NGS analyzer:
次世代シークエンス解析プログラム

独立行政法人 理化学研究所
情報基盤センター
HPCI計算生命科学推進プログラム
須永　泰弘

NGS analyzerとは？

・次世代シークエンサー(NGS)からの塩基配列データを用いて、マッピング、PCRの除去、SNPタイピング、欠失挿入の検出を行う。

・一連の作業はパイプライン化してある。

・「京」コンピュータなどの並列計算機で高速に行うことが可能

開発者：

理化学研究所統合生命医科学研究センター
角田　達彦
藤本　明洋

本日の講義内容

１．次世代シークエンス（NGS）の概要

２．NGS analyzerの概要

３．必要な計算環境とデータファイル

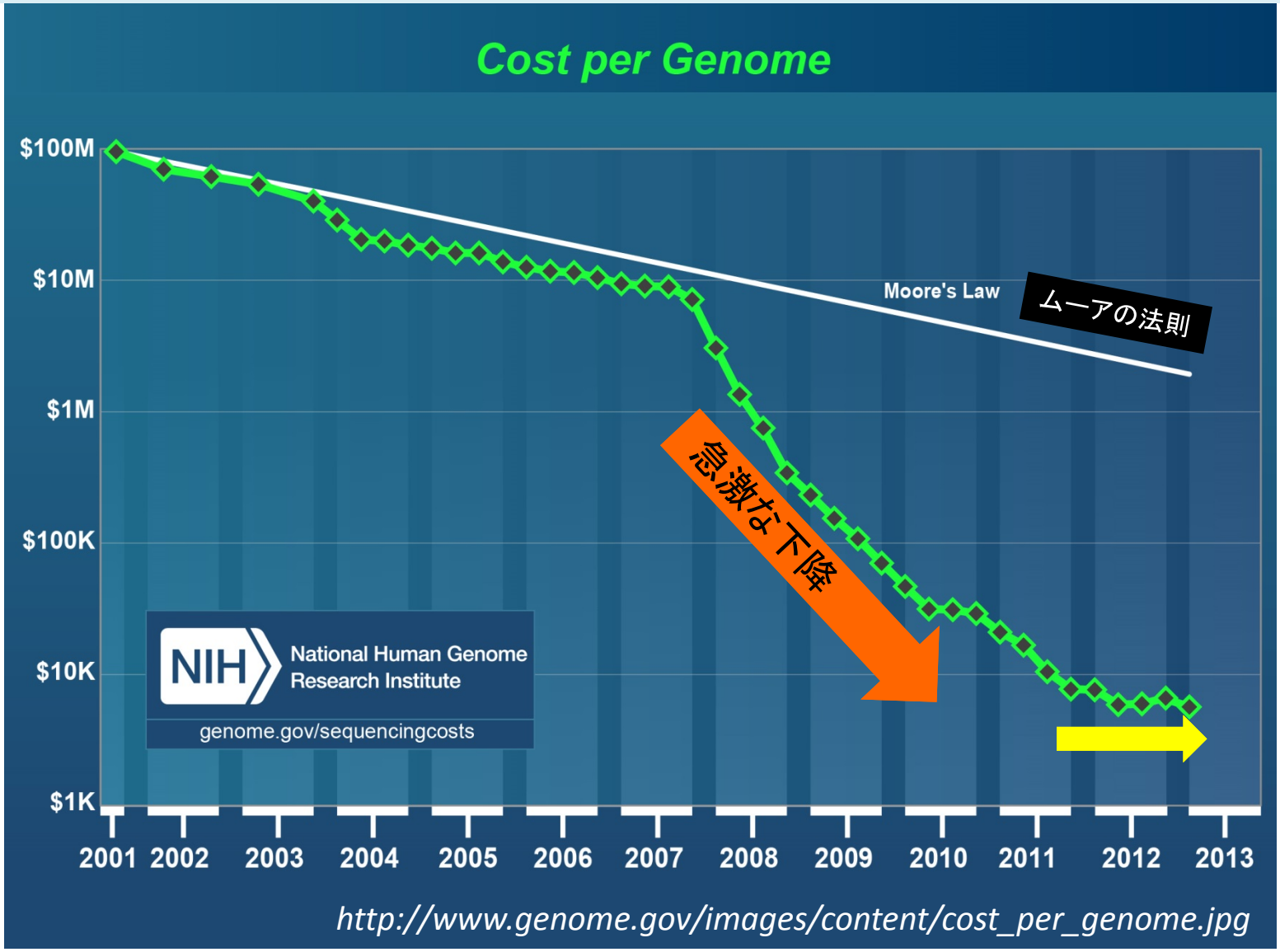
４．解析例の紹介と速度比較

５．解析の流れ（コマンドの紹介）

本日の講義内容

1. 次世代シークエンス（NGS）の概要
2. NGS analyzerの概要
3. 必要な計算環境とデータファイル
4. 解析例の紹介と速度比較
5. 解析の流れ（コマンドの紹介）

個人ゲノム決定のコスト

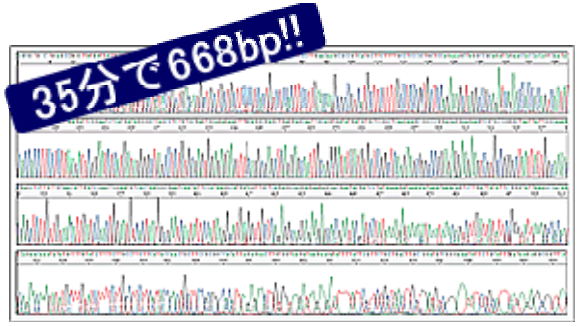


10億円

100万円

次世代シークエンサー(ハイスループット型)

参考:
第1世代自動シークエンサー
3130xl　(ABI)



GSFLX+(ロシュ)

0.7Gb/23時間

リード長:700bp

Hiseq2500(イルミナ)

600Gb/11日

リード長:2×100bp

5500xlw(ライフテクイノロジーズ)

270Gb/14日

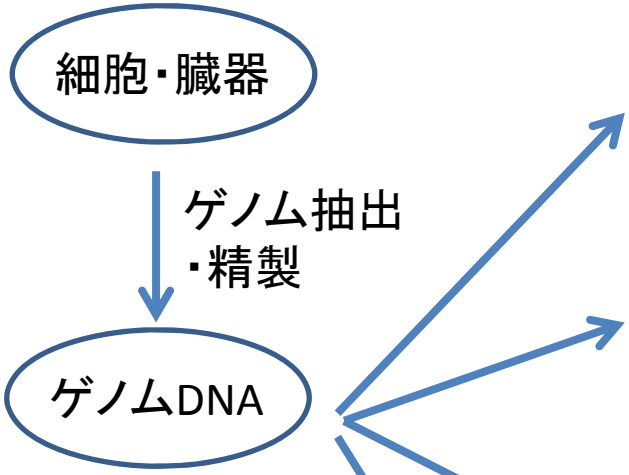
リード長:2×60bp

NGSの速度向上の歴史
1回の測定での比較

Platform	機種名	発売年	並列数	リード長(bp)	取得塩基数(GBp)
ABI Sanger	3730xl	第1世代	96	800	0.00000768
454	GS20	2005	200,000	100	0.02
Illumina	GA	2006	28,000,000	25	0.7
454	GS FLX	2007	400,000	250	1
Illumina	GA	2008	28,000,000	35	1
454	GS FLX Titanium	2009	1,000,000	500	0.45
SOLiD	1	2007	40,000,000	25	1
Illumina	Hiseq	2011	2,000,000,000	100	200
SOLiD	5500xl	2011	3,000,000,000	60	180
454	GS FLX+	2011	1,000,000	700	0.5
IonTorrent	Proton	2012	5,000,000	200	10
Illumina	Hiseq2500	2012	3,000,000,000	100	600
PacBio	RSC2XL	2012	36,000	4,300	0.155




ゲノム塩基配列決定の歴史



従来法1：
Maxam-Gilbert法

DNAを化学分解して配列を決定
PCR発明以前の主流

従来法2-1：
Sanger法

DNAを合成して配列を決定
蛍光物質を使用することで自動化
に成功。方法論は今でも主流

従来法2-2：
ショットガン法

DNAを断片化してシークエンス、
配列の重なりを計算機内で合成し、
長い断片をえる。ヒトゲノム決定の
高速化に貢献

NGS：
次世代シークエンス法

画像処理の技術を利用し、高
密度でショットガン法を行う。

NGSの代表的なアプリケーション

①新規データゲノム配列決定：De novo シークエンス

ゲノム配列が明らかでない種のゲノムを決定する。

②ゲノム再配列決定：リシークエンス

ゲノム配列がわかっている種の遺伝子多型を明らかにする。

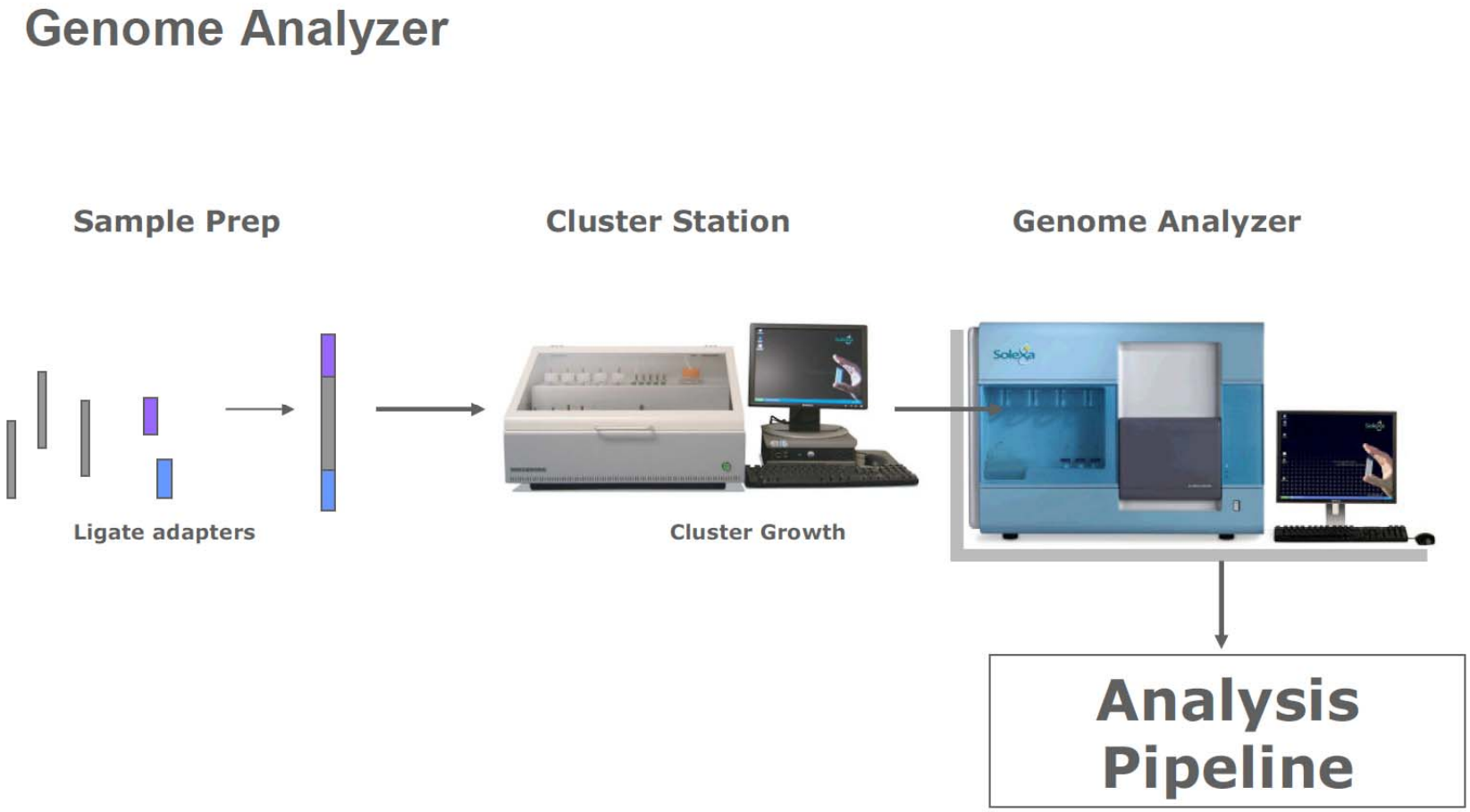
③RNA-seq

RNAの発現解析をする。 GeneChip, qRTP-CR, RNAブロット

④Chip-seq

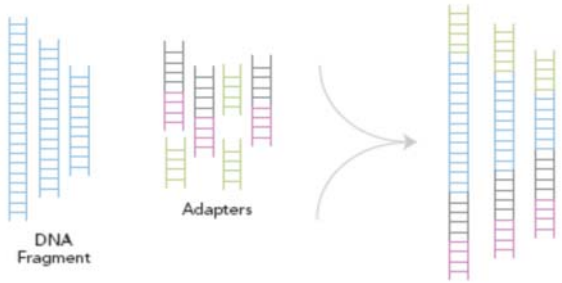
転写因子などのDNAと結合する配列を決定する。

Massively parallel sequencing technology

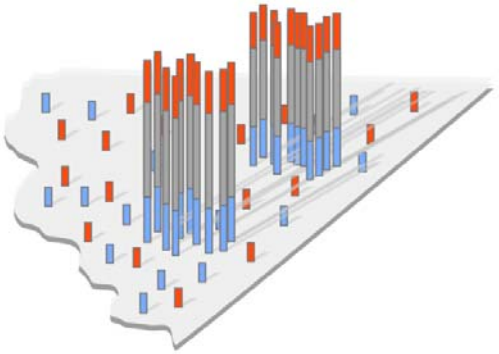


Sample preparation and sequencing

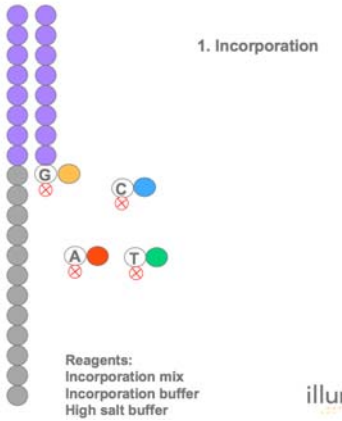
DNAの断片化とアダプター配列の付加



flowcellへの結合と増幅

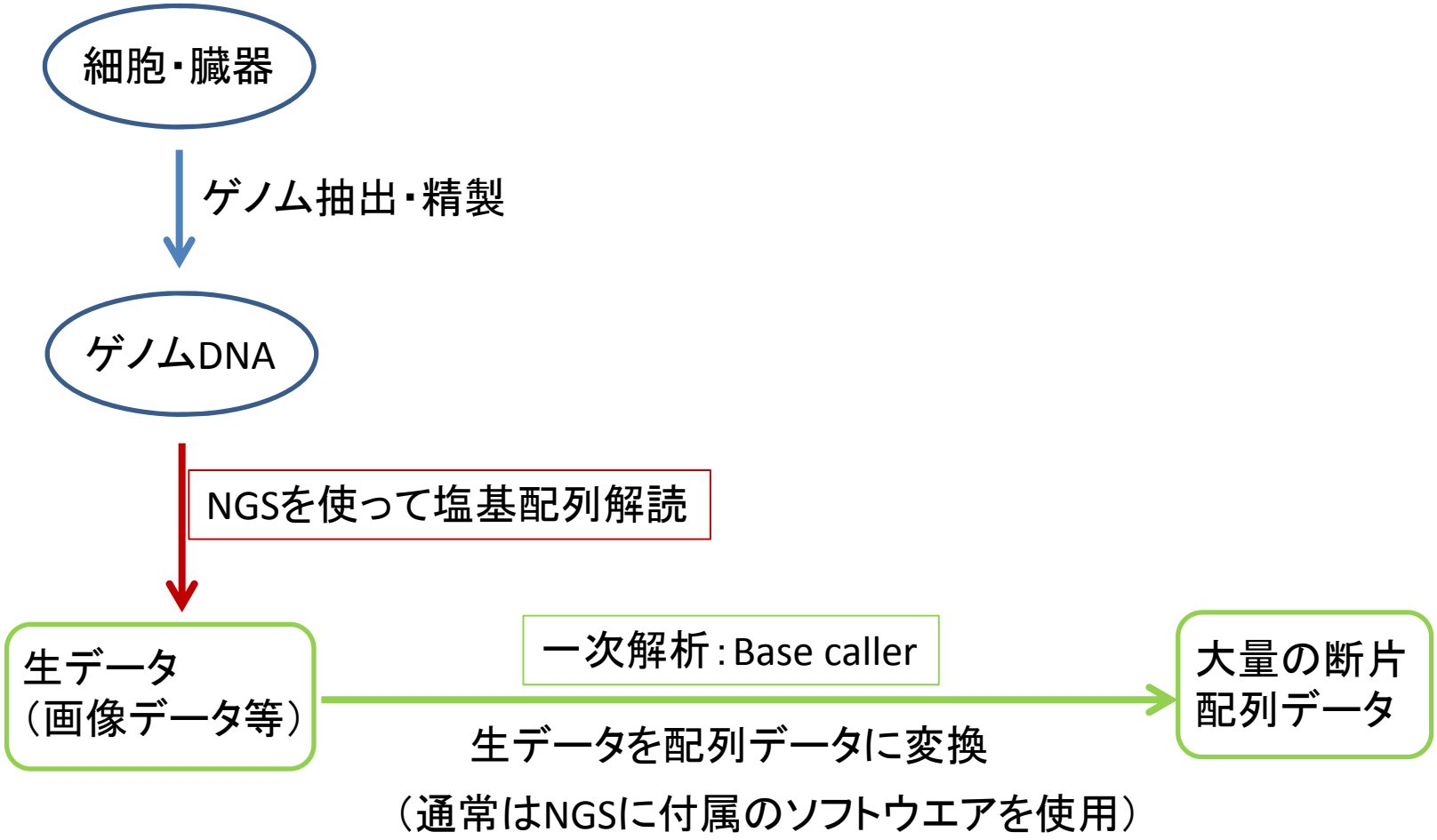


シーケンス反応



蛍光の取り込みと塩基配列決定

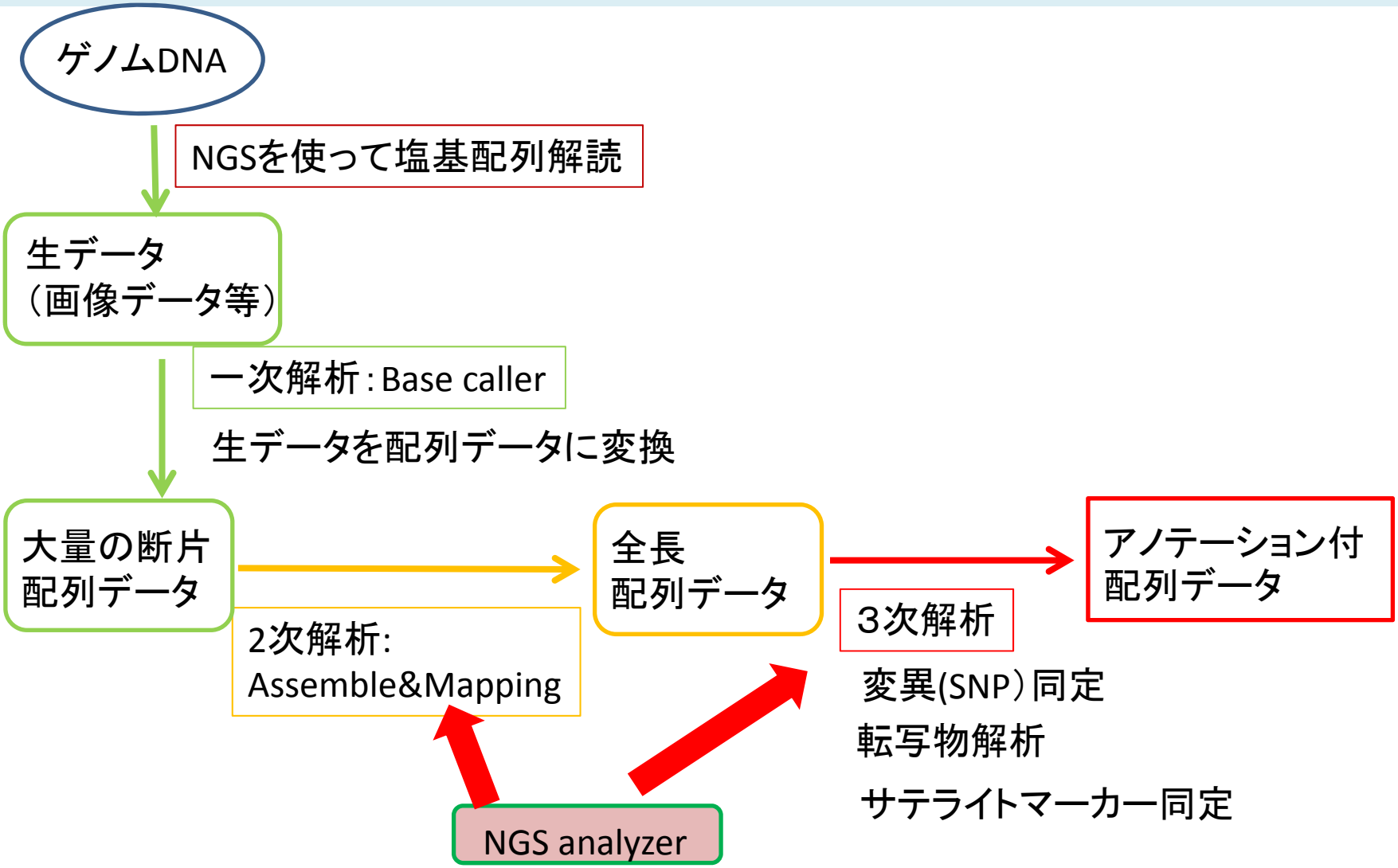
NGSのフローチャート



本日の講義内容

1. 次世代シークエンス（NGS）の概要
2. NGS analyzerの概要
3. 必要な計算環境とデータファイル
4. 解析例の紹介と速度比較
5. 解析の流れ（コマンドの紹介）

NGSのフローチャート

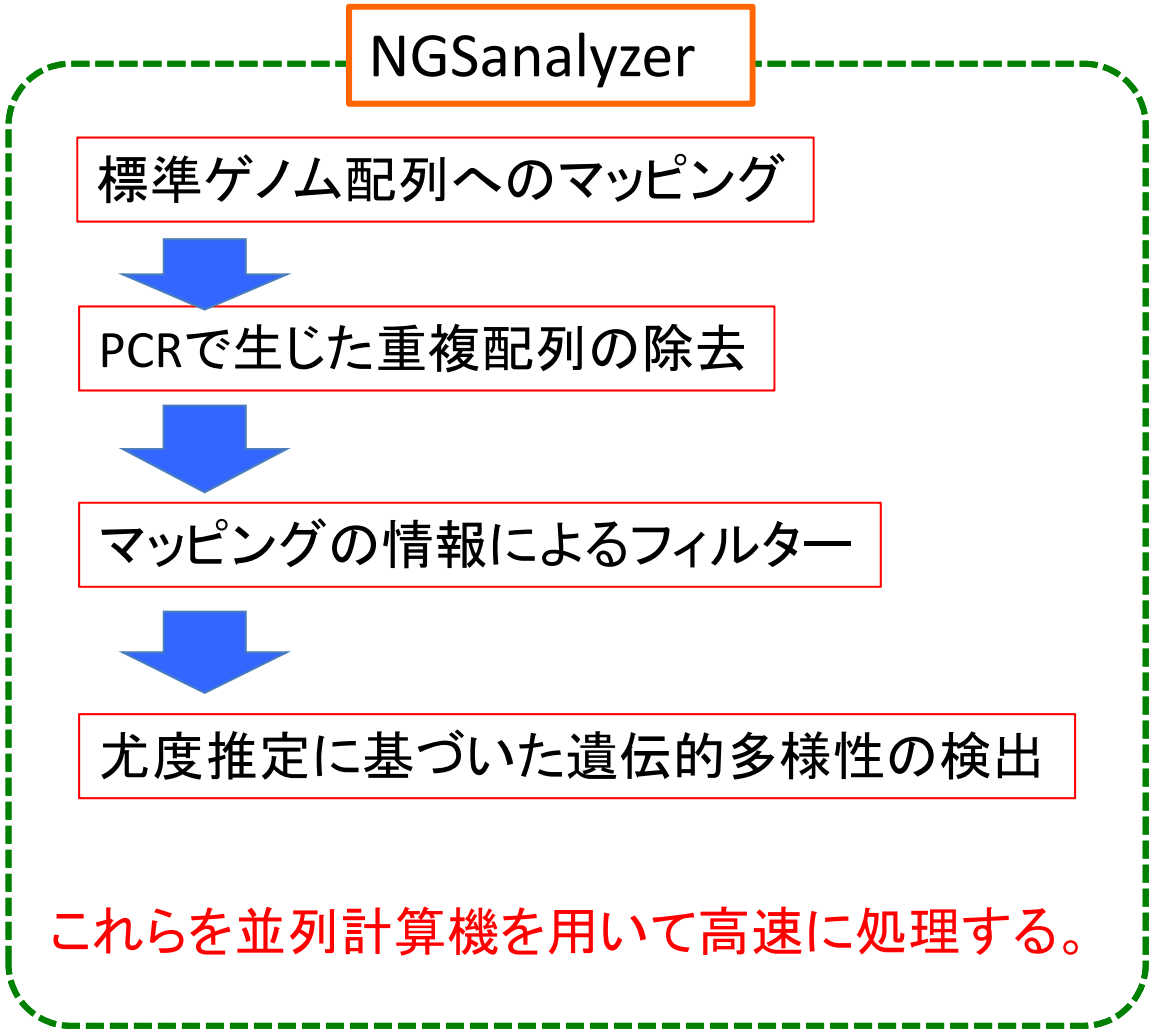


次世代シークエンサーのデータを解析するソフトウエア
NGSanalyzer

算出データ量の増大による
大量のシークエンスデータ

要望：大量のデータを
高速かつ高精度で解析

波及効果：
疾患の原因多型、がんの原因突然変異推定の高速化



NGS Analyzerの概要

次世代シークエンス解析プログラム

次世代シークエンサーの出力データを高速に解析し、
ヒト個人間の遺伝的差異やがんゲノムの突然変異を
高い正確さで同定する。

具体的には

全ゲノム（約30億塩基）の個人毎の遺伝情報の違いを
網羅的に精度よく検出することが可能

アプリケーション

がんの全突然変異を高速に検出し、
創薬のターゲット分子を探索

NGS Analyzerの特徴

大容量メモリ計算機で行っているbwaによるマッピングを、
「京」などの並列計算機で高速に行うことが可能

離散化（計算モデル化）の方法

ヒト標準ゲノム配列に対するマッピングと確率計算に
基づいた多様性検出

計算方法

直接法による密行列の対角化

動作確認環境

- 「京」
- SCLSFX-10
- RICC

NGSanalyzer解析の流れ

① bwa用のリファレンスデータの作成

② リードファイルの分割

③ アライメント

④ BAMファイルの作成とPCR duplicationの除去

⑤ pileupとSNP Call

本日の講義内容

1. 次世代シークエンス（NGS）の概要
2. NGS analyzerの概要
3. 必要な計算環境とデータファイル
4. 解析例の紹介と速度比較
5. 解析の流れ（コマンドの紹介）

必要なファイル

1. 参照(リファレンス)ゲノムデータ

Hogehoge.fa (今回はNCBIのBuild37.1を使用)

Hogehoge.fa.fai (Hogehoge.faからSmatoolsを使用して作成)

seq_contig.md (Hogehoge.faと同じ場所にあります。)

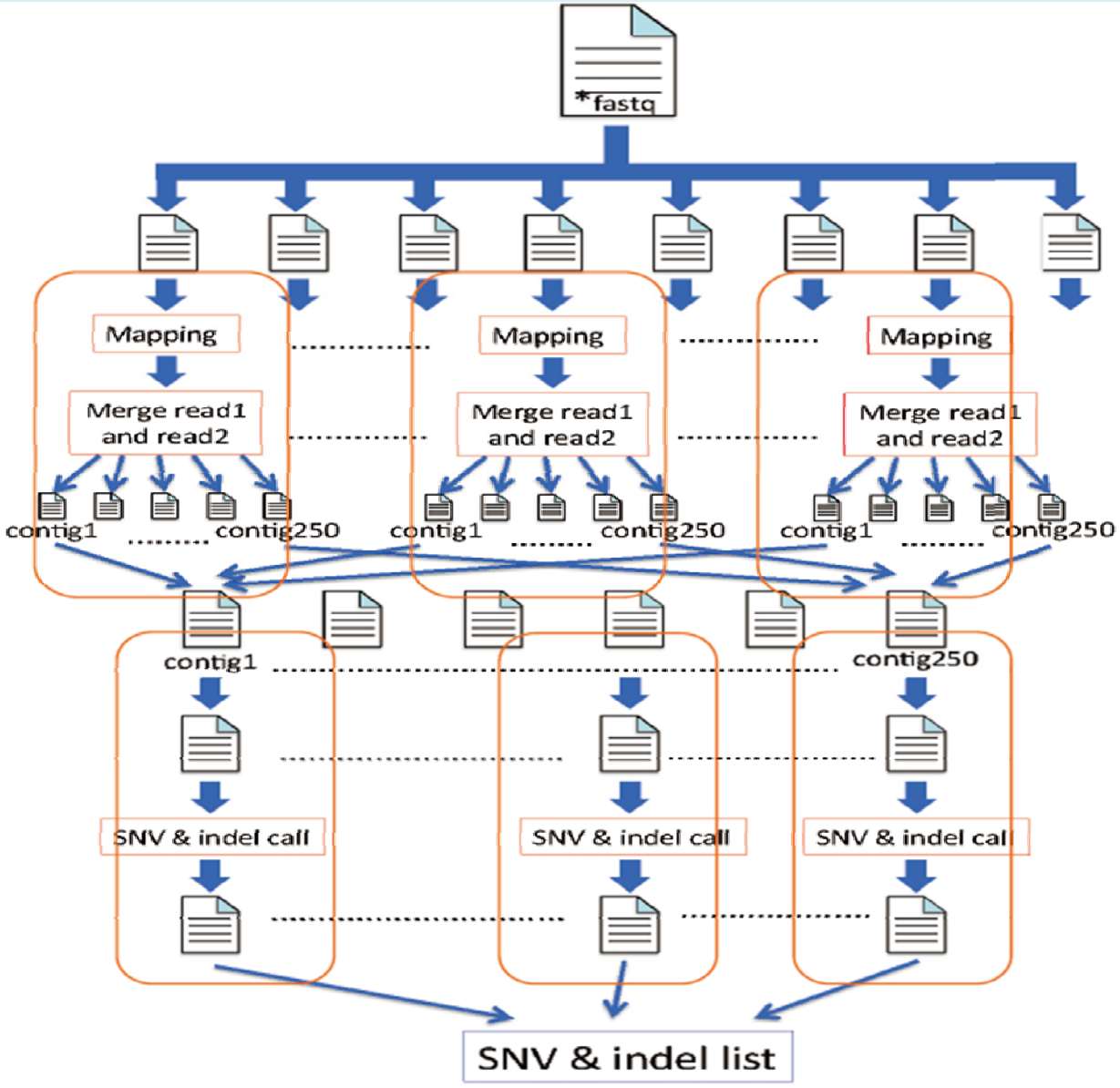
今回のデータ
ftp://ftp.ncbi.nlm.nih.gov/genomes/H_sapiens/ARCHIVE/BUILD.37.1/
hs_ref_GRCh37_chr

2. NGSデータ

動作確認しているのはIllumina社のGAから出てきたfastqファイル

今回のデータ
ftp://ftp.ddbj.nig.ac.jp/ddbj_database/dra/fastq/DRA000/DRA000222/

NGSAnayzerの並列化方法



本日の講義内容

1. 次世代シークエンス（NGS）の概要

2. NGS analyzerの概要

3. 必要な計算環境とデータファイル

4. 解析例の紹介と速度比較

5. 解析の流れ（コマンドの紹介）

NGSanalyzer解析の流れ

① bwa用のリファレンスデータの作成

② リードファイルの分割

③ アライメント

④ BAMファイルの作成とPCR duplicationの除去

⑤ pileupとSNP Call

NGSanalyzer解析の流れ

① bwa用のリファレンスデータの作成

② リードファイルの分割

③ アライメント

④ BAMファイルの作成とPCR duplicationの除去

⑤ pileupとSNP Call

開発者が行った日本人ゲノム解析

ARTICLES

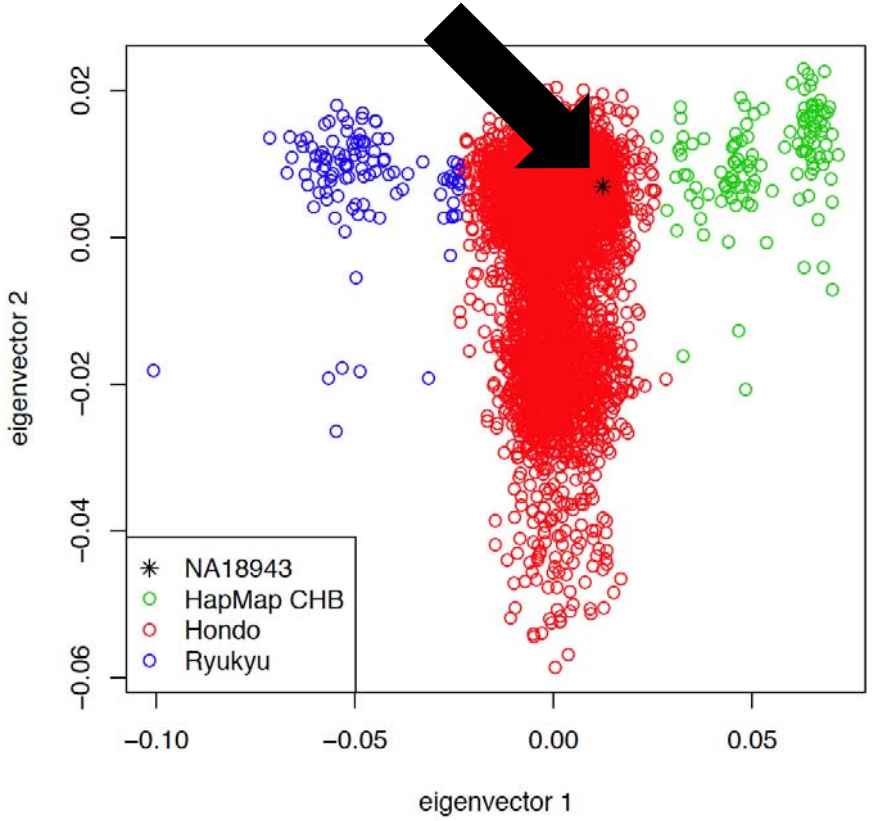
nature genetics

Whole-genome sequencing and comprehensive variant analysis of a Japanese individual using massively parallel sequencing

Akihiro Fujimoto[1,2], Hidewaki Nakagawa[1], Naoya Hosono[1], Kaoru Nakano[1], Tetsuo Abe[1], Keith A Boroevich[1], Masao Nagasaki[3], Rui Yamaguchi[3], Tetsuo Shibuya[3], Michiaki Kubo[1], Satoru Miyano[2,3], Yusuke Nakamura[1,3] & Tatsuhiko Tsunoda[1,2]

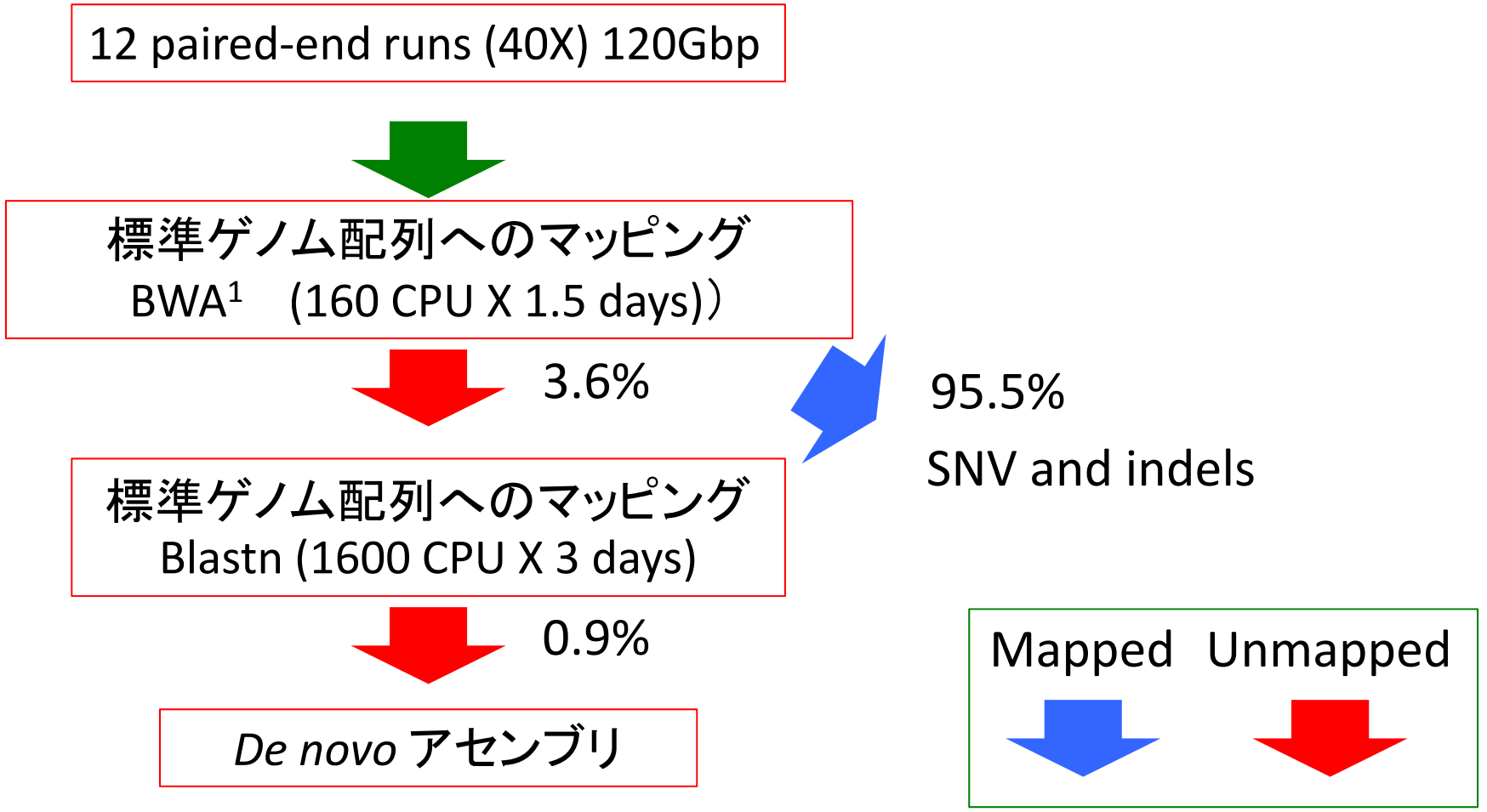
一人の日本人の全ゲノム配列決定

- HapMap JPT NA18943
  - 男性
  - 本土在住
  - 核型(染色体)の異常はない
- Sequencing
  - Illumina Genome Analyzer
  - Paired-end method
  - Library size; 200bp
  - Read length; 76 and 51bp



このデータは公開されています。

開発者が行ったNA18943の解析フローチャート



99.1% of reads were mapped to the human reference genome (build36).

1. Li & Durbin (2009)

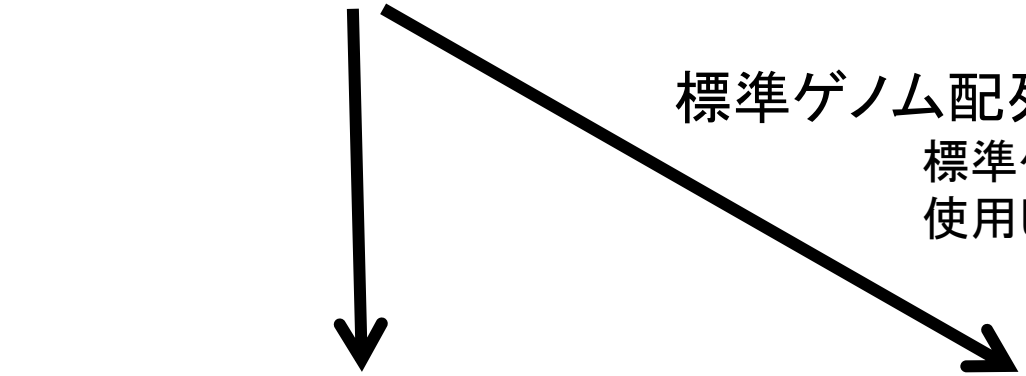
アライメント速度比較

1 paired-end runs (40X) 25Gbp

DRR000606_1.fastq (13GB)
DRR000606_2.fastq (12GB)

標準ゲノム配列へのアライメント

標準ゲノム配列としてBuild37.1を使用した。



デスクトップ計算機でbwa

CPU:Intel core i7-2820QM 2.3GHz
メモリ：16GB（4GB/core）
OS: CentOS6.4(X86_64)
コンパイラ：gcc-4.4.7
bwa:0.5.9rc(逐次)
　　0.7.4(最新版：スレッド並列)

「京」上でNGSAnalyzer

CPU:SPARC64™ VIIIfx 2GHz
メモリ：16GB/node
OS:「京」専用OS
コンパイラ:mpifccpx
NGSAnalyzer (bwa:0.5.9rc)
プロセス並列

並列化ピアノの演奏に例えると

同じ時間に多くの音を出すには？

1．人差し指1本だけでなく、5本＋両手使う

脳1個、楽譜1枚、ピアノ1台

CPU1個、メモリ1個、PC1台：スレッド並列

楽譜1枚なので1つの曲を弾く

2．人差し指1本だけで複数のピアノ＋複数の演奏者

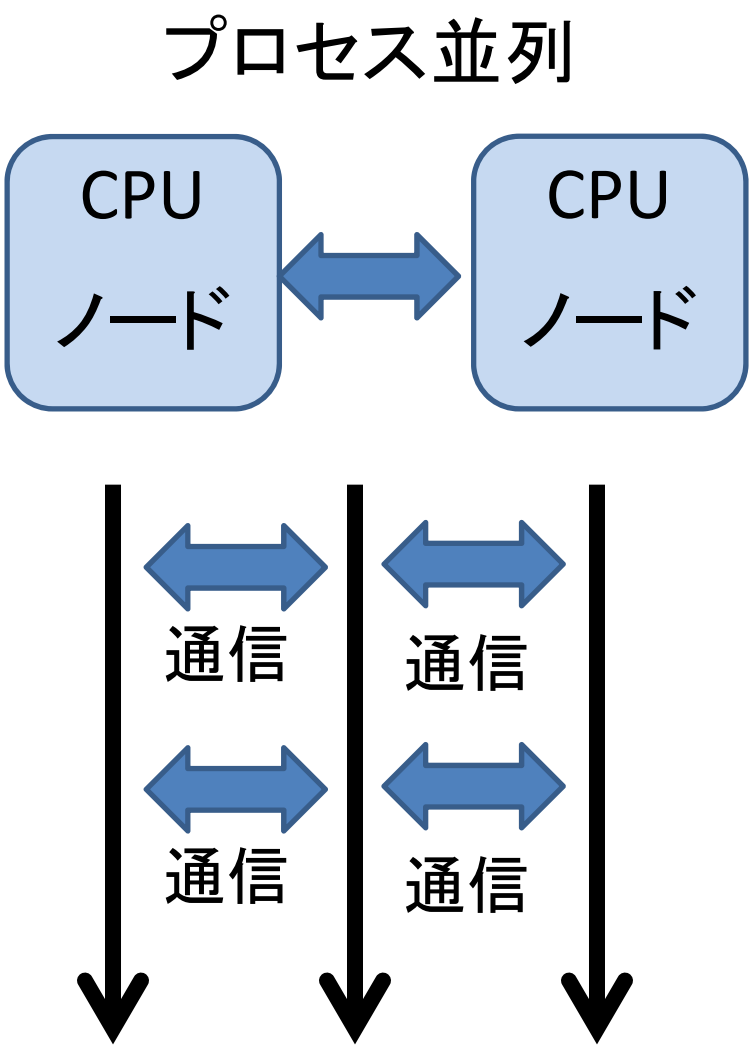
脳複数個、楽譜複数枚、ピアノ複数台

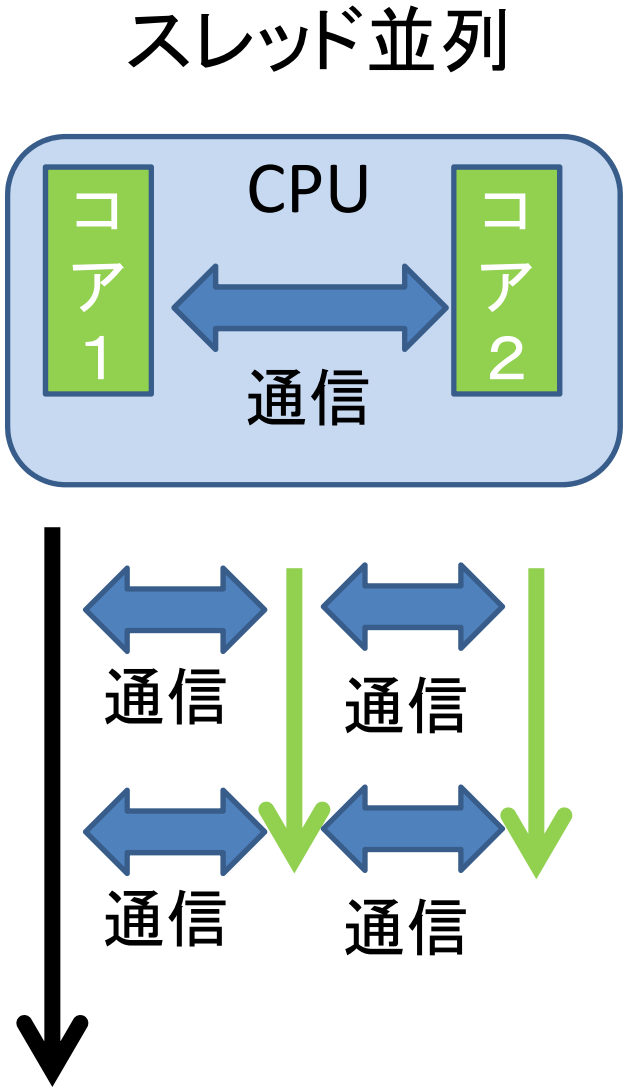
＋指揮者が必要

CPU複数、メモリ複数、PC複数：プロセス並列

楽譜複数枚なので複数（一つでも良い）の曲を同時に弾く

プロセス並列とスレッド並列





アライメント速度比較結果

NGSAnalyzer(bwa:0.5.9rc：プロセス並列)

「京」コンピュータ(1080ノード：試験利用中)：56分

「京」コンピュータ(48ノード：最新環境)：20分

bwa:0.5.9rc(逐次並列)

Intel core i7-2820QM 2.3GHz

4.5時間

bwa：0.7.4(最新版：スレッド並列（コア並列）)

Intel core i7-2820QM 2.3GHz

3時間

NGSanalyzer解析の流れ

① bwa用のリファレンスデータの作成

② リードファイルの分割

③ アライメント

④ BAMファイルの作成とPCR duplicationの除去

⑤ pileupとSNP Call

NA18943解析のまとめ

- Using BWA and blast, 99.1% of reads were mapped to human reference genome (build36).
- Quality scores were highly correlated with the observed error rates.
- The frequency and the Bayesian decision methods showed high concordance with the SNP arrays.
- False discovery rates for novel SNVs were estimated at 9.8 % and 7.1% (after excluding repeat masker regions).
- Using a Bayesian decision method, we identified 3,132,608 SNVs.
- Frequency spectrum revealed an excess of singleton nonsense and nonsynonymous SNVs, as well as singleton SNVs in conserved non-coding regions.

NGSAnalyzerの精度

DNA genotypiing arrayとの比較

DNA genotypiing arrayでgenotypingの結果を正解として、
偽陽性率、偽陰性率を見積もった。

平均40Xの全ゲノムシークエンス

偽陽性率： 0.0068%
偽陰性率： 0.17%

NGSAnalyserの速度まとめ

「京」コンピュータを使用した結果

マッピング：56分　（1080ノード）

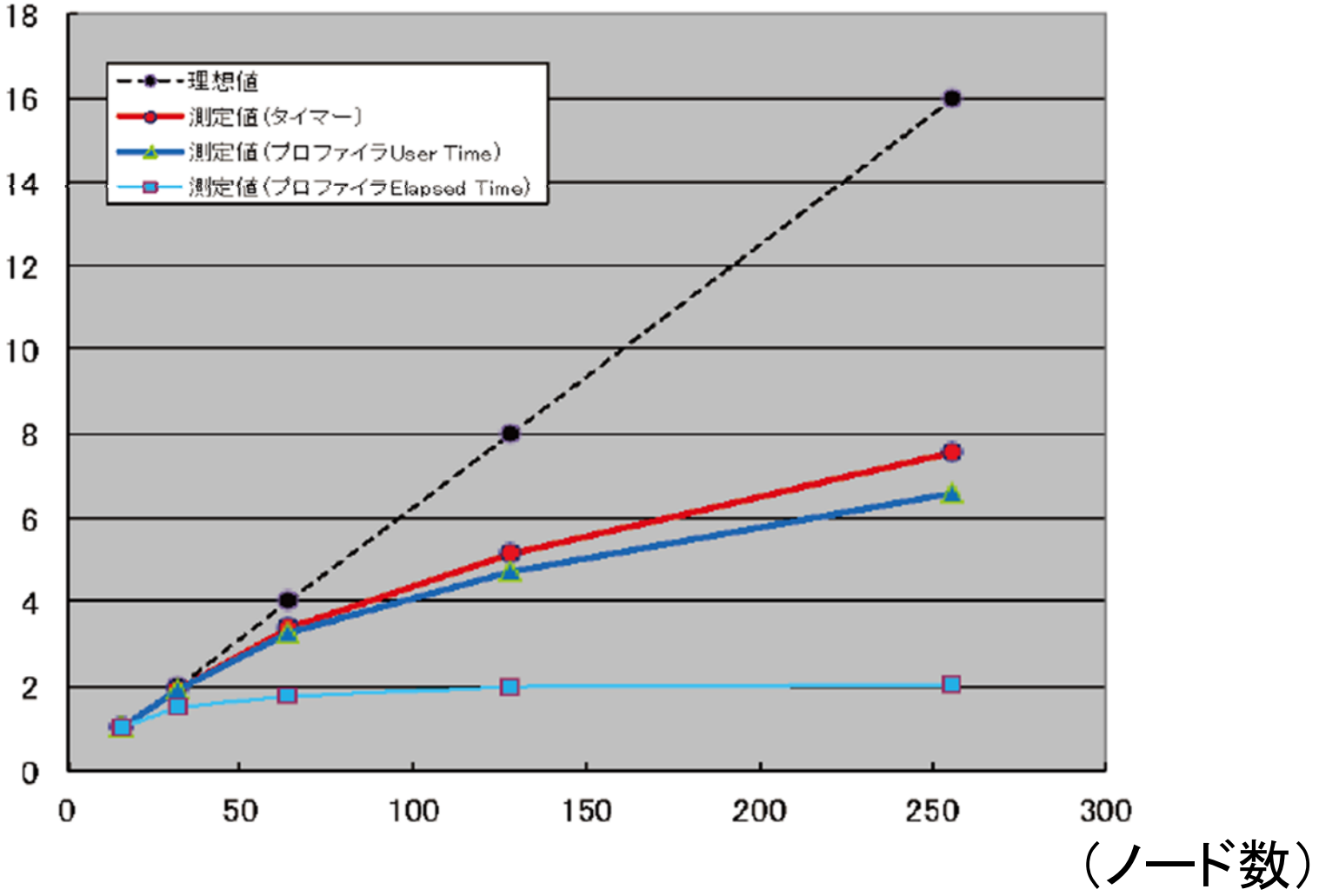
BAMファイルの作成とPCR duplicationの除去
：3時間　（32ノード）

pileupとSNP Call
：1.5時間　（32ノード）

合計：5時間半

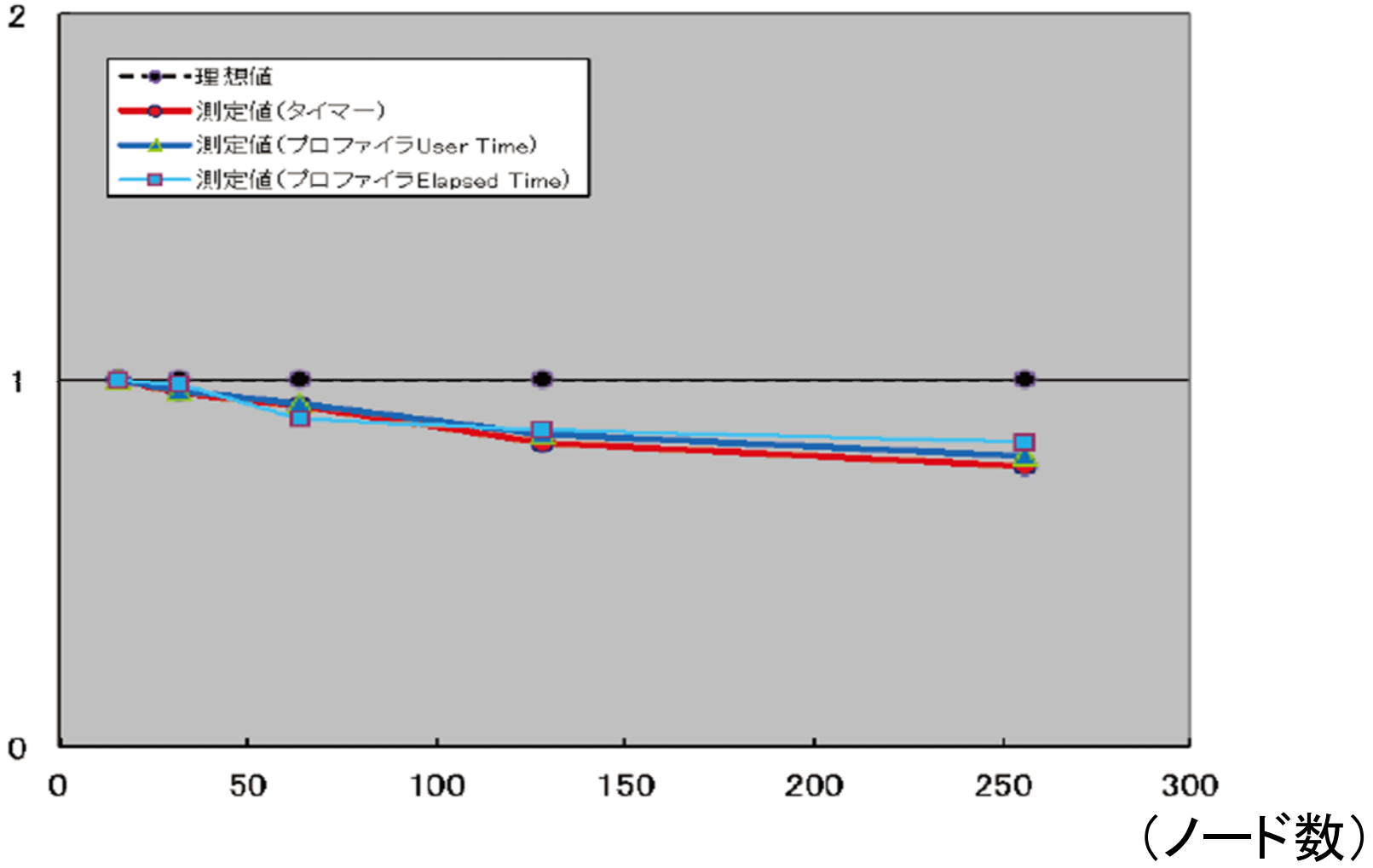
並列化効率(「京」コンピュータ:試験利用)

ストロングスケール(計算量を固定して並列数を上げる)



並列化効率(「京」コンピュータ:試験利用)

ウイークスケール(1ノードあたりの計算量が同じ)



本日の講義内容

1. 次世代シークエンス（NGS）の概要

2. NGS analyzerの概要

3. 必要な計算環境とデータファイル

4. 解析例の紹介と速度比較

5. 解析の流れ（コマンドの紹介）

NGSanlyzer の実行ファイル作成方法（build）

1. NGSanalyzer.v1.0.tgzを自分のhomeディレクトリに展開

```
$ tar -zxvf NGSanalyzer.v1.0.tgz
```

2. 展開されたkei_pipeline_r143に移動

```
$ cd kei_pipeline_r143
```

3．makeコマンドで実行ファイルを作成(build)

```
$ make
```

1．必要ファイルの準備

必要ファイルを準備

・hogehoge.fa (参照ゲノムファイル）

・Seq_Contig.md（ hogehoge.fa に対応しているファイル）

ヒト標準ゲノム配列なら以下URLからダウンロード可能

ftp://ftp.ncbi.nlm.nih.gov/genomes/H_sapiens/ARCHIVE/BUILD.37.1/CHR_*

・ hogehoge.fai (参照ゲノムリファレンスファイル)
hogehoge.fa からSamtoolsを使用して作成

・NGSデータ(fastq形式)

これらを /data/{使用許可領域}/ngs/{実験名}/inに転送

ファイルサイズが大きいため、/homeには転送できない

2. bwa用参照データを作成

bwaIndex.configをvi editorで編集

```
$vi bwaIndex.config
```

```
NODE_NO=1 (逐次実行)
ELAPSETIME=02:00:00                (実行予定時間)
TRIAL_COUNT=100000                 (pjsub を再試行する回数)
SUBMIT_SLEEP_TIME=60               (pjsub を再試行するときの待ち時間(秒))
SLEEP_TIME=30                      (ジョブ完了を監視するときの、監視間隔(秒))

BASE_DIR="/data/hp120310/k01176/ngs/DRX358"     (計算ディレクトリ)
BASE_IN_DIR="${BASE_DIR}/in"                    (入力データディレクトリ)
BASE_OUT_DIR="${BASE_DIR}/out"                  (計算結果ディレクトリ)
BASE_OUT_LOG="${BASE_DIR}/log"                  (ログディレクトリ)
GENOME_FAS_NAME=reference.fa                    (参照ゲノム名)
・・・・・・・・・・・ 以下は特に変更する必要はない。
```

計算実行

```
$ (nohup) ./do_bwaIndex.sh bwaIndex.config
```

コメント：nohupを使用すると、端末を停止しても計算は継続される

３．NGSのリードファイルを分割

split.configをvi editorで編集 $vi split.config

```
ELAPSETIME=02:00:00              (実行予定時間)
TRIAL_COUNT=100000               (pjsub を再試行する回数)
SUBMIT_SLEEP_TIME=60             (pjsub を再試行するときの待ち時間(秒))
SLEEP_TIME=30                    (ジョブ完了を監視するときの、監視間隔(秒))
BASE_DIR="/data/hp120310/k01176/ngs/DRX358" (計算ディレクトリ:例)
BASE_IN_DIR="${BASE_DIR}/in"                (入力データディレクトリ)
BASE_OUT_DIR="${BASE_DIR}/out"              (計算結果ディレクトリ)
BASE_OUT_LOG="${BASE_DIR}/log"              (ログディレクトリ)
FASTQ_DIR="${BASE_IN_DIR}/DRA000222"        (NGSデータディレクトリ)
SPLIT_SEQ_COUNT=250000                      (断片の長さ)
MPI_BIN="bin/mpi_procInfo"
SPLIT_BIN="bin/splitFastq"
JOB_CONTROL_SH=job_control.sh
COMMON_SH=common.sh
OUT_DIR="${BASE_OUT_DIR}/DRA000222"         (リードファイル分割後の結果ディレクトリ)
LOG_DIR="${BASE_OUT_LOG}/split"             (ログファイル名)
```

計算実行 $ (nohup) ./do_split.sh split.config

コメント：nohupを使用すると、端末を停止しても計算は継続される

４．アライメント

aln.configをvi editorで編集 $vi aln.config

NODE_NO=10x12x9　(計算ノード数(並列ノード数))
PROC_PER_NODE=4　(ノード内の並列数)
ELAPSETIME=05:30:00　(実行予定時間)

BASE_DIR=“/data/hp120310/k01176/ngs/DRX358”　(計算ディレクトリ例)
SEQCONTIG_MD=“${BASE_IN_DIR}/seq_contig.md”　(seq_contig.mdの場所)
GENOME_FAS_NAME=reference.fa　(参照ゲノム配列名)
GENOME_FAS="${BASE_IN_DIR}/${GENOME_FAS_NAME}"
BWA_DB_BASE="${BASE_OUT_DIR}/bwa_db"
BWA_DB="${BWA_DB_BASE}/${GENOME_FAS_NAME}"
FASTQ_SPLIT=(DRA000222 "${BASE_OUT_DIR}/DRA000222" ¥)
　　（リードファイルの分割を行った結果の名前と場所）
OUT_DIR=“${BASE_OUT_DIR}/aln“ (アライメント結果の場所)
LOG_DIR=“${BASE_OUT_LOG}/aln　(ログの場所)

計算実行　$ (nohup) ./do_aln.sh aln.config

nohupを使用すると、端末を停止しても計算は継続される

５．BAMファイル作成とPCR重複部分除去

rmdup.configをvi editorで編集　　$vi rmdup.config

```
NODE_NO=32 (計算ノード数(並列ノード数))
PROC_PER_NODE=4 (ノード内の並列数)
ELAPSETIME=04:00:00 (実行予定時間)

BASE_DIR="/data/hp120310/k01176/ngs/DRX358"
BASE_IN_DIR="${BASE_DIR}/in“ ,BASE_OUT_DIR="${BASE_DIR}/out“, BASE_OUT_LOG="${BASE_DIR}/log"

SEQCONTIG_MD="${BASE_IN_DIR}/seq_contig.md"
GENOME_FAS_NAME=reference.fa
GENOME_FAS="${BASE_IN_DIR}/${GENOME_FAS_NAME}"
SAMSRC_DIR=( ¥
  DRA000222 "${BASE_OUT_DIR}/aln/DRA000222" ¥ )

OUT_DIR="${BASE_OUT_DIR}/rmdup"
LOG_DIR="${BASE_OUT_LOG}/rmdup"
```

計算実行　　$ (nohup) ./do_rmdup.sh rmpdup.config

nohupを使用すると、端末を停止しても計算は継続される

６．PileupとSNPを同定

pileup.configをvi editorで編集　　$vi pileup.config

```
NODE_NO=32
PROC_PER_NODE=4
ELAPSETIME=03:00:00

BASE_DIR="/data/hp120311/k00548/pipe"
GENOME_FAS_NAME=reference.fa
GENOME_FAS="${BASE_IN_DIR}/${GENOME_FAS_NAME}"

PILEUP_IN_DIR=(
      "${BASE_OUT_DIR}/rmdup/DRA000222"
)

OUT_DIR="${BASE_OUT_DIR}/pileup/DRA000222"
LOG_DIR="${BASE_OUT_LOG}/pileup"
```

計算実行　　$ (nohup) ./do_rmdup.sh rmpdup.config

nohupを使用すると、端末を停止しても計算は継続される

さいごに

ぜひ、利用してください。

必要ファイル作成方法や不具合など、
ご不明な点があればいつでもご連絡ください。

お疲れ様でした。＆　ありがとうございました。