# MSPGraphBuilder: An disk-based de Bruijn graph constructor

**Sep 3, 2012**

**Version 0.1**

## Abstract

**MSPGraphBuilder is a disk-based software to build de Bruijn graph from DNA sequences.**

## 1. Synopsis

### (1) Typical:

```
java -jar BuildDeBruijnGraph_All.jar -in InputPath -k kmerLength -L readLength [-NB NumberOfBlocks] [-p MinimumSubstringLength] [-t threads] [-b bufferSize] [-r readable]
```

### (2) Step by step:

```
java -jar BuildDeBruijnGraph_Partition.jar -in InputPath -k kmerLength -L readLength [-NB NumberOfBlocks] [-p MinimumSubstringLength] [-b bufferSize]
```

```
java -jar BuildDeBruijnGraph _Map.jar -k kmerLength -NB NumberOfBlocks [-t threads] [-b bufferSize] [-c capacity]
```

```
sort -T TempDir -t $'\t' -o IDReplaceTable +0 -1 -n -m Maps/maps*
```

```
java -jar BuildDeBruijnGraph_Replace.jar -in IDReplaceTable -out OutGraph -k kmerLength -m largestID -L readLength [-b bufferSize] [-r readable]
```

## 2. Description

MSPGraphBuilder is a de Bruijn graph constructor based on the minimum substring partitioning technique. It will first partition the k-mers in DNA sequences into several disjoint partitions and compress consecutive k-mers to reduce I/O cost. Then it will assign each distinct k-mer a unique integer ID as its corresponding vertex ID in the de Bruijn graph and generate an ID replacement table for each partition. Finally it will merge all the replacement tables to generate one big global ID replacement table and finish building the global de Bruijn graph. The input of the MSPGraphBuilder is the raw short read sequences and the output of the MSPGraphBuilder is a sequence of id's mapped to the k-mers in short read sequences, in the same order (the duplicate k-mers will have the same id). MSPGraphBuilder is a disk-based method and thus can significantly reduce the memory consumption. It hashes k-mers in each partition individually, before loading all the partitions into memory. In contrast, the direct approach has to load all the k-mers into the memory and thus causing a very huge memory footprint.

To build de Bruijn graph with MSPGraphBuilder step by step, use the commands like:

```
java -jar BuildDeBruijnGraph_Partition.jar -in input.fasta -k 55 -L 101 -NB 256 –
p 6
```

```
java -jar BuildDeBruijnGraph_Map.jar -k 55 -NB 256 –t 8
```

```
sort -T /tmp -t $'\t' -o IDReplaceTable +0 -1 -n -m Maps/maps*
```

```
java -jar BuildDeBruijnGraph_Replace.jar -in IDReplaceTable -out OutGraph -k 55 -m
20000000 -L 101
```

These four commands will build the de Bruijn graph with the 55-mers in input.fasta. Specifically speaking, the first command will partition the short reads data input.fasta (whose average read length is 101) into 256 partitions using minimum substring partitioning, with the minimum substring length being 6; and the second command will hash the 55-mers and generate ID replacement tables in these 256 partitions with 8 threads; the third command will merge all the replacement tables to generate one big global ID replacement table; and the last command will do the ID replacements with respect to the global table and finish building the global de Bruijn graph.

## 3. Options

### 3.1 Partition

Function: Partition short reads data (in fasta format) using minimum substring partitioning

Usage: `java -jar BuildDeBruijnGraph_Partition.jar [options]`

Options Available:

```
[-help]: Print Help Information and Exit
[-in InputPath]: (String) Input Short Reads Data Path (Mandatory)
[-k k]: (Integer) K-mer Length, should be odd number smaller than 64 (Mandatory)
[-L readLength]: (Integer) Average Short Read Length (Mandatory)
[-NB numOfBlocks] :  (Integer) Number Of K-mer Blocks/Partitions. Default: 256
[-p pivotLength] :  (Integer) Minimum Substring Length. Default: 12
[-b bufferSize] :  (Integer) Read/Writer Buffer Size. Default: 8192
```

### 3.2 Map

Function: Assign each distinct k-mer a unique integer ID as its corresponding vertex ID in the de Bruijn graph and generate an ID replacement table for each partition.

Usage: `java -jar BuildDeBruijnGraph_Map.jar [options]`

Options Available:

```
[-help]: Print Help Information and Exit
[-k k]: (Integer) K-mer Length, should be odd number smaller than 64 (Mandatory)
```

**[-NB numOfBlocks] : (Integer) Number Of K-mer Blocks/Partitions. (Mandatory)**

**[-t numOfThreads] : (Integer) Number Of Threads. Default: 1**

**[-b bufferSize] : (Integer) Read/Writer Buffer Size. Default: 8192**

**[-c capacity] : (Integer) Hash Table Initial Capacity. Default: 1000000**

Note: The settings of ***k*** and ***NB*** should be in consistent with those in BuildDeBruijnGraph_Partition.

### 3.3 Sort&Merge

Function: Merge all the replacement tables to generate one big global ID replacement table.

Usage: `sort -T TempDir -t $'\t' -o IDReplaceTable +0 -1 -n -m Maps/maps*`

Arguments Available:

**[TempDir]: (String) Path for sorting temporaries (Mandatory)**

**[IDReplaceTable]: (String) Name of the final merged ID replacement table (Mandatory)**

### 3.4 Replace

Function: Do the ID replacements with respect to the global ID replacement table.

Usage: `java -jar BuildDeBruijnGraph_Replace.jar [options]`

Options Available:

**[-help]: Print Help Information and Exit**

**[-in InputPath]: (String) Input Global ID Replacement Table Path (Mandatory)**

**[-out OutputPath]: (String) Output De Bruijn Graph Path (Mandatory)**

**[-k k]: (Integer) K-mer Length, should be odd number smaller than 64 (Mandatory)**

**[-m largestID]: (Long) Largest ID assigned in Partition step (Mandatory)**

**[-L readLength]: (Integer) Average Short Read Length (Mandatory)**

**[-b bufferSize] : (Integer) Read/Writer Buffer Size. Default: 8192**

**[-r readable] : (Boolean) Output Format: *true* means readable text, *false* means binary. Default: false**

Note: (1) The settings of ***k*** and ***L*** should be in consistent with those in BuildDeBruijnGraph_Partition.

(2) The value of ***largestID*** will be printed out by the BuildDeBruijnGraph_Partition module automatically.

## 4. Version

Version: 0.10.0

## 5. Bug Reports

For bugs or questions or comments, please write to *yangli* at *cs* dot *ucsb* dot *edu*

## 6. Copyright