High Performance Computing Techniques for Big Data Processing

Mikailov, Mike, FDA/CDRH; Li, Weizhe, FDA/CDRH; Petrick, Nicholas, FDA/CDRH; Guo, Yan, FDA/CDER; Xu, Lei, FDA/CDER;
Weaver, James, FDA/CDER; Hyland, Paula, FDA/CDER; Luo, Fu-Jyh, FDA/CDRH
U.S. Food and Drug Administration

Abstract

In their mission to protect and promote public health, scientists at the FDA increasingly rely on innovative techniques on High Performance Computing (HPC) platforms for processing exponentially growing data in Artificial Intelligence (AI), Machine Learning (ML), Deep Learning (DL), Bioinformatics, Modeling & Simulation. Thousands of CPUs can be marshaled to process data on a scale and at speeds unthinkable in the recent past. However, the traditional techniques for processing large datasets are not adequate anymore and may overwhelm even massively parallel super computers – HPC clusters.

Introduction

This work presents novel techniques to overcome the problems associated with traditional computational techniques through efficient parallelization of data and computing and reduce the overall computation time needed for processing large datasets in **digital pathology** and **next generation sequencing (NGS) applications**.

- Digital pathology whole-slide images (WSIs) are large-size gigapixel images and image analysis based on deep learning AI technology involves pixel-wise testing of a trained deep learning neural network (DLNN) on hundreds of WSI images, which is time consuming. We take advantage of HPC facilities to parallelize this procedure. However, traditional software parallelization techniques and regular file formats can have significant scaling problems on HPC clusters. A novel algorithm is designed to localize and extract relevant patches in WSI files and group them in HDF5 files well suited for parallel I/O. HPC's array job facilities are adapted for hierarchical scaling and parallelization of WSI pre-processing and testing of trained algorithms.
- Numerous Bioinformatics software packages are used in assembly and alignment of many large NGS datasets, measured in hundreds of GBs. Traditional ways of conducting the research using these applications take advantage of the application-level built-in parallelization techniques such as POSIX multi-threading, OpenMP which are limited by the number of available CPU cores on a computing node. An approach is proposed to combine the application-level parallelization with distributed parallelization in processing many NGS datasets in parallel across the HPC cluster.

Materials and Methods

For digital pathology WSI image processing, images are partitioned into smaller subsets and grouped in HDF5 files. The job scheduler's, Son of Grid Engine's (SGE) array job of N tasks (see Figure 1) is formed and launched to partition N WSIs in parallel: each task processes one WSI and partition it into patches of a given size, then group the patches in an HDF5 file format. Then a lookup table is generated to associate the groups with the HDF5 files. The lookup table is used in the following stage (Figure 2) by the SGE array job (master job) to determine the number N of the HDF5 files and launch N tasks to process each HDF5 file in parallel. Every task in turn determines the number of groups G_i in the HDF5 file and launches another SGE array job of G_i tasks to process each group in parallel.

For NGS research, a flexible and scalable genome analysis pipeline (see Figure 3) is created to combine and launch a set of Bioinformatics applications in a predefined order. At every step of the pipeline M instances of the application are launched using an SGE array job of M tasks on the HPC cluster to process M NGS datasets in parallel. The pipeline order is maintained using SGE advanced job dependencies options "-*hold_jid*" and "-*hold_jid_ad*" which allows ordering tasks within neighboring steps.

Materials and Methods cont.

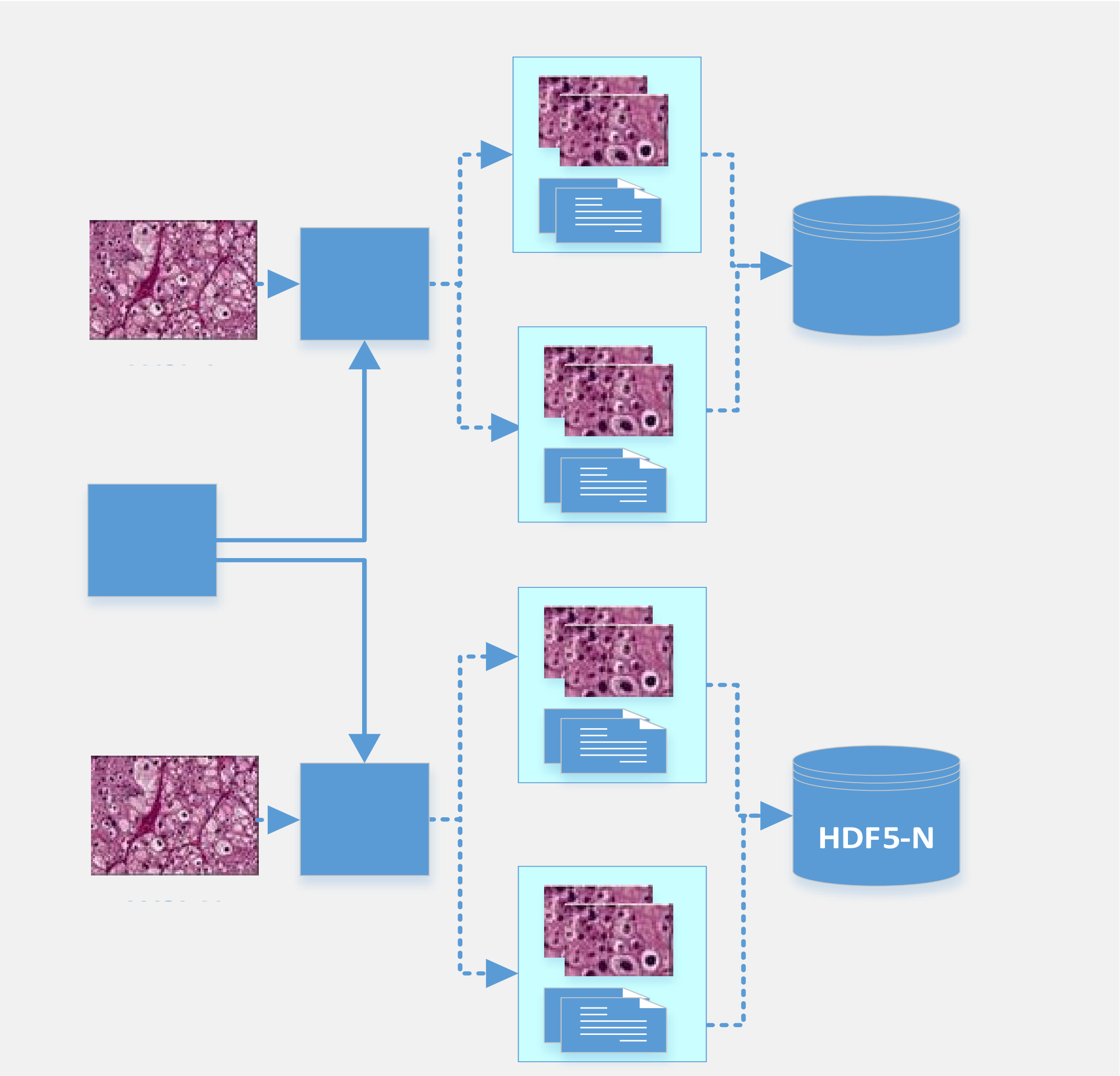
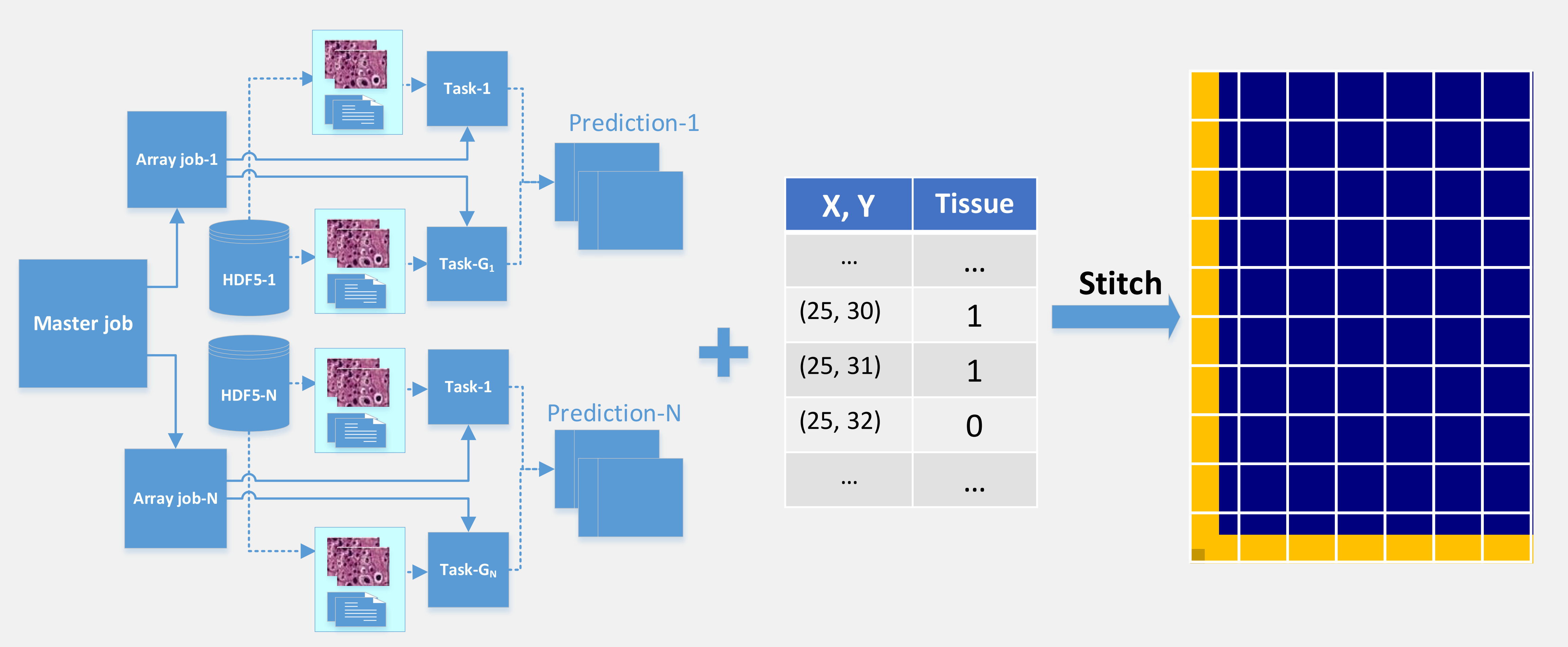


Figure 1. Extracting and grouping WSI patches in HDF5 files using independent/parallel array job tasks of the job scheduler



X, Y	Tissue
...	...
(25, 30)	1
(25, 31)	1
(25, 32)	0
...	...

Figure 2. Hierarchical scaling technique and Heatmap construction

Genome Analysis Pipeline

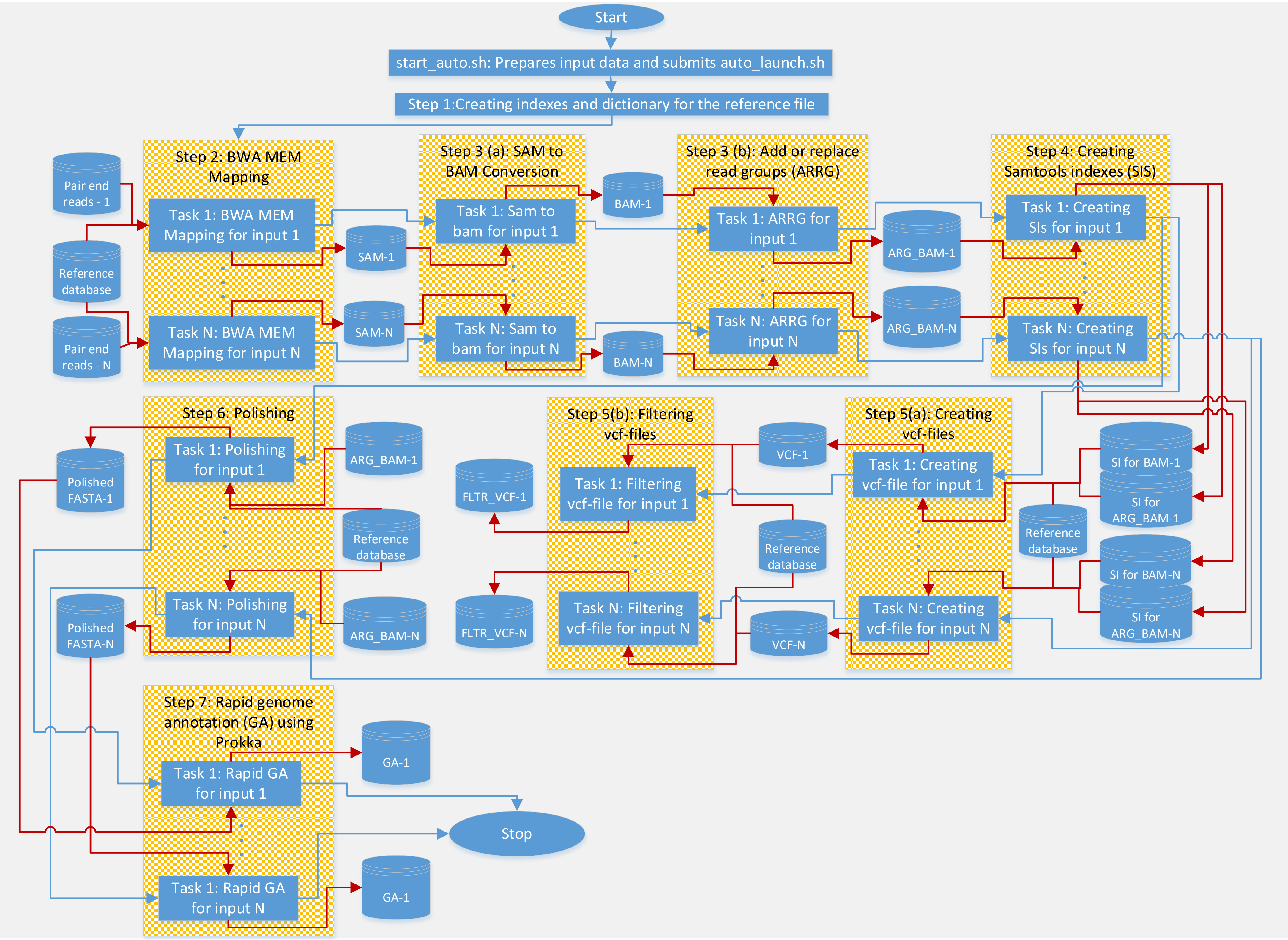

Start
start_auto.sh: Prepares input data and submits auto_launch.sh
Step 1:Creating indexes and dictionary for the reference file
Pair end reads - 1
Reference database
Pair end reads - N
Step 2: BWA MEM Mapping
Task 1: BWA MEM Mapping for input 1
Task N: BWA MEM Mapping for input N
SAM-1
SAM-N
Step 3 (a): SAM to BAM Conversion
Task 1: Sam to bam for input 1
Task N: Sam to bam for input N
BAM-1
BAM-N
Step 3 (b): Add or replace read groups (ARRG)
Task 1: ARRG for input 1
Task N: ARRG for input N
ARG_BAM-1
ARG_BAM-N
Step 4: Creating Samtools indexes (SIS)
Task 1: Creating SIs for input 1
Task N: Creating SIs for input N
SI for BAM-1
SI for ARG_BAM-1
SI for BAM-N
SI for ARG_BAM-N
Reference database
Step 5(a): Creating vcf-files
Task 1: Creating vcf-file for input 1
Task N: Creating vcf-file for input N
VCF-1
VCF-N
Reference database
Step 5(b): Filtering vcf-files
Task 1: Filtering vcf-file for input 1
Task N: Filtering vcf-file for input N
FLTR_VCF-1
FLTR_VCF-N
Step 6: Polishing
Task 1: Polishing for input 1
Task N: Polishing for input N
ARG_BAM-1
Reference database
ARG_BAM-N
Polished FASTA-1
Polished FASTA-N
Step 7: Rapid genome annotation (GA) using Prokka
Task 1: Rapid GA for input 1
Task 1: Rapid GA for input N
GA-1
GA-1
Stop

Figure 3. Scalable microbial genome analysis pipeline for Illumina NGS data processing

Results

Applying the methodologies for processing large data sets reduced the processing times from years to days.

- Processing the CAMELYON datasets containing 399 whole slide digital pathology images reduced the theoretical processing time of 18 years on a single CPU or 30 days on a single GPU to less than 45 hours on an HPC cluster of over 4,000 CPU cores.
- The scalable genome analysis pipeline implementation reduced a week of computation to less than 6 hours.

Conclusion

In this work, we demonstrated:

- The usefulness of our parallelization technique for processing large amount of image data in a high-performance computing environment.
- A genome analysis pipeline which combines shared-memory parallelization techniques with distributed-memory parallelization techniques in processing many NGS datasets in parallel and scalable manner.

Important benefits of our big data methodologies include broadening the range of investigations that can be performed in silico; increasing the speed of innovation; and potentially improving confidence in devices and drug regulatory decisions using novel evidence obtained through efficient big data processing.