
Algorithm 4: Partition of BAM file with overlapped blocks

Input: *overlapSize*

Output: *Splits*

```
1: splits  $\leftarrow$  Get the data blocks
2: splits are sorted by their file path and number
3: for split in splits do
4:   if split is not the last block then
5:     split.end  $\leftarrow$  split.end + overlapSize
6:   end if
7:   split.start  $\leftarrow$  the position of the first BAM record in the split
8:   split.end  $\leftarrow$  the end position of the last BGZF block in the split
9: end for
10: return splits
```

The algorithm description for acquiring overlapped data segments is shown as algorithm 4. The input parameter is the size of the overlapped area (*overlapSize*), which will affect the result of the subsequent variant calling. The experiment is conducted to evaluate it in detail in the later chapter. In the Algorithm 2, the data block information of the BAM file is obtained firstly and then data blocks are sorted according to the file path and the block number to ensure that the last block is at the last position of the link list and that an out-of-bounds error is avoided. Then the program traverses all the data blocks and except for the last data block, the rest need to be extended the size of overlapped area. The BAM file is compressed based on the BGZF (Blocked GNU Zip Format), which consists of many BGZF blocks. For each data block, the start position of the first BAM record is identified by modulus operation on BGZF blocks and other redundant information. To guarantee the integrity of the end of the expanded block, the algorithm shifts it slightly to cover the complete BGZF block. The method of determining the location comes from Hadoop-BAM. When finishing the processing, the program returns all the overlapped blocks.

