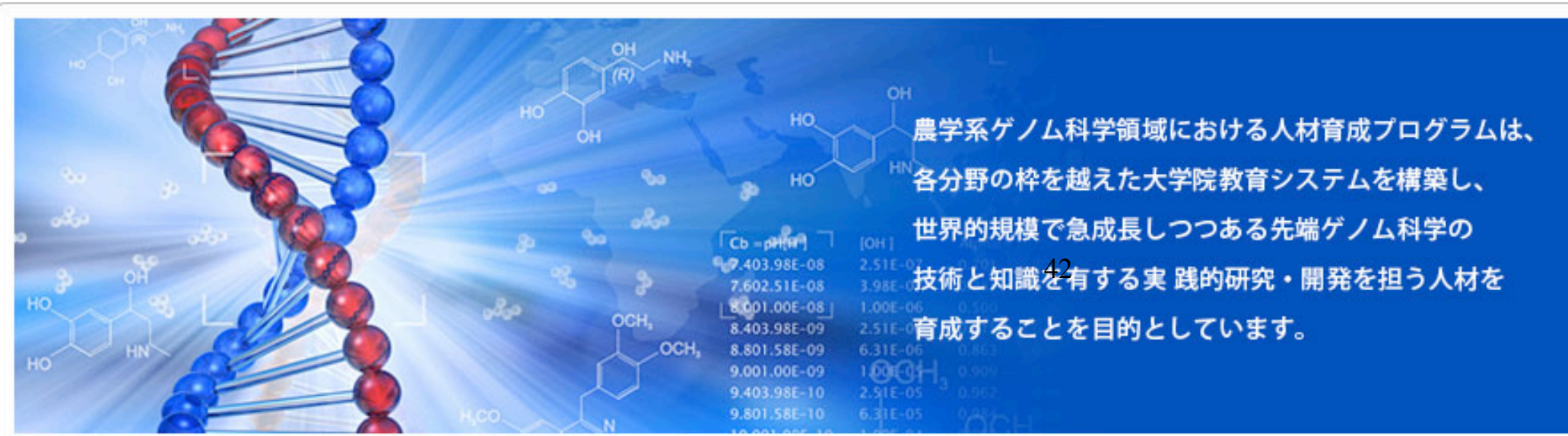
まとめ

１）次世代シーケンサーデータのクオリティは均一でないケースがある。特に、ランが失敗した例では不均一性顕著に出る。

２）クオリティチェックに使用されているソフト（FASTQC）は最初２０万塩基しかチェックしていないので、ランが失敗した場合のクオリティチェックデータの解釈は注意が必要。

→FASTQCの結果　＝　ラン全体のクオリティ
になるとは限らない。

解析例　II
進化系統樹の最適化



目的:

447種類の遺伝子を用いた28種の近縁生物種の分岐図を作成する

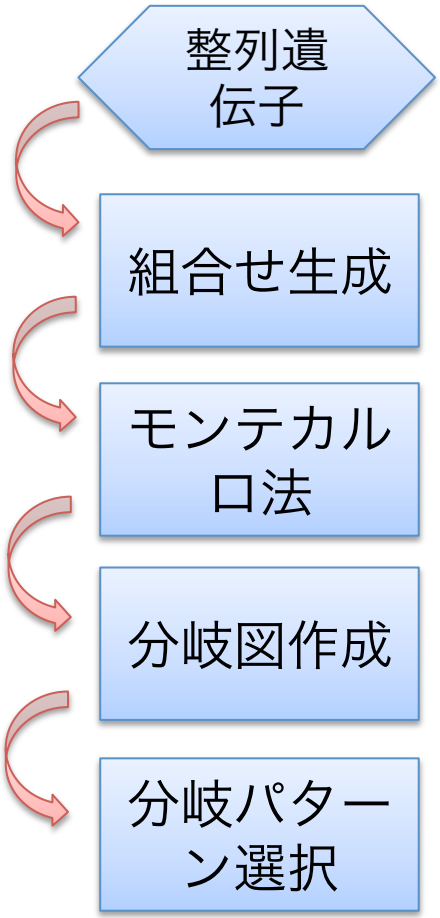
従来、分岐図（進化系統樹）は、ミトコンドリアや16S RNAなど比較的短い配列情報をもとに作成していた。

近年の、ゲノム解析技術の進歩から、ゲノム全体に渡る配列情報を用いて分岐図を作成することが可能となった。

その遺伝子（配列）の組合せはほぼ無限にあり、
どのような配列をもとに分岐図を作成するかは、議論の余地がある。

しかし、この検討には大量の計算が必要である。
今回、モンテカルロ法による無作為抽出と、
ビッグデータ処理による最適化法を確立することを目的とした。

方法:
分岐図作成の最適化検討のためのシナリオ



1、447種類の遺伝子のマルチプルアラインメントデータを使用。

2、447種類の遺伝子の組合せ数を計算。組合せのパターンを生成。
447種類の遺伝子のうち4つまでの遺伝子を除く組合せの数が、1656133808であったので、今回はこれで、最適化法の確立の検討を行なった。447個の遺伝子の総当たりの組合せは、6.109568e+99であり、これを検討することは現実には無理。

3、1656133808の数から、一様乱数を発生。該当する遺伝子の組合せたマルチプルアラインメントを作成(ブートストラップ法)。

4、分岐図を作成し、分岐パターンを抽出。最尤法による推定により分岐図を作成(分岐図作成ソフトRAxMLを使用。)

5、分岐パターンを集計して、最適化分岐図を作成し、
分岐図作成に対し寄与度の高い遺伝子を検出した。

実施結果：

447個の遺伝子から任意の個数の遺伝子を選択する組合せ数の計算 (with S-PLUS/R)

```
> choose(447,1)  # 447個の遺伝子から1遺伝子を選択
[1] 447
> choose(447,1)+choose(447,2) # 2遺伝子を選択
[1] 100128
> choose(447,1)+choose(447,2)+choose(447,3)
[1] 14886143                                  # 3遺伝子を選択
>choose(447,1)+choose(447,2)+choose(447,3)+
choose(447,4)                                 # 4遺伝子を選択
[1] 1656133808 <- 今回はこれで検証(該当する遺伝子を除く)
>choose(447,1)+choose(447,2)+choose(447,3)+.....+
choose(447,447)                               # 447遺伝子を選択
[1] 6.109568e+99
```

実施結果:
組合せ数列を生成するためのPerlのスクリプト

```
#!/usr/local/bin/perl
use strict;
use warnings;
our $number = 447;
our $test_flag = 0;

reflex('', 1, $number) if $test_flag;
for(my $rest = 1; $rest <= $number; $rest++){
        reflex('', 1, $rest);
}
sub reflex{
        my($compu
        if($rest ==
                for(m

                }
                retur
        }else{
                $rest
                my $end = $number - $rest;
                for(my $i = $start; $i <= $end; $i++){
                        reflex("$computed$i,", $i + 1, $rest);
                }
        }
}
```

447個の遺伝子から
4個までを選択する
1656133808個の
組合せ数列を産生

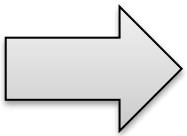
実施結果:
乱数の発生と対応する組合せ数列の選択

1、S-PLUS/Rを用いて一様乱数(runif()を発生)

run1 <- round(runif(100000, min = 1, max = 1656133808)) # 10万個の乱数を発生

2、乱数に対応する組合わせ数列を選択

発生させた乱数	乱数に対応する数列
81571247	5,121,150,372
365913044	27,29,176,229
1023396239	95,160,302,413
644416555	51,199,232,430
691641727	56,126,142,201
1268663549	135,342,359,446
982640498	89,303,405,437
670232037	54,98,149,213
516837044	39,234,259,417
938086069	84,108,135,403



3、447個のうち乱数に対応する遺伝子を除き、残りのすべての遺伝子を連結させた
マルチプルアラインメントファイルを作成

実施結果:
RAxML の実行による分岐図の作成

マルチプルアラインメントデータのfasta 形式ファイルから phylip 形式ファイルへの変換

/usr/local/EMBOSS-6.4.0/bin/seqret fasta::1.fasta phylip::1r.phy

分岐図作成ソフトRAxMLの実行

/usr/local/packages/RAxML/RAxML-7.2.8-ALPHA/raxmlHPC-PTHREADS -f a -x 12345
-p 12345 -# 20 -m GTRGAMMA -s 1.phy -n 1phy.out -T 16

RAxML の出力結果から以下のコマンドにより、数値や記号などを取り去り、
分岐図の簡素化パターンに変換

less RAxML_bestTree.200.out | tr -d [0-9] | tr -d ":" | tr -d "." | tr -d ";"
> RAxML_bestTree.200SUMMARY.out

実施結果：
RAxML の出力結果を分岐図パターンに変換

RAxML の出力結果から以下のコマンドにより、数値や記号などを取り去り、分岐図パターンに変換

less RAxML_bestTree.out | tr -d [0-9] | tr -d ":" | tr -d "." | tr -d ";" > RAxML_bestTreeSUMMARY.out

==> RAxML_bestTree.out <== # RAxML の出力結果

(((x:0.005314048137224604 11,b:0.005826809298017833 14):0.025971266250438409 39,(m:
0.036453249559941544 59,((i:0.021301221600792997 34,g:0.031447907657455420 60):0.004241512818670545 65
,u:0.029486867696565367 82):0.020403600469400217 52):0.015217749948996110 03):0.008408062190607702 37,
((((z:0.343447671899541562 28,(t:0.142626139545608793 26,l:0.096110243065704065 17):
0.124827271887547816 55):0.177386793894591998 64,(((q:0.092209036363336674 41,c:0.078497404029646056 23
0.021622137100446872 65,(((((e:0.022096220188703392 94,k:0.028932831190996814 72):0.008971545350474785 5
p:0.085261674267942760 83):0.013931939494733546 62,(f:0.013183566304447993 59,(n:0.019524905232023027 91
&:0.016931273087794251 19):0.008136049288154000 03):0.078179888554669924 04):0.022634323150847322 08,(r
0.045904457675196408 41,h:0.046373153889997971 44):0.038653186796641453 29):0.007891913738147052 56,(v:
0.032846119179964099 19,d:0.022502105683185325 70):0.179230327984116921 68):0.010010494870981927 20):
0.103257341897420487 63,(y:0.200862493548863818 57,(j:0.183217529753788327 40,#:0.209679853903260327 02)
0.12992858329809614 526):0.018805238453222176 96):0.038574145917489026 38):0.066201010319012223 99,(o:
0.038546485200935606 82,s:0.035527049274526989 46):0.127068551707286592 21):0.056308300369021499 49,w:
0.139336296263945474 96):0.071577804094613770 03,a:0.037820217201590664 02):0.0;

==> RAxML_bestTreeSUMMARY.out <== # 変換された簡素化分岐図パターン

(((x,b),(m,((i,g),u))),((((z,(t,l)),(((q,c),(((((e,k),p),(f,(n,&))),(r,h)),(v,d))),(y,(j,#)))),(o,s)),w),a)

実施結果:
分岐図パターンの集計

分岐図パターンの頻度を計数することで数値化

以下のように、先行20回の試行解析のうち
19回: (((x,b),(m,((i,g),u))),((((z,(t,l)),(((q,c),(((((e,k),p),(f,(n,&))),(r,h)),(v,d))),(y,(j,#)))),(o,s)),w),a)

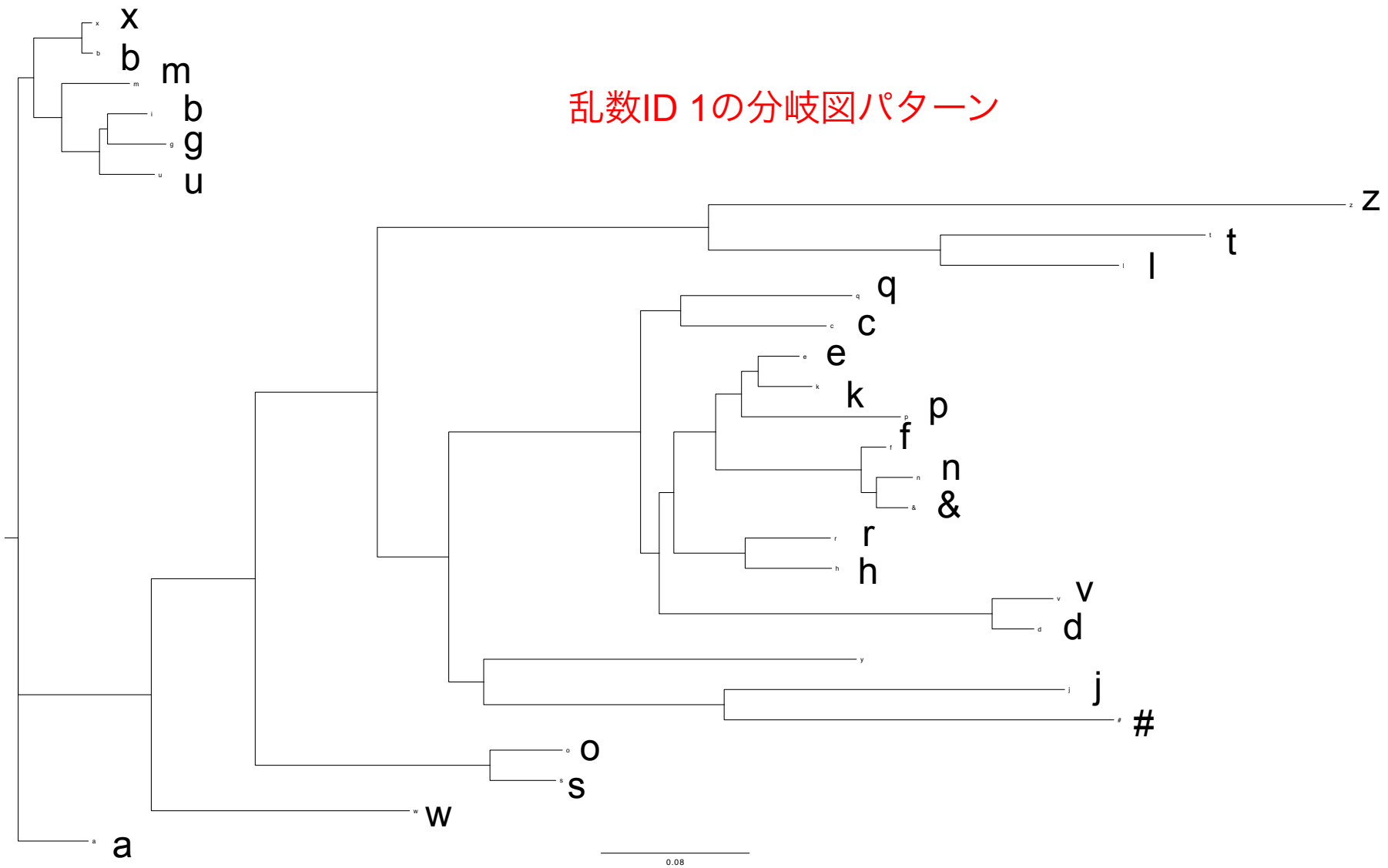
1回: (((x,b),(m,((i,g),u))),((((z,(t,l)),(((q,c),((v,d),((((e,k),p),(f,(n,&))),(r,h)))),(y,(j,#)))),(o,s)),w),a)

となり最適化分岐図が求められた。（ただし、この程度では、検討に足りない、、。）
また、189、203、337、393が分岐図作成に強く寄与していることが推定された。

乱数ID	遺伝子1	遺伝子2	遺伝子3	遺伝子4	Process Name	Tree Pattern
1	5	121	150	372	xaaN1	(((x,b),(m,((i,g),u))),((((z,(t,l)),(((q,c),(((((e,k),p),(f,(n,&))),(r,h)),(v,d))),(y,(j,#)))),(o,s)),w),a)
2	27	29	176	229	xaaN2	(((x,b),(m,((i,g),u))),((((z,(t,l)),(((q,c),(((((e,k),p),(f,(n,&))),(r,h)),(v,d))),(y,(j,#)))),(o,s)),w),a)
3	95	160	302	413	xaaN3	(((x,b),(m,((i,g),u))),((((z,(t,l)),(((q,c),(((((e,k),p),(f,(n,&))),(r,h)),(v,d))),(y,(j,#)))),(o,s)),w),a)
4	51	199	232	430	xaaN4	(((x,b),(m,((i,g),u))),((((z,(t,l)),(((q,c),(((((e,k),p),(f,(n,&))),(r,h)),(v,d))),(y,(j,#)))),(o,s)),w),a)
5	56	126	142	201	xaaN5	(((x,b),(m,((i,g),u))),((((z,(t,l)),(((q,c),(((((e,k),p),(f,(n,&))),(r,h)),(v,d))),(y,(j,#)))),(o,s)),w),a)
6	135	342	359	446	xaaN6	(((x,b),(m,((i,g),u))),((((z,(t,l)),(((q,c),(((((e,k),p),(f,(n,&))),(r,h)),(v,d))),(y,(j,#)))),(o,s)),w),a)
7	89	303	405	437	xaaN7	(((x,b),(m,((i,g),u))),((((z,(t,l)),(((q,c),(((((e,k),p),(f,(n,&))),(r,h)),(v,d))),(y,(j,#)))),(o,s)),w),a)
8	54	98	149	213	xaaN8	(((x,b),(m,((i,g),u))),((((z,(t,l)),(((q,c),(((((e,k),p),(f,(n,&))),(r,h)),(v,d))),(y,(j,#)))),(o,s)),w),a)
9	39	234	259	417	xaaN9	(((x,b),(m,((i,g),u))),((((z,(t,l)),(((q,c),(((((e,k),p),(f,(n,&))),(r,h)),(v,d))),(y,(j,#)))),(o,s)),w),a)
10	84	108	135	403	xaaN10	(((x,b),(m,((i,g),u))),((((z,(t,l)),(((q,c),(((((e,k),p),(f,(n,&))),(r,h)),(v,d))),(y,(j,#)))),(o,s)),w),a)
11	107	277	308	420	xaaN11	(((x,b),(m,((i,g),u))),((((z,(t,l)),(((q,c),(((((e,k),p),(f,(n,&))),(r,h)),(v,d))),(y,(j,#)))),(o,s)),w),a)
12	132	379	399	432	xaaN12	(((x,b),(m,((i,g),u))),((((z,(t,l)),(((q,c),(((((e,k),p),(f,(n,&))),(r,h)),(v,d))),(y,(j,#)))),(o,s)),w),a)
13	20	223	367	414	xaaN13	(((x,b),(m,((i,g),u))),((((z,(t,l)),(((q,c),(((((e,k),p),(f,(n,&))),(r,h)),(v,d))),(y,(j,#)))),(o,s)),w),a)
14	23	96	240	421	xaaN14	(((x,b),(m,((i,g),u))),((((z,(t,l)),(((q,c),(((((e,k),p),(f,(n,&))),(r,h)),(v,d))),(y,(j,#)))),(o,s)),w),a)
17	20	243	277	295	xaaN17	(((x,b),(m,((i,g),u))),((((z,(t,l)),(((q,c),(((((e,k),p),(f,(n,&))),(r,h)),(v,d))),(y,(j,#)))),(o,s)),w),a)
18	6	219	248	317	xaaN18	(((x,b),(m,((i,g),u))),((((z,(t,l)),(((q,c),(((((e,k),p),(f,(n,&))),(r,h)),(v,d))),(y,(j,#)))),(o,s)),w),a)
20	56	181	237	313	xaaN20	(((x,b),(m,((i,g),u))),((((z,(t,l)),(((q,c),(((((e,k),p),(f,(n,&))),(r,h)),(v,d))),(y,(j,#)))),(o,s)),w),a)
21	220	247	327	390	xaaN21	(((x,b),(m,((i,g),u))),((((z,(t,l)),(((q,c),(((((e,k),p),(f,(n,&))),(r,h)),(v,d))),(y,(j,#)))),(o,s)),w),a)
22	189	203	337	393	xaaN22	(((x,b),(m,((i,g),u))),((((z,(t,l)),(((q,c),((v,d),((((e,k),p),(f,(n,&))),(r,h)))),(y,(j,#)))),(o,s)),w),a)
23	149	251	300	316	xaaN23	(((x,b),(m,((i,g),u))),((((z,(t,l)),(((q,c),(((((e,k),p),(f,(n,&))),(r,h)),(v,d))),(y,(j,#)))),(o,s)),w),a)

実施結果:
最適化された分岐図パターン

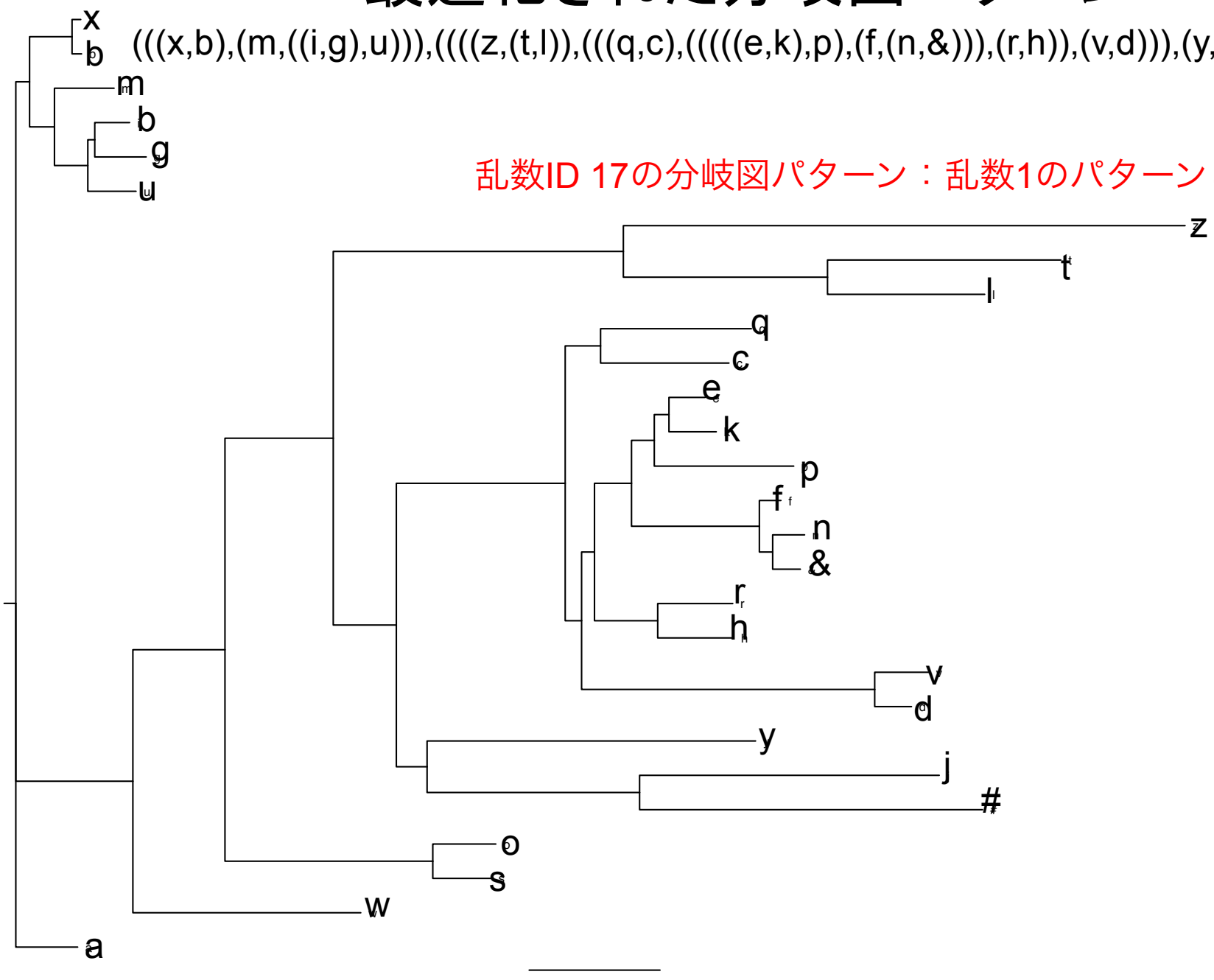
(((x,b),(m,((i,g),u))),((((z,(t,l)),(((q,c),(((((e,k),p),(f,(n,&))),(r,h)),(v,d))),(y,(j,#)))),(o,s)),w),a)



実施結果:
最適化された分岐図パターン

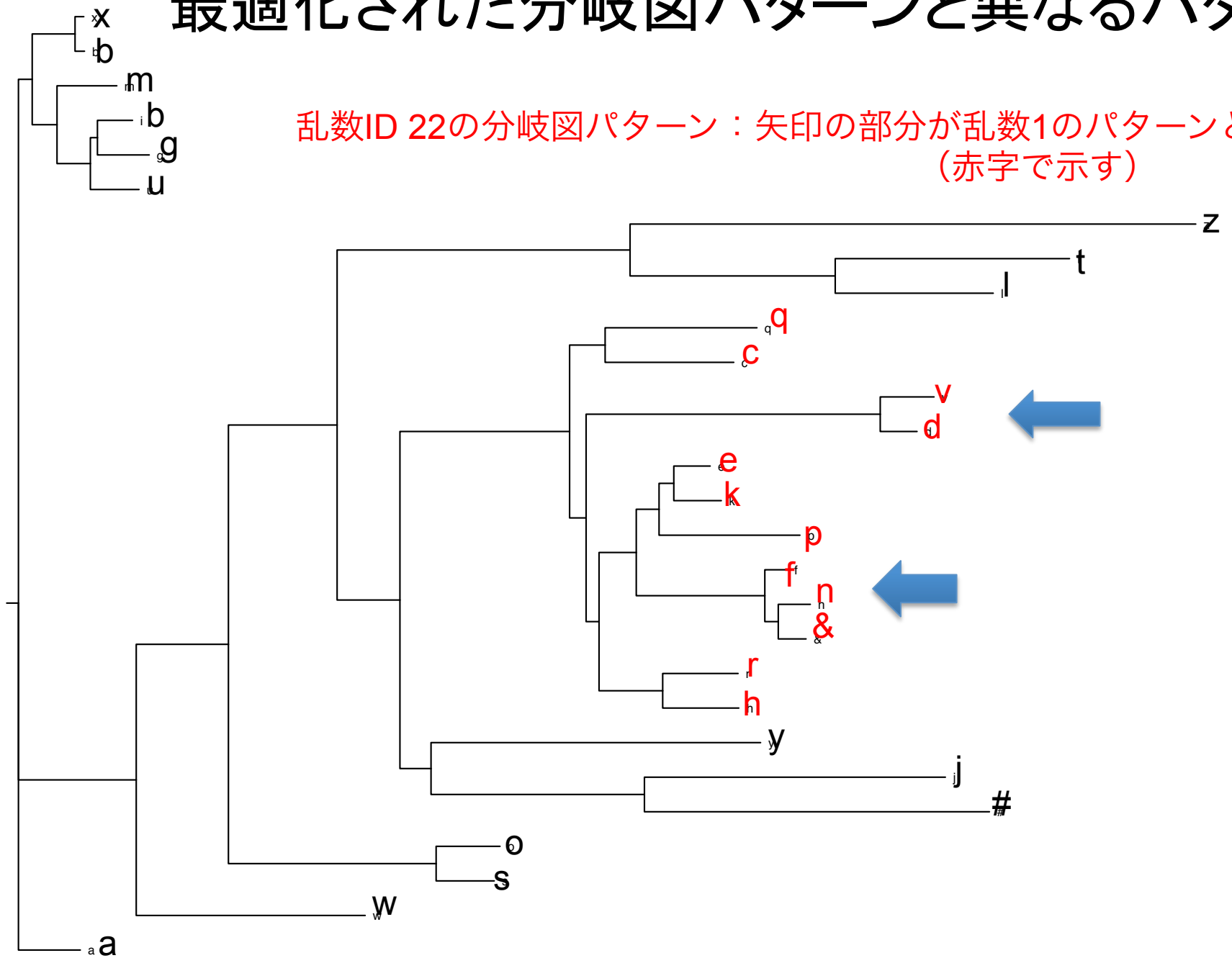
(((x,b),(m,((i,g),u))),((((z,(t,l)),(((q,c),(((((e,k),p),(f,(n,&))),(r,h)),(v,d))),(y,(j,#)))),(o,s)),w),a);

乱数ID 17の分岐図パターン：乱数1のパターンと類似



実施結果：
最適化された分岐図パターンと異なるパターン

乱数ID 22の分岐図パターン：矢印の部分が乱数1のパターンと異なる
（赤字で示す）



(((x,b),(m,((i,g),u))),((((z,(t,l)),(((q,c),((v,d),(((e,k),p),(f,(n,&)))),(r,h)))),(y,(j,#)))),(o,s)),w),a)

まとめ:

１、モンテカルロ法による無作為抽出法による分岐図の最適化法を確立した。
しかし、現状の解析は十分でない。理論上考えられるすべての組合せを計算し、
最適化するというのは、まだ困難。さらに工夫が必要
（分散処理によるスケールアウトを含む）。

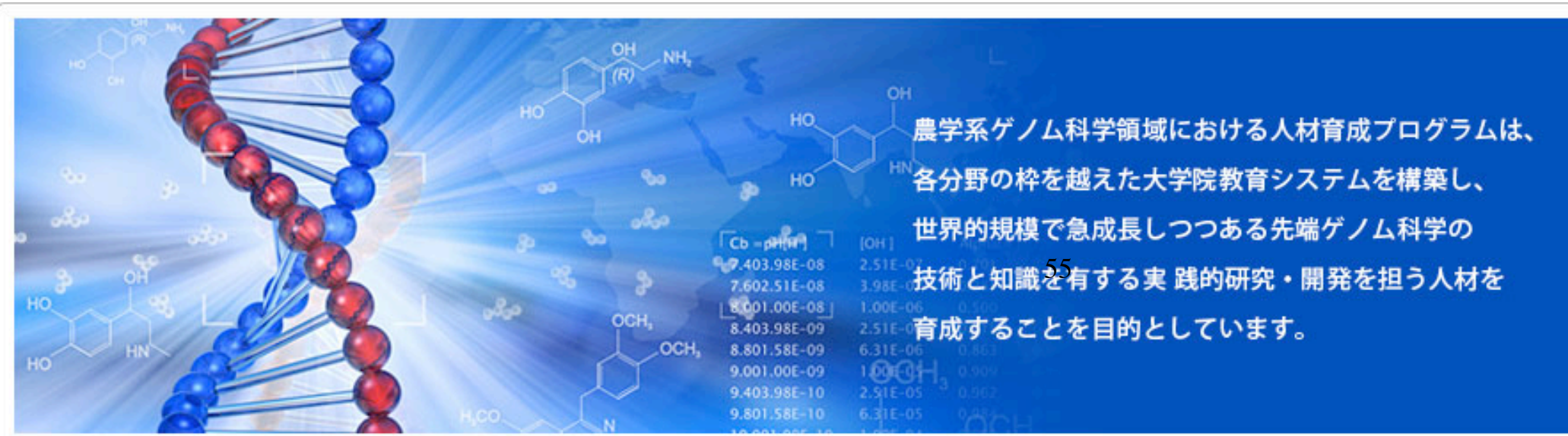
２、その実施には、大規模なコンピュータリソースを必要とし、
気軽にできるとは言いがたい。

３、今後、ゲノム科学のデータ産生量の増加に伴い、同様の大量の計算機リソース
を必要とする場面が増えてくることが予想される。

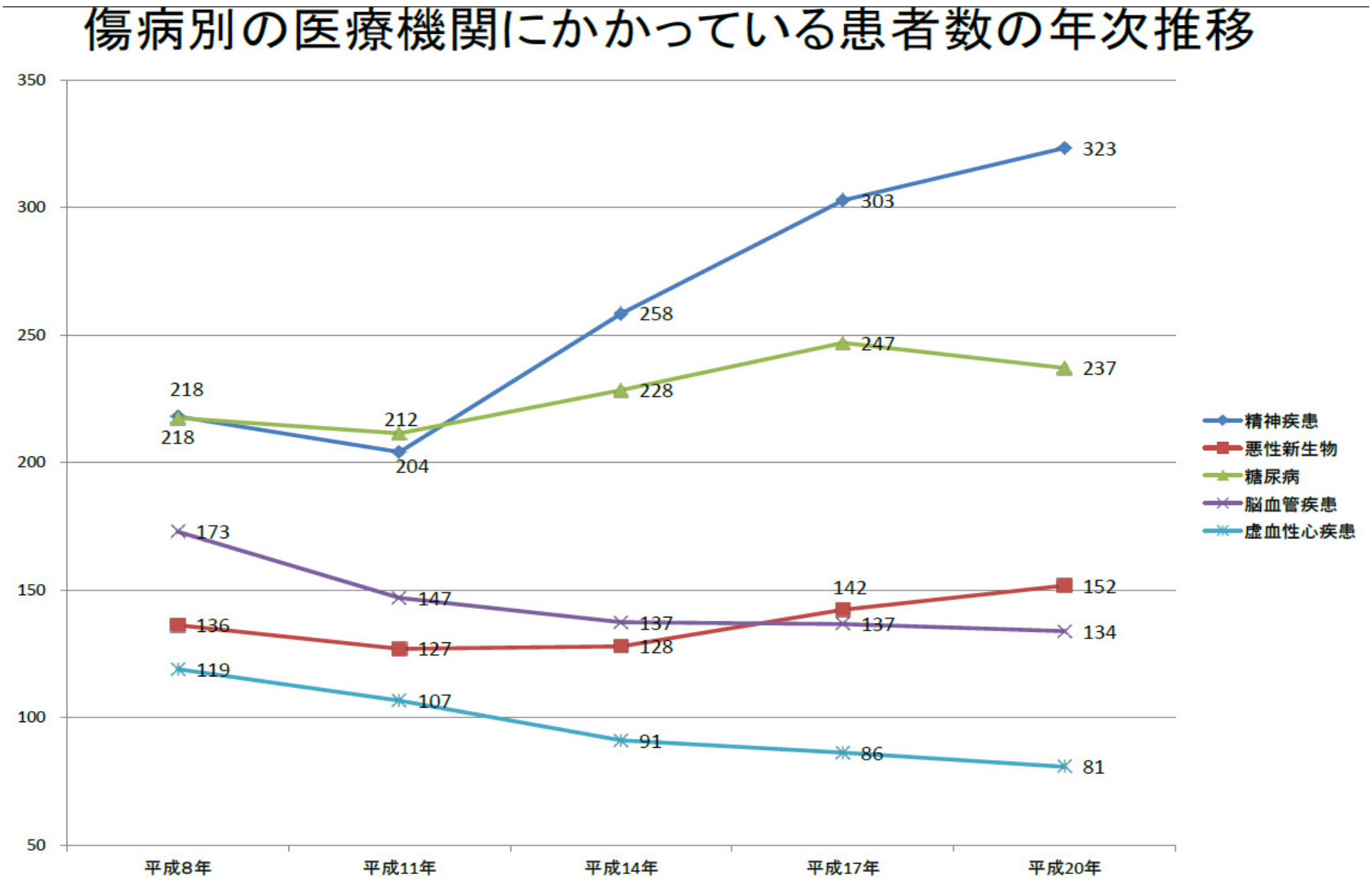
Big Iron や**クラウド**などを導入したビッグデータ処理環境の需要の高まりが
予想される。

¥

解析例　III
精神神経系疾患診断系の確立

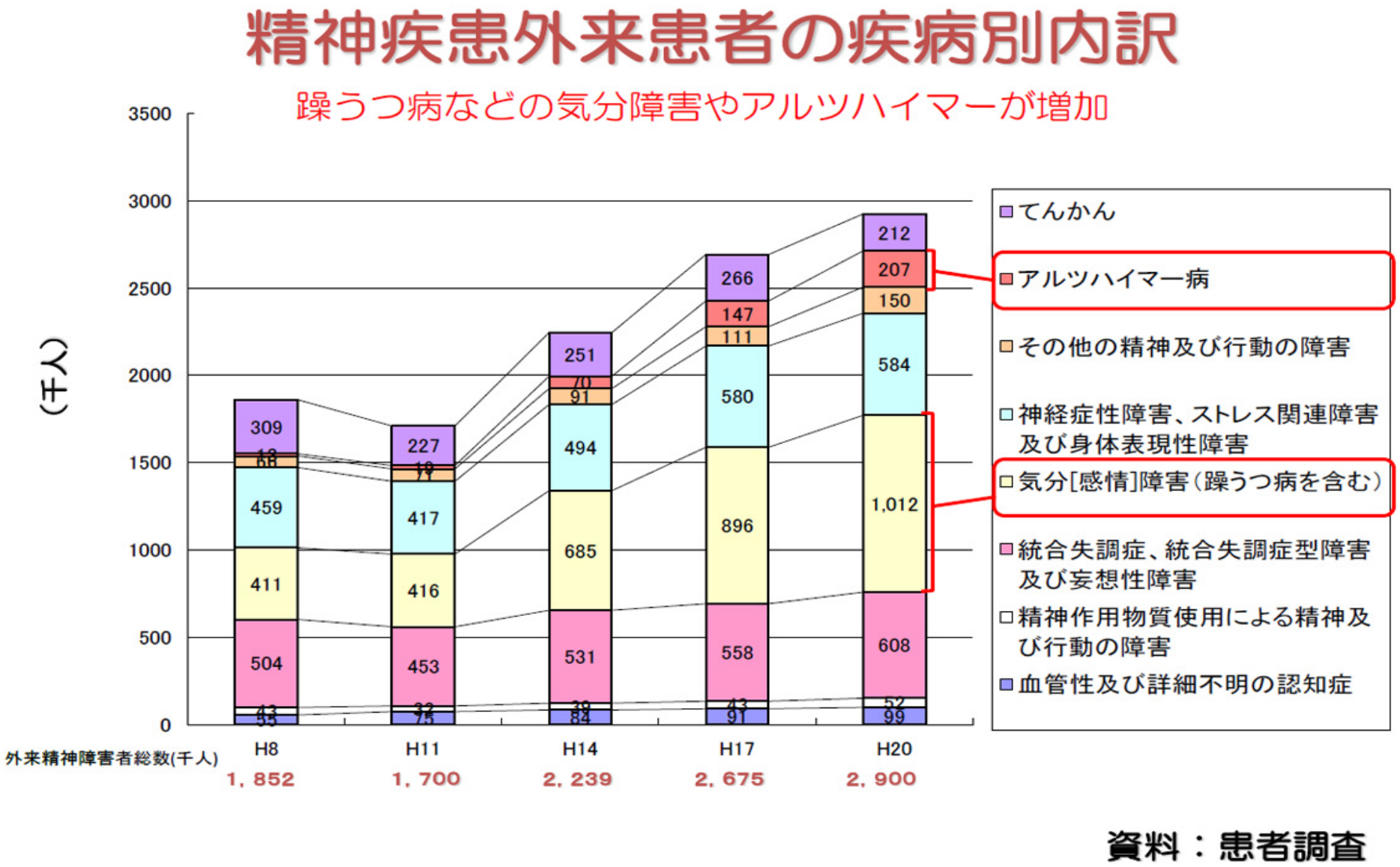


同省が把握する2009年時点のデータによると、がん患者数およそ152万人、糖尿病患者数およそ237万人に対し、精神疾患数は既に323万人となっており、最も多いとされるがんの2倍以上にも及んでいることが明らかになった（http://cl-co.com/headline/news/790/より）。



http://www.mhlw.go.jp/stf/shingi/2r9852000001hx9n.htmlより引用

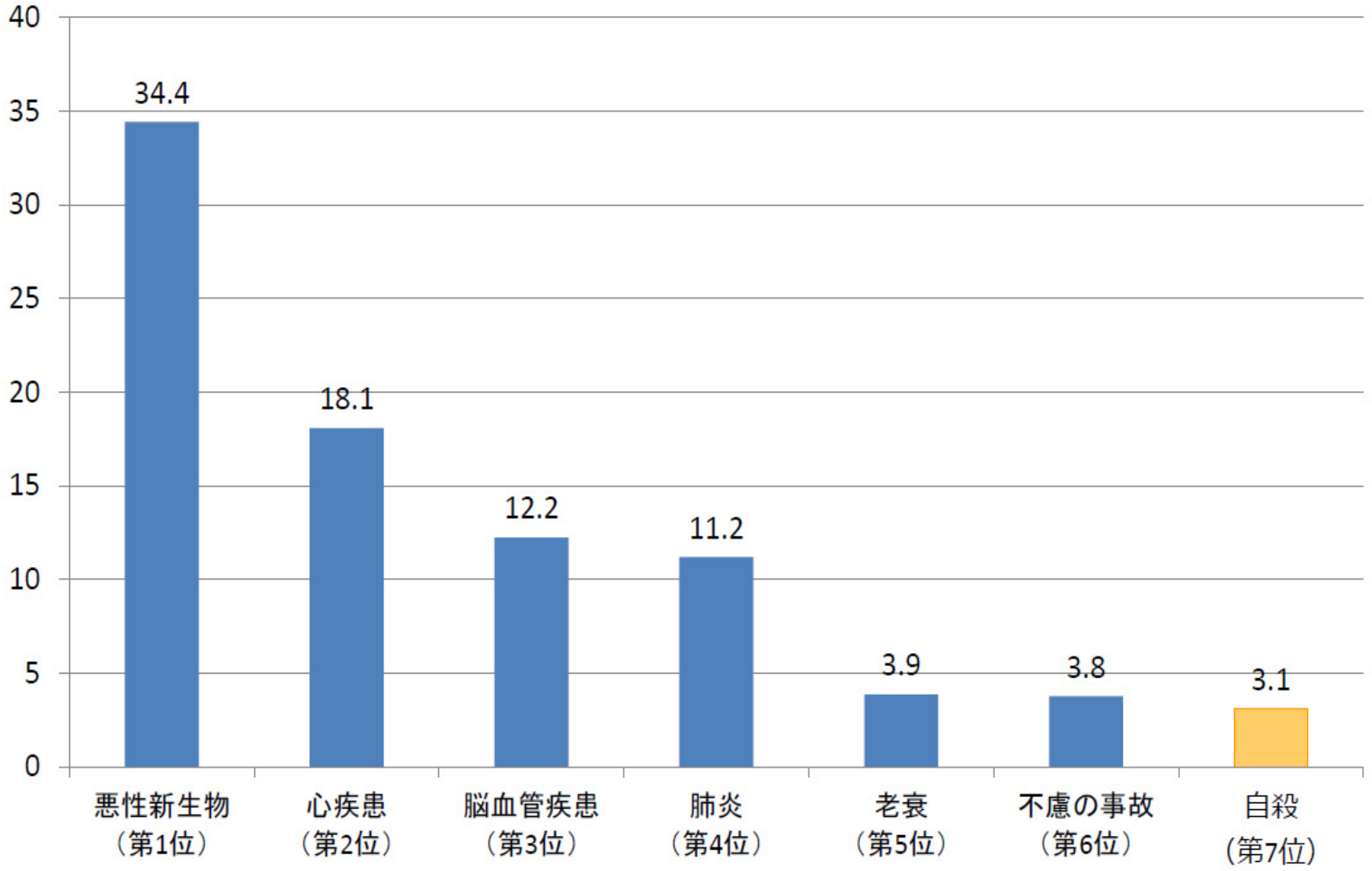
医療機関にかかっている患者数の年次推移を疾病別に示したグラフによると、
がん患者数がほぼ横ばいなのに対し、精神疾患患者数はここ10年間で1.5倍に急増。



資料：患者調査

http://www.mhlw.go.jp/stf/shingi/2r9852000001hx9n.htmlより引用

死因順位別の死亡数



※単位：万人
※出典：平成21年人口動態統計

http://www.mhlw.go.jp/stf/shingi/2r9852000001hx9n.html より引用

精神神経系疾患の特徴

- 近年の社会情勢、国内事情により、急激に患者数が増加。罹患数でガンの2倍以上。

- 2011年、厚生労働省の指定する5大疾患に追加。
- 2013年度から医療計画に反映。

- 問診による医師の経験的診断に基づき、効果的な客観的診断法が存在しない。
- いわゆるアンメットメディカルニーズである。

精神神経系疾患系の構築

精神神経系疾患を対象とする。

臨床血液検体を用いて、網羅的ゲノム解析を行う。

PCRおよびマイクロアレイで予備的解析を行う。

次世代シーケンサーによる検査項目の探索。
（遺伝子発現や遺伝子修飾など）

感度と特異度

医学臨床の検査キットなどの性能を評価する指標に、
感度（sensitivity）と特異度（specificity）というものがある。

		真の状態 (生検などの詳細検査の結果で決定)		
		陽性	陰性	
検査結果	陽性	真陽性	偽陽性 (第1種の過誤)	陽性適中率 = 真陽性の数 / 検査陽性の数
	陰性	偽陰性 (第2種の過誤)	真陰性	陰性適中率 = 真陰性の数 / 検査陰性の数
		感度 = 真陽性の数 / 真陽性+偽陰性	特異度 = 真陰性の数 / 偽陽性+真陰性	

Wikipedia より引用

方法：予備的解析

臨床血液検体（患者、健常者）よりDNA、RNAを採取。
網羅的ゲノム解析を行なう。

測定値を、統計的有意差検定などで検定し、上位の検査項目をリストアップ。

多変量解析、クラスター解析、機械学習、ベイズ統計などでの分別法で分別。
判別式、統計モデルを作成。

感度、特異度を求めて分別力を評価する。

患者　vs　健常者
比較

（少数の検査項目の組み合わせでもある程度の感度、
特異度で完全な診断が可能であることを確認）

詳細データは後日公開

線形モデルによるモデル式の作成
モデル式なども後日公開

今後の展開

多検体、多項目で総合的に検討し、診断法を確立。

〇高次元データによる解析アプローチ

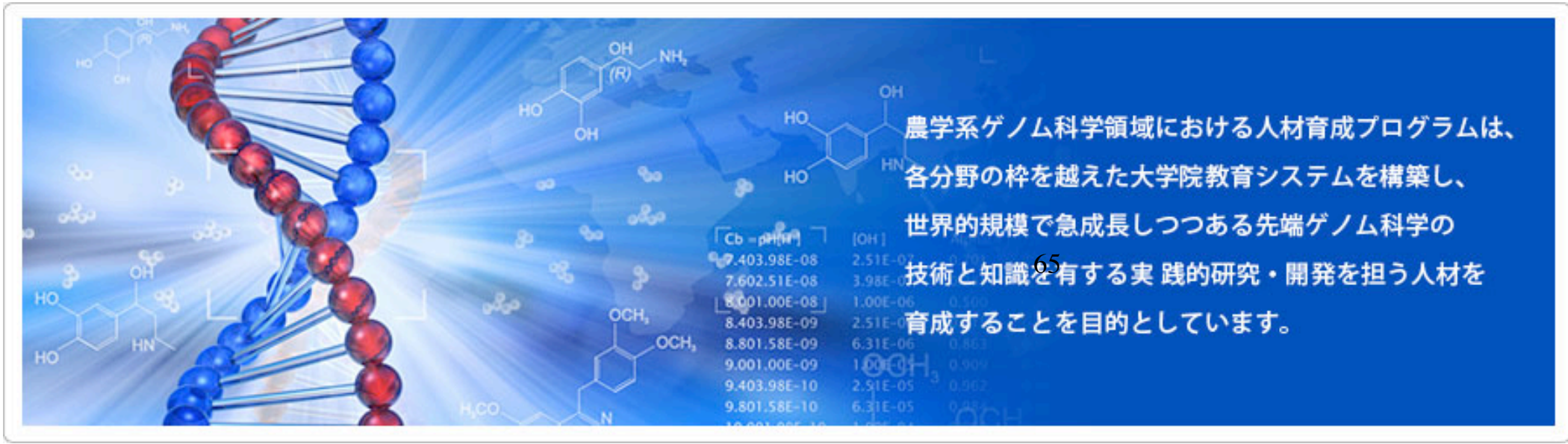
〇説明変数の絞り込み（スパースモデリング）

2つの方法で、診断系を確立する。

次世代シーケンサーだと7.5テラバイト。
アマゾンクラウド＋Hadoop
各種分別法で、感度、特異度の検討。

事例　IV
情報処理学会 人材育成

ビッグデータ活用実務フォーラム



情報処理学会
ビッグデータ活用実務フォーラム

IT 勉強会の活動をモデルとして、
ビッグデータ活用の現場で勤務するデータサイエンティストや、
データサイエンティストを目指す学生や若手技術者を対象に、
ビッグデータの現場での活用に関する情報交換、情報共有、
情報発信の場を提供し、もって若手データサイエンティストの
人材育成に寄与することを目的する。

セミナー、勉強会など

１、ビッグデータ現場の会
２、オープンソースカンファレンス
３、日本技術士会CPD中央講座
４、情報処理学会ソフトウェアジャパン2014
５、その他、実務講習会

雑誌特集号など

情報処理学会誌など
「ビッグデータ活用実務特集号」企画

今後

データサイエンティスト
人材育成に貢献したい

共同研究者

東京農工大学
佐藤暁、古崎利紀、有江力

茨城大学
松田朋子、後藤哲雄

徳島大学
沼田周助、木下誠、伊賀淳一、渡部真也、大森哲郎