

医疗智能体

最佳实践

文档版本 01
发布日期 2024-11-05



版权所有 © 华为云计算技术有限公司 2024。保留一切权利。

非经本公司书面许可，任何单位和个人不得擅自摘抄、复制本文档内容的部分或全部，并不得以任何形式传播。

商标声明



HUAWEI和其他华为商标均为华为技术有限公司的商标。

本文档提及的其他所有商标或注册商标，由各自的所有人拥有。

注意

您购买的产品、服务或特性等应受华为云计算技术有限公司商业合同和条款的约束，本文档中描述的全部或部分产品、服务或特性可能不在您的购买或使用范围之内。除非合同另有约定，华为云计算技术有限公司对本文档内容不做任何明示或暗示的声明或保证。

由于产品版本升级或其他原因，本文档内容会不定期进行更新。除非另有约定，本文档仅作为使用指导，本文档中的所有陈述、信息和建议不构成任何明示或暗示的担保。

目录

1 基于二代测序的基因组突变检测.....	1
1.1 NGS 流程简介.....	1
1.2 配置命令行工具.....	3
1.3 上传数据.....	6
1.4 制作并上传镜像.....	6
1.5 创建应用.....	10
1.6 搭建 NGS 流程.....	17
1.7 执行分析作业.....	19
1.8 批量执行 NGS 分析.....	24
2 新型冠状病毒（COVID-19）虚拟药物筛选.....	28
2.1 虚拟药物筛选简介.....	28
2.2 获取示例数据.....	29
2.3 创建虚拟药物筛选任务.....	30

1 基于二代测序的基因组突变检测

1.1 NGS 流程简介

二代基因组测序即Next Generation Sequencing (NGS)是一种基于边合成边测序的方式。NGS在保持了测序高准确度的同时，大幅地提高了测序速度，有力推动了相关研究。目前，NGS已广泛应用于全基因组测序、外显子测序、表观遗传学修饰等重要的生物学问题。

本示例中NGS流程基于医疗智能体（EIHealth）平台搭建，流程以fastq格式数据作为输入，对碱基的质量信息进行评估，判断可靠程度，通过质控、比对、变异检测等步骤，最终输出包含样本SNP、INDEL的VCF文件。

📖 说明

该案例介绍NGS的搭建步骤，涵盖镜像、应用、流程制作方法。用户也可以使用“资产市场”提供的已经搭建好的“Variant Calling Based On NGS”流程。该案例比“资产市场”流程多出VCF文件进行质控步骤。

功能介绍

- **测序数据质量的总体评估**
评估测序的Reads数目，测序Base数，测序深度等。
- **低质量Reads过滤**
过滤低质量的测序Reads，得到Clean Reads。
- **基因组比对**
将Clean Reads比对到参考基因组上，同时输出比对率、深度、覆盖度的统计信息。
- **基因组变异检测**
基于上述比对得到的bam文件，通过GATK4做Variant Calling，输出变异检测结果。
- **基因组变异检测质控**
通过VariantQC对vcf进行质量控制，输出变异数目，变异类型统计等指标。

流程优势

- 使用Unix管道技术连接比对和排序步骤，以缩短bwa和samtools的存放、读取、删除中间文件的时间。
- 流程针对GATK4中的限速步骤，进行了系统的优化加速。流程从contig-file中提取contig，根据contig下发对应的任务，并依据不同任务，指定并行下发的任务数，以降低流程整体的运行时间。

流程执行信息

NGS流程由fastp、bwa-mem、picard-insertsize、qualimap-bamqc、gatk-markduplicates、gatk-bqsr、gatk-applybqsr、gatk-haplotypecaller、gatk-mergevcfs和discvrseq-variantqc应用构成。NGS流程执行步骤如表1-1所示。

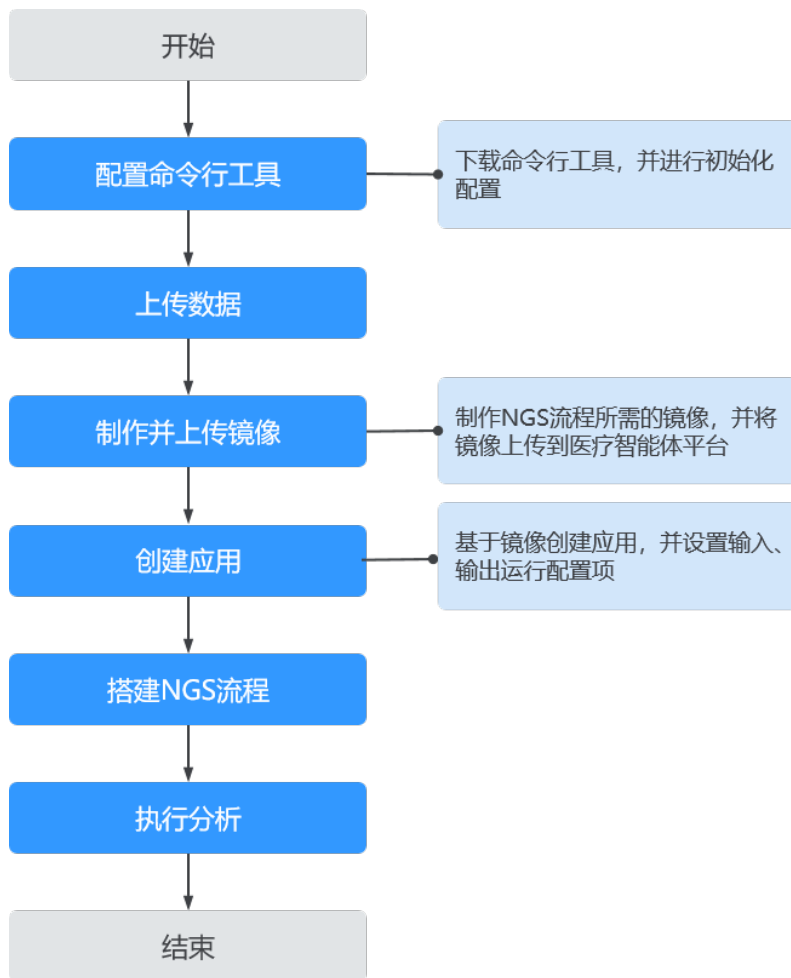
表 1-1 NGS 执行步骤

步骤	描述
Read Quality	对测序得到的fastq数据进行质控。
Mapping and Sort and index	将质控之后得到的Clean Reads比对到参考基因组上。
Insert Size Estimation	针对构建Index后的bam文件，统计测序数据的Insert size的分布。
Bam QC	评估比对得到的bam文件的质量。
GATK MarkDuplicates	标记比对bam文件中的重复Reads。
gatk BaseRecalibrator	基于比对bam文件评估校正参数。
gatk ApplyBQSR	基于比对bam文件进行校正。
gatk HaplotypeCaller	基于比对和校正之后的bam文件进行Variant Calling的工作。
gatk MergeVcfs	合并分bin变异检测的VCF文件。
Variant QC	针对输出的VCF文件进行质控。

图 1-1 NGS 执行步骤



图 1-2 搭建步骤



1.2 配置命令行工具

医疗智能体平台命令行工具（eihealth-toolkit）是配套EIHealth平台，为EIHealth平台各功能组件提供命令行管理工具。借助此工具，可以辅助您对EIHealth平台项目中数据、应用、流程和作业资源进行管理和使用。

操作步骤

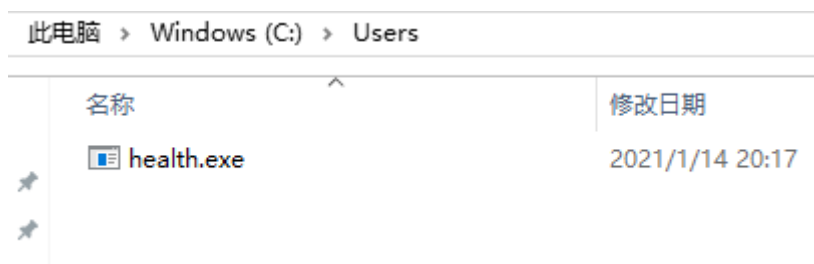
步骤1 下载命令行工具。

步骤2 安装命令行工具。

本示例中以Windows系统为例，介绍安装命令行工具的方法。

1. 下载Windows版本的客户端，得到health.exe文件，health文件无需安装，放置在任一文件夹中即可。

图 1-3 下载命令行工具



2. 使用win键+R，输入cmd打开windows的cmd窗口。进入工具所在的目录，输入health命令，即可使用。

如果cmd窗口显示目录不是health文件所在目录，请使用cd命令切换路径。例如，切换至D盘：

```
cd /d d:
```

说明

使用Linux版本命令行工具时，您需要在本地搭建Linux环境，并将下载的health文件放至所需的目录下。

- 如果当前目录为health所在目录，可以使用./health命令使用命令行工具。
- 如果当前目录不是health所在目录，需要使用绝对路径。如当前目录为/opt，假设health存放在/root/health-toolkit/下，需要指定/root/health-toolkit/health路径进行使用。
- 如果无法运行，提示Permission denied，请使用chmod 755 health命令设置执行权限。

步骤3 初始化配置。

在使用命令行工具前，需要初始化配置信息。执行health config add命令配置AK/SK，区域名称，华为云项目ID信息，获取方法请参见[获取认证信息](#)。

- 命令结构
health config add [flags]

表 1-2 参数说明

参数	简写	是否必选	说明
--ak	-a	是	AK(Access Key ID)：访问密钥ID。
--sk	-s	是	SK(Secret Access Key)：与访问密钥ID结合使用的密钥。
--region	-r	是	服务区域名称。
--platform-id	-i	是	华为云项目ID，请按 获取认证信息 中的方法获取。
--log-path	-l	否	日志路径，不填写时默认为命令行工具当前路径下healthcli.log文件。
--http-proxy	-p	否	HTTP代理配置，格式为“http://username:password@your-proxy:your-port”。

参数	简写	是否必选	说明
--swr-endpoint	-t	是	SWR镜像仓库地址。 获取方式： 1. 登录容器镜像服务管理控制台。 2. 单击界面右侧“登录指令”，获取内网登录指令末尾的SWR镜像仓库地址。例如100.78.15.50:20202。
--iam-endpoint	-m	是	IAM终端节点名称，请在 地区与终端节点 中获取。
--health-endpoint	-e	是	EIHealth终端节点名称，请在 地区与终端节点 中获取。
--obs-endpoint	-o	是	OBS终端节点名称，请在 地区与终端节点 中获取。
--obs-install-path	-q	否	设置obsutil安装路径，默认安装在当前运行目录。 设置时，该路径必须为obsutil运行文件名，如/home/path/obsutil、/home/path/obsutil-1.1.1
--obs-down-load-url	-D	否	obsutil下载链接，obsutil将下载到obs-install-path上。 参数有改动时才会触发下载。 下载链接的内容可以是zip、tar.gz文件、二进制文件，如果是压缩文件，文件夹内的obsutil必须命名为obsutil（和obsutil官方链接保持一致）。
--force	-f	否	强制操作。如果下载obsutil时，指定的obs-install-path上已经有同名文件，不带-f时会提示用户，带上-f会直接覆盖原文件。

- 命令示例

```
health config add --ak CAIxxxxxxxxFE --sk QLFxxxxxxxxxtNvsF --region cn-north-4 --platform-id
catdi9fb689 --swr-endpoint 100.78.15.50:20202 --iam-endpoint iam.cn-north-4.myhuaweicloud.com
--health-endpoint eihealth.cn-north-4.myhuaweicloud.com --obs-endpoint obs.cn-
north-4.myhuaweicloud.com
# 执行成功返回结果如下
add ak successfully!
add sk successfully!
add region successfully!
add platform-id successfully!
add swr-endpoint successfully!
add iam-endpoint successfully!
add health-endpoint successfully!
add obs-endpoint successfully!
```


📖 说明

- 执行以上命令，会在系统所在的用户目录下自动生成“.health”文件夹，文件夹中包含config.ini配置文件，用于存储任务执行所涉及到的配置，如密钥、区域、当前项目等信息。
- 生成的配置文件不建议直接修改，如需改动请使用命令行工具修改。
- 配置文件中保存有用户的AK、SK信息，为了避免密钥泄露，会对文件中的SK进行加密以保护密钥安全。

----结束

1.3 上传数据

NGS流程中需使用二代测序得到的原始fastq文件、参考基因组序列、参考Variants数据集。

本示例中以Windows系统命令行工具为例，介绍如何将本地数据上传到EIHealth平台。更多的命令介绍请参见[命令行工具](#)。

步骤1 使用命令行工具，用switch命令进入待操作的项目。

例如，使用health switch project ngs-project命令进入到名为ngs-project的项目中。

步骤2 使用命令行工具，用mkdir命令创建存储数据的文件夹。

例如，使用health mkdir input-data命令创建名为input-data的文件夹。

步骤3 将本地数据上传至项目文件夹中。

例如，将Linux系统下root/health_test路径中xxx.R1.fastq.gz数据上传至ngs-project项目的input-data文件夹中。

```
./health upload /root/health_test/xxx.R1.fastq.gz /input-data/
```

例如，Windows系统下将本地D盘中xxx.R1.fastq.gz数据上传至ngs-project项目的input-data文件夹中。

```
health upload D:\local\data\xxx.R1.fastq.gz /input-data/
```

----结束

1.4 制作并上传镜像

NGS流程由fastp、bwa-mem、picard-insertsize、qualimap-bamqc、gatk-markduplicates、gatk-bqsr、gatk-applybqsr、gatk-haplotypecaller、gatk-mergevcfs和discvrseq-variantqc应用构成。在创建应用前，请先制作并上传应用所需的镜像。搭建NGS流程所需的镜像和版本如[表1-3](#)所示。

表 1-3 NGS 流程镜像信息

NGS流程步骤	描述	依赖镜像及版本	镜像下载命令
Read Quality	对测序得到的fastq数据进行质控。	fastp:0.20.1	docker pull biocontainers/fastp:v0.20.1_cv1

NGS流程步骤	描述	依赖镜像及版本	镜像下载命令
Mapping and Sort and index	将质控之后得到的 Clean Reads比对到参考基因组上。	bwa-mem:0.7.17	该镜像由bwa和samtools合并而成，镜像制作方法请参见 制作bwa-mem镜像 。
Insert Size Estimation	针对构建Index后的bam文件，统计测序数据的Insert size的分布。	picard-insertsize:2.23.3	docker pull broadinstitute/picard:2.23.3
Bam QC	评估比对得到的bam文件的质量。	qualimap-bamqc:2.0.0	docker pull maxulyse/qualimap:2.0.0
GATK MarkDuplicates	标记比对bam文件中的重复Reads。	gatk-markduplicates:4.1.9.0	docker pull broadinstitute/gatk:4.1.9.0
gatk BaseRecalibrator	基于比对bam文件评估校正参数。	gatk-bqsr:4.1.9.0	
gatk ApplyBQSR	基于比对bam文件进行校正。	gatk-applybqsr:4.1.9.0	
gatk HaplotypeCaller	基于比对和校正之后的bam文件进行Variant Calling的工作。	gatk-haplotypecaller:4.1.9.0	该镜像由gatk和parallel合并而成，以提高并行运行效率，镜像制作方法请参见 制作gatk-haplotypecaller镜像 。
gatk MergeVcfs	合并分bin变异检测的VCF文件。	gatk-mergevcfs:4.1.9.0	docker pull broadinstitute/gatk:4.1.9.0
Variant QC	针对输出的VCF文件进行质控。	discvrseq-variantqc:1.17	docker pull bbimber/discvrseq:release-1.17

制作 bwa-mem 镜像

步骤1 在本地搭建Docker环境。

要求安装的容器引擎版本必须为1.11.2及以上。

步骤2 下载bwa和samtools软件。

```
wget http://downloads.sourceforge.net/project/bio-bwa/bwa-0.7.17.tar.bz2
wget https://github.com/samtools/samtools/releases/download/1.10/samtools-1.10.tar.bz2
```

步骤3 编写Dockerfile将bwa和samtools镜像合并。

详细的Dockerfile指令请参见[Dockerfile参考](#)。

1. 执行**vi Dockerfile**命令，进入Dockerfile文件中，编写文件。

```
FROM centos

ENV PATH $PATH:/usr/local/samtools/bin:/usr/local/bwa-0.7.17

ADD ./bwa-0.7.17.tar.bz2 /usr/local
ADD ./samtools-1.10.tar.bz2 /opt
RUN yum makecache && \
    yum install -y make gcc ncurses-devel bzip2-devel xz-devel zlib-devel&& \
    cd /usr/local/bwa-0.7.17 && make && \
    cd /opt/samtools-1.10 && ./configure --prefix=/usr/local/samtools && make && make install
```

2. 按Esc键，并执行:wq保存并退出Dockerfile。

3. 制作镜像。

```
docker build -t bwa_samtools:0.7.17-1.10 .
```

----结束

制作 gatk-haplotypecaller 镜像

步骤1 在本地搭建Docker环境。

要求安装的容器引擎版本必须为1.11.2及以上。

步骤2 编写Dockerfile制作gatk-haplotypecaller镜像。

1. 执行**vi Dockerfile**命令，进入Dockerfile文件中，编写文件。

```
FROM broadinstitute/gatk:4.1.9.0
RUN apt-get update
RUN apt-get install -y parallel
```

2. 按Esc键，并执行:wq退出Dockerfile。

3. 制作镜像。

```
docker build -t gatk-haplotypecaller:4.1.9.0 .
```

详细的Dockerfile指令请参见[Dockerfile参考](#)。

----结束

上传镜像

请依据[表1-3](#)提供的镜像下载命令下载搭建NGS流程所需的镜像。并依据[制作bwa-mem镜像](#)和[制作gatk-haplotypecaller镜像](#)制作镜像。制作好后的镜像如[图1-4](#)所示，请按照以下步骤将镜像上传至EIHealth平台。

图 1-4 NGS 流程镜像

REPOSITORY	TAG	IMAGE ID	CREATED	SIZE
bwa_samtools	0.7.17-1.10	83cabbe38205	24 hours ago	435MB
gatk-haplotypecaller	4.1.9.0	c4420566e7b2	32 hours ago	4.7GB
broadinstitute/gatk	4.1.9.0	7704495fb519	6 months ago	4.66GB
broadinstitute/picard	2.23.3	eb2818980cc5	9 months ago	1.35GB
bbimber/discvrseq	release-1.17	b6cb5edb1696	9 months ago	2.32GB
biocontainers/fastp	v0.20.1_cv1	9a1f8667dcfd	10 months ago	890MB
zlskidmore/samtools	1.10	ad023751d7b5	14 months ago	735MB
maxulyse/qualimap	2.0.0	d4bb27954030	3 years ago	783MB

- 步骤1 使用命令行工具，使用**health switch project <project-name>**命令进入待操作的项目。

例如，使用 `health switch project ngs-project` 命令进入到名为 `ngs-project` 的项目中。

步骤2 通过 `health docker tag` 命令给要上传的镜像打标签。

- 命令结构

`health docker tag <source-image-name:source-tag-name> <target-image-name:target-tag-name>`

命令中 `source-image-name` 为镜像名称，`source-tag-name` 为镜像标签，`target-image-name` 和 `target-tag-name` 可自定义。

- 打标签命令

依次执行以下命令为镜像打标签。

```
health docker tag biocontainers/fastp:v0.20.1_cv1 fastp:0.20.1
health docker tag bwa_samtools:0.7.17-1.10 bwa_samtools:0.7.17-1.10
health docker tag broadinstitute/picard:2.23.3 picard-insertsize:2.23.3
health docker tag maxulyse/qualimap:2.0.0 qualimap-bamqc:2.0.0
health docker tag broadinstitute/gatk:4.1.9.0 gatk-markduplicates:4.1.9.0
health docker tag broadinstitute/gatk:4.1.9.0 gatk-bqsr:4.1.9.0
health docker tag broadinstitute/gatk:4.1.9.0 gatk-applybqsr:4.1.9.0
health docker tag gatk-haplotypecaller:4.1.9.0 gatk-haplotypecaller:4.1.9.0
health docker tag broadinstitute/gatk:4.1.9.0 gatk-mergevcs:4.1.9.0
health docker tag bbimber/discvrseq:release-1.17 discvrseq-variantqc:1.17
```

步骤3 使用 `health docker push` 命令上传镜像。

- 命令结构

`health docker push <image-name:tag-name>`

- 上传镜像命令

依次执行以下命令将镜像上传至 EIHealth 平台。

```
health docker push fastp:0.20.1
health docker push bwa_samtools:0.7.17-1.10
health docker push picard-insertsize:2.23.3
health docker push qualimap-bamqc:2.0.0
health docker push gatk-markduplicates:4.1.9.0
health docker push gatk-bqsr:4.1.9.0
health docker push gatk-applybqsr:4.1.9.0
health docker push gatk-haplotypecaller:4.1.9.0
health docker push gatk-mergevcs:4.1.9.0
health docker push discvrseq-variantqc:1.17
```

步骤4 登录医疗智能体平台，进入项目并选择“镜像”页签，在镜像列表中查看已上传的镜像。

图 1-5 镜像列表

名称	源项目	芯片类型	镜像类型	描述	创建时间	更新时间	操作
druglikeness_apitest_k	eihealth_api_project_0	X86	OTHER	--	2022/01/07 15:43...	2022/01/07 15:43...	删除
autodruglikeness	eihealth_api_project_0	X86	OTHER	--	2022/01/07 15:35...	2022/01/07 15:35...	删除
bwa_apitest_other_loc	eihealth_api_project_0	--	OTHER	--	2022/01/07 15:10...	2022/01/07 15:10...	删除
test_test	eihealth_api_project_0	--	OTHER	--	2022/01/04 17:35...	2022/01/04 17:35...	删除
test-test	eihealth_api_project_0	ARM	OTHER	--	2022/01/04 17:35...	2022/01/04 17:35...	删除
druglikeness	eihealth_api_project_0	--	OTHER	--	2022/01/04 17:25...	2022/01/04 17:25...	删除

---结束

1.5 创建应用

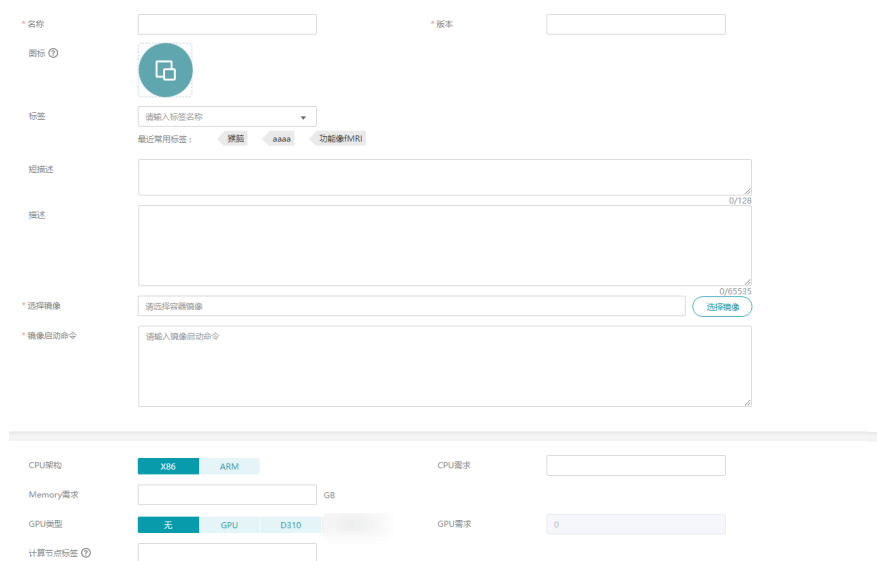
步骤1 登录医疗智能体平台，进入项目并选择“工具 > 应用”页签，单击“新建应用”。

图 1-6 新建应用



步骤2 依据“应用参数说明表”依次创建搭建NGS流程所需的应用。

图 1-7 填充应用内容



说明

对于测序得到的大量数据，如果需要批量执行NGS分析，可以选取以下任意一种方式进行批量执行：

- 方式一：对于输入参数，打开“并发”开关，在启动作业时，每个参数可以设置多个参数值，自动生成多个作业并发执行。并发执行的作业数为设置的参数值个数的乘积。
例如，存在输入参数a和输入参数b，在启动作业时，分别给参数a设置了2个参数值，给参数b设置了2个参数值。那么，系统将自动生成4个作业并发执行。
- 方式二：参考[批量执行NGS分析](#)操作。

步骤3 参数确认无误后，单击“立即创建”，创建应用。

----结束

表 1-4 fastp、bwa_samtools 应用参数说明

<p>应用名称和版本</p>	<ul style="list-style-type: none"> 应用名称: fastp 版本: 0.20.1 	<ul style="list-style-type: none"> 应用名称: bwa_samtools 版本: 0.7.17
<p>镜像和启动命令</p>	<ul style="list-style-type: none"> 镜像: fastp:0.20.1 启动命令: fastp --fix_mgi_id -w 16 -i \${fastq-file1} -l \${fastq-file2} -o \${fq-file1} -O \${fq-file2} -j \${json-file} -h \${html-file} 	<ul style="list-style-type: none"> 镜像: bwa_samtools:0.7.17-1.10 启动命令: bwa mem -t 32 -M -R '@RG\tID:\${sample-id}\tPL:\${seq-platform}\tPU:\${sample-id}\tSM:\${sample-id}\tLB:\${sample-id}' \${ref-file} \${fq-file1} \${fq-file2} samtools sort -@32 -m 3G -o \${sorted-bam} && samtools index \${sorted-bam} && samtools flagstat \${sorted-bam} -@10 > \${flagstat-file}
<p>CPU、内存和 GPU</p>	<ul style="list-style-type: none"> CPU架构: X86 CPU需求: 0.1 Memory: 0.1 GPU类型: 无 GPU需求: 0 	<ul style="list-style-type: none"> CPU架构: X86 CPU需求: 16 Memory: 10 GPU类型: 无 GPU需求: 0
<p>输入参数</p>	<ul style="list-style-type: none"> 参数1 <ul style="list-style-type: none"> 参数名称: fastq-file1 数据类型: File 必传: 是 参数2 <ul style="list-style-type: none"> 参数名称: fastq-file2 数据类型: File 必传: 是 	<ul style="list-style-type: none"> 参数1 <ul style="list-style-type: none"> 参数名称: sample-id 数据类型: String 必传: 是 参数2 <ul style="list-style-type: none"> 参数名称: seq-platform 数据类型: String 必传: 是 参数3 <ul style="list-style-type: none"> 参数名称: fq-file1 数据类型: File 必传: 是 参数4 <ul style="list-style-type: none"> 参数名称: fq-file2 数据类型: File 必传: 是 参数5 <ul style="list-style-type: none"> 参数名称: ref-file 数据类型: File 必传: 是

输出参数	<ul style="list-style-type: none"> ● 参数1 <ul style="list-style-type: none"> - 参数名称: fq-file1 - 数据类型: File - 必传: 是 - 默认值: /fastp_1.fq.gz ● 参数2 <ul style="list-style-type: none"> - 参数名称: fq-file2 - 数据类型: File - 必传: 是 - 默认值: /fastp_2.fq.gz ● 参数3 <ul style="list-style-type: none"> - 参数名称: json-file - 数据类型: File - 必传: 是 - 默认值: /fastp.json ● 参数4 <ul style="list-style-type: none"> - 参数名称: html-file - 数据类型: File - 必传: 是 - 默认值: /fastp.html 	<ul style="list-style-type: none"> ● 参数1 <ul style="list-style-type: none"> - 参数名称: sorted-bam - 数据类型: File - 必传: 是 - 默认值: /bwa-mem.sorted.bam ● 参数2 <ul style="list-style-type: none"> - 参数名称: flagstat-file - 数据类型: File - 必传: 是 - 默认值: /samtools-flagstat.txt
-------------	--	--

表 1-5 qualimap-bamqc、picard-insertsize 应用参数说明

应用名称和版本	<ul style="list-style-type: none"> ● 应用名称: qualimap-bamqc ● 版本: 2.0.0 	<ul style="list-style-type: none"> ● 应用名称: picard-insertsize ● 版本: 2.23.3
镜像和启动命令	<ul style="list-style-type: none"> ● 镜像: qualimap-bamqc:2.0.0 ● 启动命令: qualimap bamqc -bam \${bam-file} -outdir \${out-dir} -nt 8 --java-mem-size=80G 	<ul style="list-style-type: none"> ● 镜像: picard-insertsize:2.23.3 ● 启动命令: java -jar /usr/picard/picard.jar CollectInsertSizeMetrics I=\${bam-file} R=\${ref-file} O=\${insertsize-txt} H=\${insertsize-pdf}
CPU、内存和 GPU	<ul style="list-style-type: none"> ● CPU架构: X86 ● CPU需求: 0.1 ● Memory: 0.2 ● GPU类型: 无 ● GPU需求: 0 	<ul style="list-style-type: none"> ● CPU架构: X86 ● CPU需求: 0.1 ● Memory: 0.1 ● GPU类型: 无 ● GPU需求: 0

<p>输入参数</p>	<ul style="list-style-type: none"> ● 参数名称: bam-file ● 数据类型: File ● 必传: 是 	<ul style="list-style-type: none"> ● 参数1 <ul style="list-style-type: none"> - 参数名称: bam-file - 数据类型: File - 必传: 是 ● 参数2 <ul style="list-style-type: none"> - 参数名称: ref-file - 数据类型: File - 必传: 是
<p>输出参数</p>	<ul style="list-style-type: none"> ● 参数名称: out-dir ● 数据类型: Directory ● 必传: 是 ● 默认值: /bamqc 	<ul style="list-style-type: none"> ● 参数1 <ul style="list-style-type: none"> - 参数名称: insertsize-txt - 数据类型: File - 必传: 是 - 默认值: /picard-insertsize.txt ● 参数2 <ul style="list-style-type: none"> - 参数名称: insertsize-pdf - 数据类型: File - 必传: 是 - 默认值: /picard-insertsize.pdf

表 1-6 gatk-markduplicates、gatk-bqsr 应用参数说明

<p>应用名称和版本</p>	<ul style="list-style-type: none"> ● 应用名称: gatk-markduplicates ● 版本: 4.1.9.0 	<ul style="list-style-type: none"> ● 应用名称: gatk-bqsr ● 版本: 4.1.9.0
<p>镜像和启动命令</p>	<ul style="list-style-type: none"> ● 镜像: gatk-markduplicates:4.1.9.0 ● 启动命令: gatk MarkDuplicates --CREATE_INDEX --TMP_DIR \${out-dir} -I \${bam-file} -M \${matrices-file} -O \${markduped-bam} 	<ul style="list-style-type: none"> ● 镜像: gatk-bqsr:4.1.9.0 ● 启动命令: gatk BaseRecalibrator -R \${ref-file} -I \${markduped-bam} --known-sites \${know-site1} --known-sites \${know-site2} --known-sites \${know-site3} -O \${recal-table}
<p>CPU、内存和 GPU</p>	<ul style="list-style-type: none"> ● CPU架构: X86 ● CPU需求: 3 ● Memory: 0.2 ● GPU类型: 无 ● GPU需求: 0 	<ul style="list-style-type: none"> ● CPU架构: X86 ● CPU需求: 3 ● Memory: 0.2 ● GPU类型: 无 ● GPU需求: 0

<p>输入参数</p>	<ul style="list-style-type: none"> ● 参数名称: bam-file ● 数据类型: File ● 必传: 是 	<ul style="list-style-type: none"> ● 参数1 <ul style="list-style-type: none"> - 参数名称: ref-file - 数据类型: File - 必传: 是 ● 参数2 <ul style="list-style-type: none"> - 参数名称: markduped-bam - 数据类型: File - 必传: 是 ● 参数3 <ul style="list-style-type: none"> - 参数名称: know-site1 - 数据类型: File - 必传: 是 ● 参数4 <ul style="list-style-type: none"> - 参数名称: know-site2 - 数据类型: File - 必传: 是 ● 参数5 <ul style="list-style-type: none"> - 参数名称: know-site3 - 数据类型: File - 必传: 是
<p>输出参数</p>	<ul style="list-style-type: none"> ● 参数1 <ul style="list-style-type: none"> - 参数名称: out-dir - 数据类型: Directory - 必传: 是 - 默认值: /gatk4-tmp ● 参数2 <ul style="list-style-type: none"> - 参数名称: matrices-file - 数据类型: File - 必传: 是 - 默认值: /markduplicats.matrics ● 参数3 <ul style="list-style-type: none"> - 参数名称: markduped-bam - 数据类型: File - 必传: 是 - 默认值: /gatk4-markduped.bam 	<ul style="list-style-type: none"> ● 参数名称: recal-table ● 数据类型: File ● 必传: 是 ● 默认值: /gatk4-bqsr-recal.table

表 1-7 gatk-applybqsr、gatk-haplotypecaller 应用参数说明

应用名称和版本	<ul style="list-style-type: none"> 应用名称: gatk-applybqsr 版本: 4.1.9.0 	<ul style="list-style-type: none"> 应用名称: gatk-haplotypecaller 版本: 4.1.9.0
镜像和启动命令	<ul style="list-style-type: none"> 镜像: gatk-applybqsr:4.1.9.0 启动命令: gatk ApplyBQSR --bqsr-recal-file \${recal-table} --create-output-bam-index -R \${ref-file} -I \${markduped-bam} -O \${bqsr-bam} 	<ul style="list-style-type: none"> 镜像: gatk-haplotypecaller:4.1.9.0 启动命令: cd \${out-dir} && parallel -j 4 --xapply "gatk HaplotypeCaller -ERC GVCF -R \${ref-file} --pcr-indel-model NONE --tmp-dir \${out-dir} -I \${bqsr-bam} -O tmp_{1}.HC.vcf.gz --native-pair-hmm-threads 4 -L {1} " ::: `head -n-1 \${contig-file} cut -f1`
CPU、内存和GPU	<ul style="list-style-type: none"> CPU架构: X86 CPU需求: 3 Memory: 0.2 GPU类型: 无 GPU需求: 0 	<ul style="list-style-type: none"> CPU架构: X86 CPU需求: 3 Memory: 5 GPU类型: 无 GPU需求: 0
输入参数	<ul style="list-style-type: none"> 参数1 <ul style="list-style-type: none"> 参数名称: ref-file 数据类型: File 必传: 是 参数2 <ul style="list-style-type: none"> 参数名称: markduped-bam 数据类型: File 必传: 是 参数3 <ul style="list-style-type: none"> 参数名称: recal-table 数据类型: File 必传: 是 	<ul style="list-style-type: none"> 参数1 <ul style="list-style-type: none"> 参数名称: ref-file 数据类型: File 必传: 是 参数2 <ul style="list-style-type: none"> 参数名称: bqsr-bam 数据类型: File 必传: 是 参数3 <ul style="list-style-type: none"> 参数名称: contig-file 数据类型: File 必传: 是
输出参数	<ul style="list-style-type: none"> 参数名称: bqsr-bam 数据类型: File 必传: 是 默认值: /gatk4-bqsr.bam 	<ul style="list-style-type: none"> 参数名称: out-dir 数据类型: Directory 必传: 是 默认值: /gatk4

表 1-8 gatk-mergevcfs、discvrseq-variantqc 应用参数说明

应用名称和版本	<ul style="list-style-type: none"> 应用名称: gatk-mergevcfs 版本4.1.9.0 	<ul style="list-style-type: none"> 应用名称: discvrseq-variantqc 版本: 1.17
镜像和启动命令	<ul style="list-style-type: none"> 镜像: gatk-mergevcfs:4.1.9.0 启动命令: <pre>cd \${in-dir} && ls -R tmp*.vcf.gz > vcf.list && gatk MergeVcfs --TMP_DIR \${in-dir} -l vcf.list -O \${vcf-file} && rm -rf ./*</pre> 	<ul style="list-style-type: none"> 镜像: discvrseq-variantqc:1.17 启动命令: <pre>java -jar /DISCVRSeq.jar VariantQC --maxContigs `grep '>' \${ref-file} wc -l` --rawData \${json-file} --output \${html-file} --reference \${ref-file} --variant \${variants-file}</pre>
CPU、内存和GPU	<ul style="list-style-type: none"> CPU架构: X86 CPU需求: 0.5 Memory: 10 GPU类型: 无 GPU需求: 0 	<ul style="list-style-type: none"> CPU架构: X86 CPU需求: 0.1 Memory: 0.2 GPU类型: 无 GPU需求: 0
输入参数	<ul style="list-style-type: none"> 参数名称: in-dir 数据类型: Directory 必传: 是 	<ul style="list-style-type: none"> 参数1 <ul style="list-style-type: none"> 参数名称: ref-file 数据类型: File 必传: 是 参数2 <ul style="list-style-type: none"> 参数名称: variants-file 数据类型: File 必传: 是
输出参数	<ul style="list-style-type: none"> 参数名称: vcf-file 数据类型: File 必传: 是 默认值: /gatk4-HC.vcf.gz 	<ul style="list-style-type: none"> 参数1 <ul style="list-style-type: none"> 参数名称: json-file 数据类型: File 必传: 是 默认值: /gatk4-VariantQC.json 参数2 <ul style="list-style-type: none"> 参数名称: html-file 数据类型: File 必传: 是 默认值: /gatk4-VariantQC.html

1.6 搭建 NGS 流程

步骤1 登录医疗智能体平台，进入项目并选择“工具 > 流程”页签，单击“新建流程”。

图 1-8 新建流程

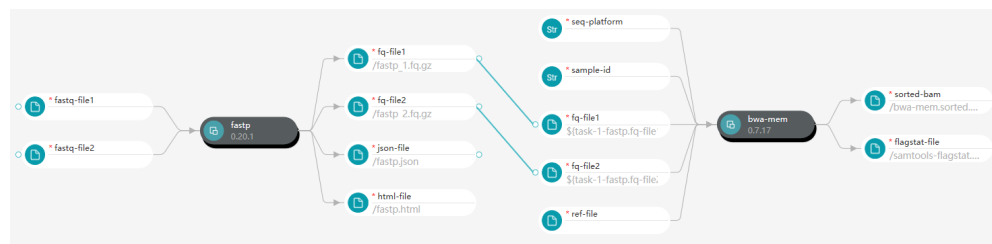


步骤2 在弹出的“流程设置”页面填写“流程名称”和“版本”，其他参数可选填。参数填写完成后，单击“确定”，完成流程设置。

步骤3 在流程设计器左侧应用列表中选择fastp、bwa-mem应用，并使用鼠标拖拽至画布中。

步骤4 将fastp的输出参数fq-file1、fa-file2和bwa-mem的输入参数fq-filefq-file2相连。

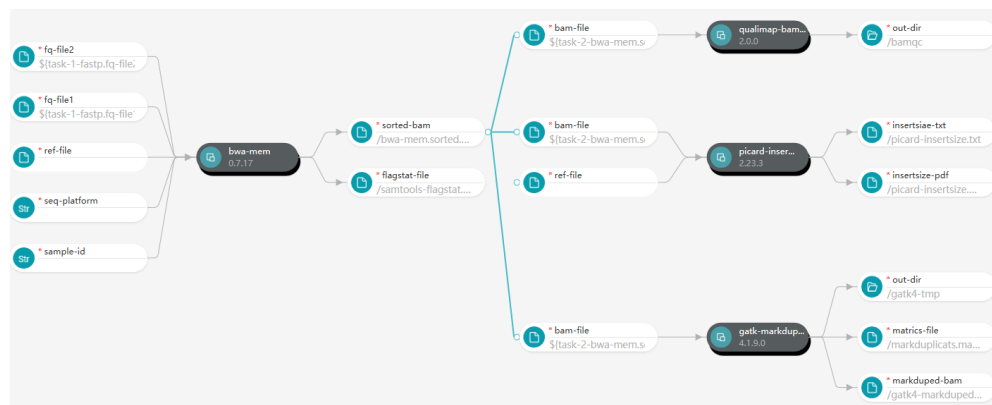
图 1-9 fastp、bwa-mem 连线关系



步骤5 在流程设计器左侧应用列表中选择qualimap-bamqc、picard-insertsize、gatk-markduplicates应用，并使用鼠标拖拽至画布中。

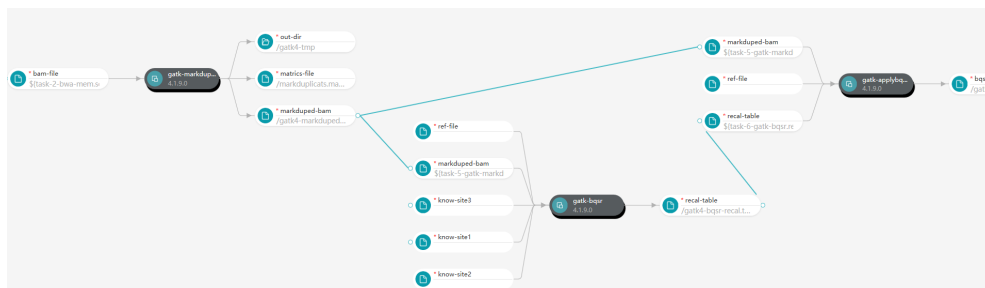
步骤6 将bwa-mem的输出参数sorted-bam与步骤4中三个应用的输入参数bam-file相连。

图 1-10 bwa-mem、qualimap-bamqc、picard-insertsize、gatk-markduplicates 连线关系



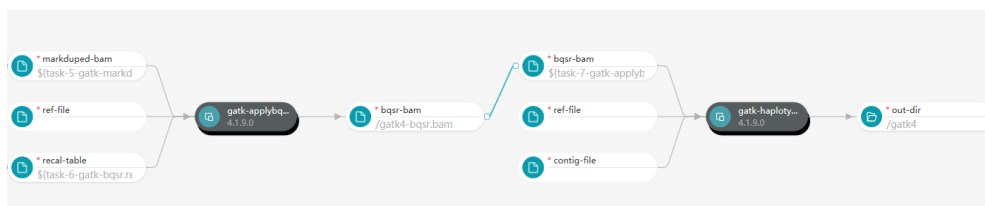
- 步骤7** 在流程设计器左侧应用列表中选择gatk-bqsr、gatk-applybqsr应用，并使用鼠标拖拽至画布中。
- 步骤8** 将gatk-markduplicates的输出参数markduped-bam与gatk-bqsr、gatk-applybqsr的输入参数markduped-bam相连。将gatk-bqsr的输出参数recal-table与gatk-applybqsr的输入参数recal-table相连。

图 1-11 gatk-markduplicates、gatk-bqsr、gatk-applybqsr 连线关系



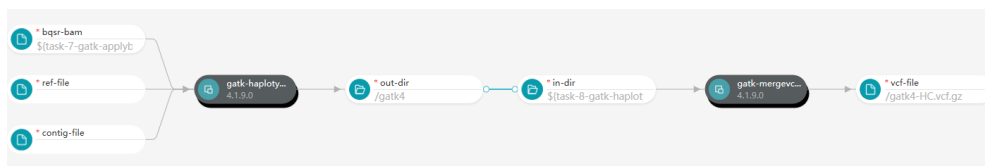
- 步骤9** 在流程设计器左侧应用列表中选择gatk-haplotypecaller应用，并使用鼠标拖拽至画布中。
- 步骤10** 将gatk-applybqsr的输出参数bqsr-bam与gatk-haplotypecaller的输入参数bqsr-bam相连。

图 1-12 gatk-applybqsr、gatk-haplotypecaller 连线关系



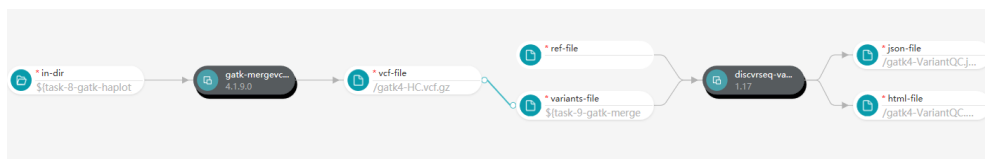
- 步骤11** 在流程设计器左侧应用列表中选择gatk-mergevcfs应用，并使用鼠标拖拽至画布中。
- 步骤12** 将gatk-haplotypecaller的输出参数out-dir与gatk-mergevcfs的输入参数in-dir相连。

图 1-13 gatk-haplotypecaller、gatk-mergevcfs 连线关系



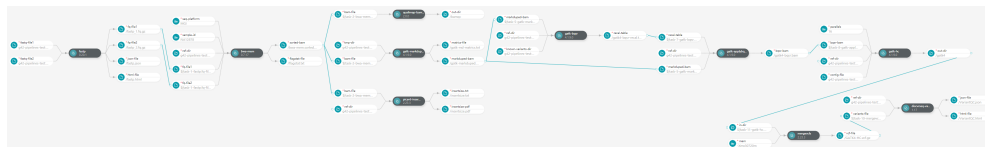
- 步骤13** 在流程设计器左侧应用列表中选择discvrseq-variantqc应用，并使用鼠标拖拽至画布中。
- 步骤14** 将gatk-mergevcfs的输出参数vcf-file与discvrseq-variantqc的输入参数variants-file相连。

图 1-14 gatk-mergevcfs、discvrseq-variantqc 连线关系



步骤15 搭建好的流程如下图所示。单击流程设计器上方保存按钮，保存流程。创建好的NGS将显示在“项目管理 > 工具”页签中。

图 1-15 NGS 流程



----结束

1.7 执行分析作业

创建分析作业

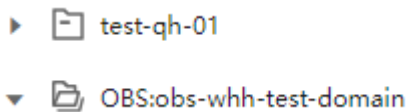
步骤1 登录医疗智能体平台，进入项目并选择“工具 > 流程”页签，单击NGS流程行的“启动作业”。

步骤2 请参考[配置输入和依赖数据](#)章节，设置NGS流程数输入数据。

步骤3 在新建作业页面，填写作业信息。

- 基本信息：包含作业名称、标签、描述。
- 输出路径：存放输出结果的路径，格式以/开头。例如项目中的output文件夹，输出路径可设置为/output。
不填写路径时，默认以“作业名-UUID”格式生成输出路径。
- 优先级：运行优先级，分为0~9级，优先级高的作业会被优先执行（该特性规划中，暂未上线）。
- 计算节点标签：作业会调度到含有相应标签的计算节点上。
当应用也配置了标签，如果应用和作业的计算节点标签在同一计算节点上，则应用调度至该计算节点上；应用和它的作业，不管节点标签是否一致，都会被调度到应用的节点标签所对应的计算节点上。
如果设置了不存在的计算节点标签，作业会进入等待，直至配置了相应的标签。
- 超时时间：作业运行时间超过设置时间时，认为超时，默认1440分钟，最大可设置为144000分钟，即作业运行至多100天。
- 加速类型：
加速效率：IO加速>本地盘加速>无
 - 无：作业运行于OBS中，不使用加速。
 - IO加速：IO加速使用弹性文件服务（SFS）提供高性能的数据读写，作业运行时，会将非最终结果的数据存储在SFS中用以提高任务运行效率，作业执行完成后会清理释放SFS空间。对于涉及频繁读写场景的任务建议开启IO加速，开启前需要先[购买性能加速](#)。
 - 本地盘加速：使用计算节点的本地盘进行加速。使用本地盘加速时，需保证购买的**计算节点**带有“数据盘”。OBS桶中的数据不支持本地盘加速，使用OBS桶中数据用于本地盘加速，可能会导致作业运行失败。

图 1-16 “OBS” 标签代表数据引用来源为 OBS 桶



步骤4 单击“确定”，保存作业信息。

----结束

配置输入和依赖数据

NGS流程中涉及的输入、输出和依赖数据如表1-9所示。配置数据前，请先参考[上传数据](#)，上传原始Fastq文件和依赖数据。

说明

如果在创建应用时打开了“并发”开关，可以设置多个参数值，批量执行作业。


数据上传完成后，在流程设计器页面，分别单击应用参数左侧  图标，设置输入和依赖数据。NGS流程中输入输出参数说明如表1-10所示。

表 1-9 流程输入、输出和依赖

类别	类型	说明
输入	Fastq	输入基于二代测序得到的原始Fastq文件，支持来源于多个barcode和路径的输入。
依赖	Reference Genome	输入的参考基因组序列，已经通过bwa构建了index。
依赖	Variant Sets	GATK4在做Variant Calling阶段需要输入的参考Variants数据集。
输出	FastQC Report	原始测序数据的质控报告，以HTML文件形式展示。
输出	BamQC Report	测序比对数据的质量控制报告，以HTML文件的形式展示。
输出	VCF	样本的突变信息，包含有SNP和INDEL信息，以VCF的格式存储。
输出	VCF Report	样本突变信息的质量控制报告，以HTML文件的形式展示。

表 1-10 参数说明

应用名称	参数	名称	类型	说明
fastp	输入参数	fastq-file1	file	二代测序fastq的Read1文件。
		fastq-file2	file	二代测序fastq的Read2文件。
	输出参数	fq-file1	file	Read1过滤之后输出fq.gz文件。
		fq-file2	file	Read2过滤之后输出fq.gz文件
		json-file	file	以JSON文件的格式输出的质控报告。
html-file	file	以HTML的格式输出易于阅读的质控报告。		
bwa-mem	输入参数	fq-file1	file	测序得到的fastq1文件。
		fa-file2	file	测序得到的fastq2文件。
		ref-file	file	参考基因组序列。
		seq-platform	string	测序平台，如MGI、Illumina。
		sample-id	string	文件前缀，如NA12878。
	输出参数	sorted-bam	file	比对和排序之后得到的bam文件。
		flagstat-file	file	基于bam做统计。
qualimap-bamqc	输入参数	bam-file	file	输入已经排序好的bam文件。
	输出参数	out-dir	directory	质控报告的输出目录。
picard-insertsize	输入参数	bam-file	file	经过比对和排序之后得到的bam文件。
		ref-file	file	参考基因组序列。
	输出参数	insertsize-txt	file	输出的insert size分布的文本文件。
		insertsize-pdf	file	输出的insert size分布的pdf文件。
gatk-markduplicates	输入参数	bam-file	file	输入比对之后经过sort的bam文件。
	输出参数	out-dir	directory	经过gatk-markduplicates处理之后得到的bam文件。
		matrices-file	file	质控报告文件。

应用名称	参数	名称	类型	说明
		markduped-bam	file	经过gatk-markduplicates处理之后得到的bam文件。
gatk-bqsr	输入参数	ref-file	file	参考基因组序列。
		markduped-bam	file	经过gatk-markduplicates处理之后得到的bam文件。
		know-site1	file	已知变异位点对应的vcf文件（其一）。
		know-site2	file	已知变异位点对应的vcf文件（其二）。
	know-site3	file	已知变异位点对应的vcf文件（其三）。	
	输出参数	recal-table	file	输出经过BQSR评估得到的参数文件。
gatk-applybqsr	输入参数	markduped-bam	file	经过gatk-markduplicates处理之后得到的bam文件。
		ref-file	file	参考基因组序列。
		recal-table	file	通过 GATK-BQSR得到参数评估文件。
		输出参数	bqsr-bam	file
gatk-haplotype caller	输入参数	bqsr-bam	file	经过gatk-applybqsr处理之后得到的bam文件。
		ref-file	file	参考基因组序列。
		contig-file	file	与参考基因组对应的contigs文件，包含contigs清单。
		输出参数	out-dir	directory
gatk-mergevcfs	输入参数	in-dir	directory	分interval进行Variant calling之后得到的vcf的list文件。
	输出参数	vcf-file	file	输出合并之后的Variant Calling的vcf文件。
discvrseq-variantqc	输入参数	ref-file	file	参考基因组序列。
		variants-file	file	变异检测软件（gatk4）生成的变异文件（vcf file）。
	输出参数	json-file	file	以JSON文件的格式输出的质控报告。
		html-file	file	以HTML文件的格式输出的质控报告。

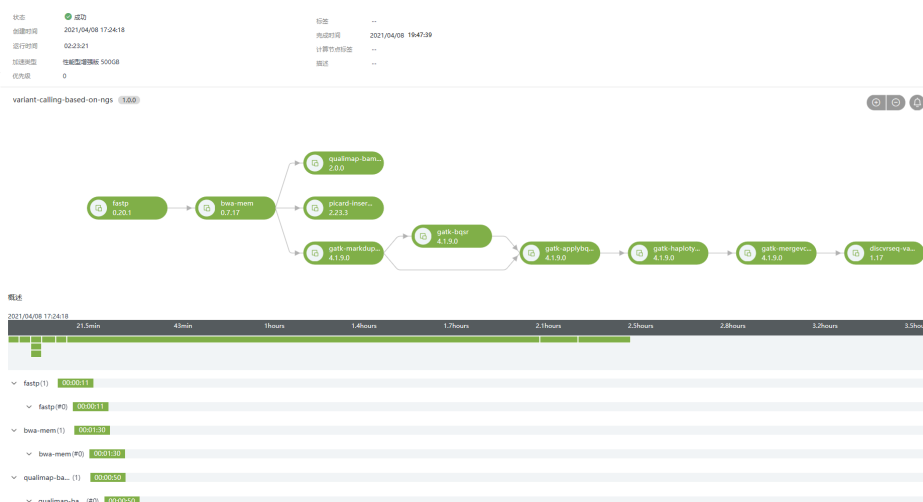
执行分析作业

单击流程设计器上方“新建作业”，弹出启动作业对话框，单击“确定”，即可运行。

查看分析结果

作业运行后，页面将跳转至“项目管理 > 作业”页面。您可以在该页面，查看作业的执行状态。单击作业名称即可进入作业详情页，查看运行进展和执行结果。

图 1-17 作业详情



单击“概述”列 ∨ 按钮，在展开的信息栏中查看作业的输入&输出、节点参数、应用，单击输入输出参数的路径，即可跳转至数据管理页面，查看相应数据。在作业的子任务中可以查看日志、事件。如果并发执行了多个作业，则会产生多个子任务。

图 1-18 信息栏

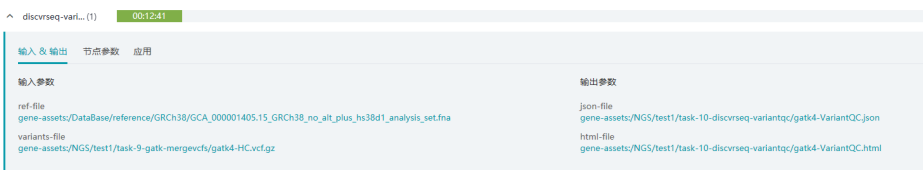
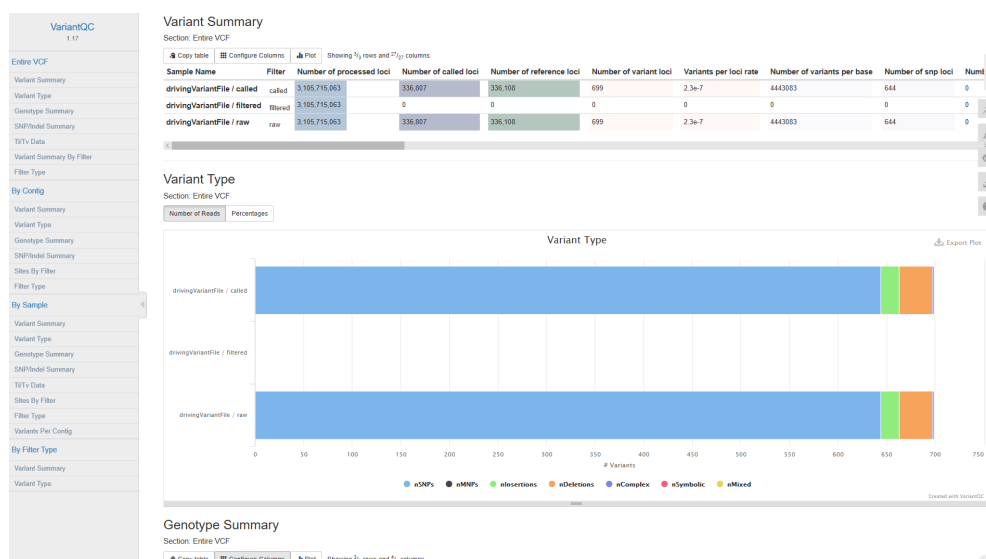


图 1-19 数据文件



图 1-20 质控报告



1.8 批量执行 NGS 分析

对于测序得到的大量数据，批量并自动执行NGS分析是提高工作效率的有效方式。

从搭建、执行NGS流程中可以看出，图形化的操作界面提供了友好、便捷的操作体验，但是当面临大批量的测序数据时，需要重复设置输入、输出、执行等步骤。为进一步提高NGS流程的执行效率，本章节介绍如何通过循环读取输入数据，批量运行NGS。同时，您也可以参考本示例，将批量运行的方法复制到其他分析任务中。

配置命令行工具

批量执行分析需要通过命令行工具完成。请参考[配置命令行工具](#)章节，下载命令行工具并完成配置。

获取 NGS 作业配置文件

编写NGS作业配置文件有两种方式，建议您使用第一种，通过获取已经执行成功的NGS配置文件，并在该配置文件基础上进行修改，得到可以用于批量执行NGS的配置文件。本示例介绍使用方法一获取配置文件的方法。

- 方式一
使用EIHealth平台完成NGS流程的搭建，并执行成功，然后在“分析作业”页面导出作业信息.yaml文件。
- 方式二
使用命令行工具完成NGS流程的搭建，进而获取相应的配置文件。详细的操作请参见[命令行工具](#)。

步骤1 使用switch命令进入NGS流程所在的项目。

例如，使用health switch project ngs-project命令进入到名为ngs-project的项目中。

步骤2 使用health get job命令获取该项目下所有的作业信息。

步骤3 查询NGS作业对应的job-id，使用health get job job-id命令获取NGS作业的信息。使用health get workflow命令查询NGS作业对应的workflow-id。获取到的作业信息如图1-21所示。

图 1-21 NGS 作业信息

```

"tasks": [
  {
    "task_name": "task-1-fastp", NGS作业中第一个任务为执行fastp分析
    "display_name": "",
    "output_dir": "",
    "whole_output_dir": "/ngs-test-7ef85a07-21b6-4fed-b18a-a5183b4b1b05/task-1-fastp",
    "resources": {
      "cpu": "0.1C",
      "memory": "0.1G",
      "gpu_type": "",
      "gpu": "0"
    },
    "inputs": [
      {
        "name": "fastq-file2", fastp的输入参数名称
        "values": [
          "ngs-project:/ngs/NA12878_0.R2.fastq.gz" 输入数据路径
                                                    格式为“项目名称/数据路径”
        ]
      },
      {
        "name": "fastq-file1",
        "values": [
          "ngs-project:/ngs/NA12878_0.R1.fastq.gz"
        ]
      }
    ],
    "app_info": {
      "app_id": "f723afd2-b350-42c1-ad43-c80f3c0d2808",
      "app_name": "fastp",
    }
  }
]

```

步骤4 health get job -s命令获取启动分析作业的模板。依据模板要求，将步骤3中获取到的NGS作业信息和workflow-id填充至模板中，修改好的配置文件示例请参见[NGS配置文件示例](#)。

请将该模板保存为.yaml格式至本地，并在本地完成模板修改。例如，命名为ngs.yaml。

----结束

编写执行脚本并提交作业

运行分析作业时，流程中的每一个应用称之为一个任务（Task），通过循环读取Task的输入数据，可以实现作业的批量执行。

例如，您可以在本地创建.bat格式的批处理文件，执行该脚本即可批量运行NGS分析作业。

```
@echo off

set list="task-1-fastp.fastq-file1=asset0331:/ngs/NA12878_0.R1.fastq.gz;task-1-fastp.fastq-file2=asset0331:/ngs/NA12878_0.R2.fastq.gz" "task-1-fastp.fastq-file1=asset0331:/ngs/NA12878_0.R1.fastq.gz;task-1-fastp.fastq-file2=asset0331:/ngs/NA12878_0.R2.fastq.gz"

health switch project ngs-project

for %%a in (%list%) do (
    echo %%a
    health create job -y D:\test\ngs.yaml -i %%a
    echo/
)
pause
```

图 1-22 批处理文件说明

```
1 @echo off
2
3 set list="task-1-fastp.fastq-file1=
  asset0331:/ngs/NA12878_0.R1.fastq.gz;task-1-fastp.fastq-file2=
  asset0331:/ngs/NA12878_0.R2.fastq.gz"
  "task-1-fastp.fastq-file1=
  asset0331:/ngs/NA12878_0.R1.fastq.gz;task-1-fastp.fastq-file2=
  asset0331:/ngs/NA12878_0.R2.fastq.gz"
4
5 health switch project ngs-project
6
7 for %%a in (%list%) do (
8     echo %%a
9     health create job -y D:\test\ngs.yaml -i %%a
10    echo/
11 )
12
13 pause
```

数据格式：
task名称.输入参数名称=项目名:/数据文件路径
不同的数据使用；分隔

执行第一次分析任务所需数据

执行第二次分析任务所需数据

说明

- 如果执行NGS批量任务时需要变更不同的原始数据、参考基因序列、测序平台、文件前缀等，请参考上述批处理文件示例，将需要变更的数据补充完整。
- .bat批处理文件需要和命令行工具放在同一路径下，同时，命令行工具需为登录状态。

NGS 配置文件示例

NGS作业由十个Task执行完成，本示例以fastp和bwa-mem两个Task为例，介绍.yaml文件填写规则，完整的NGS配置文件请参考本示例以及[获取NGS作业配置文件](#)章节得到的作业信息和模板填写。

```
job:
  name: ngs-test
  description ""
  priority: 0
  timeout: 1440
  output_dir: ""
  workflow_id: ngs-workflow::1.0.0:ngs-project
```

```
tasks:
- task_name: task-1-fastp
  inputs:
  - name: fastq-file1
    values:
    - 'ngs-project:/ngs/NA12878_0.R1.fastq.gz'
  inputs:
  - name: fastq-file2
    values:
    - 'ngs-project:/ngs/NA12878_0.R2.fastq.gz'
  resources:
    cpu: 0.1C
    memory: 0.1G
    gpu_type: ""
    gpu: '0'
- task_name: task-2-bwa-mem
  inputs:
  - name: fq-file1
    values:
    - '${task-1-fastp.fq-file1}'
  - name: fq-file2
    values:
    - '${task-1-fastp.fq-file2}'
  - name: ref-file
    values:
    - 'ngs-project:/ngs/GCA_000001405.15_GRCh38_no_alt_plus_hs38d1_analysis_set.fna'
  - name: seq-platform
    values:
    - 'MGI'
  - name: sample-id
    values:
    - 'NA12878'
  resources:
    cpu: 16C
    memory: 10G
    gpu_type: ""
    gpu: '0'
```

2 新型冠状病毒（COVID-19）虚拟药物筛选

2.1 虚拟药物筛选简介

虚拟药物筛选功能

医疗智能体平台支持根据靶点蛋白和小分子药物的3D结构，计算蛋白与药物之间的结合能量，进而预测小分子是否有成为候选药物的可能性。

虚拟药物筛选可实现如下功能。

- 整合分子对接结果，生成结合能矩阵。
- 整合受体与分子对接产生的配体构象，用于可视化展示。
- 对配体分子进行注释，包括DrugBank编号、分类、化学式、XLOGP3、TPSA、靶点、Csp3比例、分子量、可旋转键数目。

功能演示请参见[视频帮助](#)。

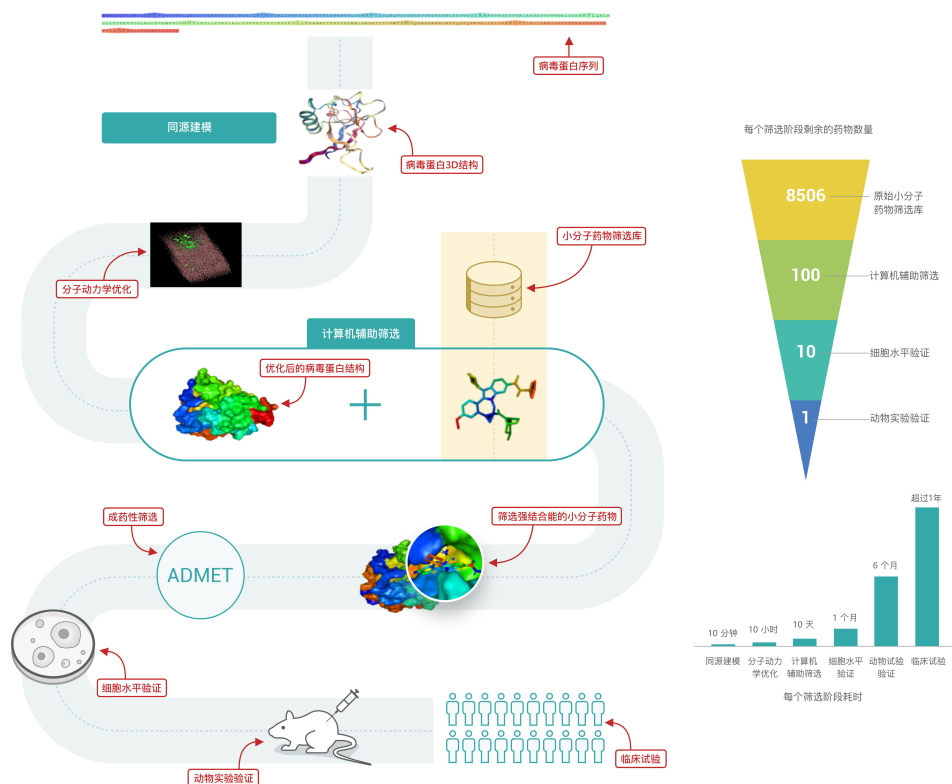
新型冠状病毒（COVID-19）虚拟药物筛选

新型冠状病毒（COVID-19）的出现在全球范围影响了人类健康，寻找有效治愈新冠肺炎的治疗方式是临床医生和药物研发人员最紧迫的工作。

为了全面、系统地评估药物对新冠病毒所有靶点蛋白的结合情况，华为云EI医疗智能体团队与华中科技大学同济医学院基础医学院、华中科技大学同济医学院附属武汉儿童医院、西安交通大学第一附属医院、中科院北京基因组研究所迅速成立联合团队，从新冠病毒蛋白序列开始，针对所有21个靶点蛋白进行同源建模、分子动力学模拟优化，获取靶点蛋白的3D结构，对超过8500个已上市、进入临床的小分子药物进行了约18万种药物-靶点配对情况的计算评估，让研究人员可以同时从21个蛋白的角度，综合、无偏地评估药物效果，从而为后续的药物机制研究、临床试验提供线索。

本案例介绍如何使用EIhealth平台虚拟药物筛选功能复现上述研究成果（<https://doi.org/10.1021/acs.jcim.0c00821>），并搭建虚拟药物筛选数据库。

图 2-1 药物筛选之旅



2.2 获取示例数据

本示例中使用小分子化合物SMILES结构式文件和蛋白3D结构PDB文件作为输入数据，可通过如下方式获取。

登录医疗智能体平台，在“资产市场”中订阅“docking summary测试数据”至所需的项目中。

图 2-2 示例数据



2.3 创建虚拟药物筛选任务

创建药物虚拟筛选任务

虚拟药物筛选可以使用资产市场中预置的“Docking Summary”流程对小分子化合物配体和蛋白受体进行对接。

使用步骤如下所示。

1. 登录医疗智能体平台，在“资产市场”中订阅“Docking Summary”流程至所需的项目中。
2. 进入“专题”页签，单击“新建研究”。

图 2-3 新建研究



3. 填写任务的基本信息，包括选择任务所属项目，研究的名称和描述。

图 2-4 基本信息



4. 选择配体分子和受体蛋白。
 - 作业名称：自定义名称。
 - 类型：选择小分子化合物。
 - 流程：选择从资产市场中订阅的“Docking Summary”流程。
 - 配体分子：配体分子文件，支持SMILES、SD SDF、PDBQT格式。
 - 受体蛋白：受体蛋白文件，支持PDB、PDBQT格式。


图 2-5 选择配体分子和受体蛋白

选择配体分子和受体蛋白

* 作业名称

* 类型 小分子化合物 多肽

* 流程 [进入资产市场订阅更多流程](#)

* 配体分子 
注：从obs中导入配体分子文件，支持的文件格式：SMILES，3D SDF 或 PDBQT

* 受体蛋白 
注：从obs中导入受体蛋白文件，支持的文件格式：PDB或PDBQT

5. 设置数据库。

数据库功能可以将任务运行过程中产生的数据文件按照模板生成数据库。

- 数据库模板：使用资产市场订阅的流程时，模板已预置，无需选择数据库模板。
- 数据库名称：数据库的名称。
- 输出文件格式：可以将流程生成的分子对接结果，保存为.txt、.csv或.vcf格式。使用“Docking Summary”流程时，保存格式为.txt。
- 相对路径：流程运行完成后，会按照流程子任务的名称生成数据文件，相对路径指按照哪个数据路径中的结果文件生成数据库。

对于“Docking Summary”流程，包含5个子任务，默认在task-5-docking summary中保存有汇总的数据文件。

- task-1-ligand 3dsdf to pdbqt：将配体的sdf文件转换为pdbqt文件。
- task-2-ligand smiles to 3dsdf：将配体的smiles文件转换为3dsdf文件。
- task-3-receptor pdb to pdbqt：将受体的pdb文件转换为pdbqt文件。
- task-4-qvina-w：分子对接。
- task-5-docking summary：汇总分子对接结果。

图 2-6 数据路径和流程图

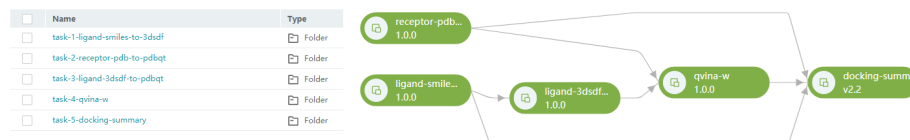
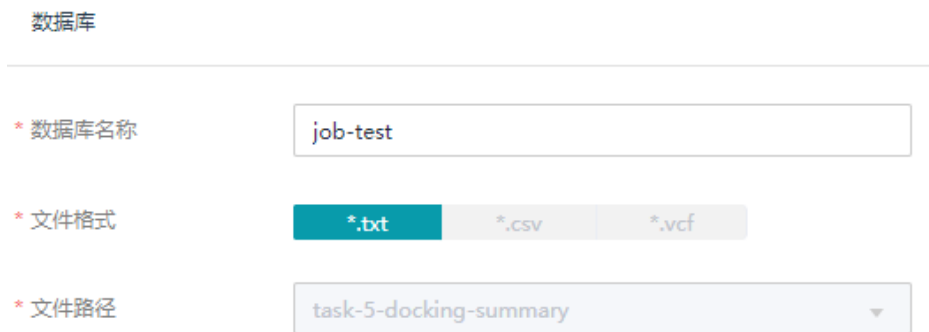
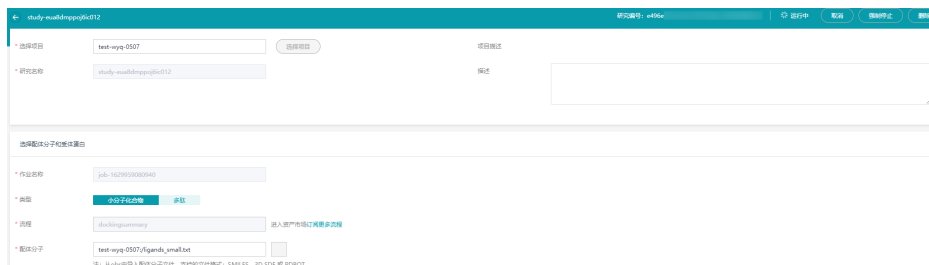


图 2-7 设置数据库



6. 设置完成后，单击“提交”，执行药物虚拟筛选任务。
对于“运行中”的任务，允许取消、强制停止或删除。

图 2-8 运行状态



查看执行结果

药物和蛋白结合能通过热点图呈现，并统计出结合能的均值和标准差。单击热点图，可查看详细3D分子对接结果、结合能排序、药物注释信息。

同时，基于预置的数据库模板，生成包含药物名称、结合能等信息的数据库。

图 2-9 结合能图

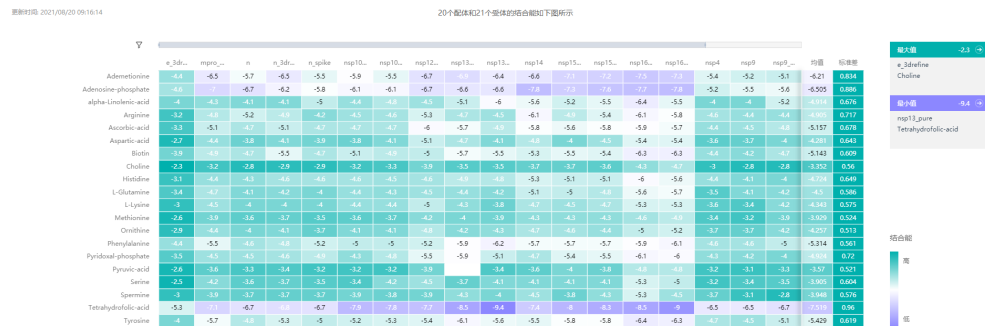


图 2-10 对接结果

