**How to run this example?**

* If you are using the graphical interface, (1) choose the "NOSEP\_i" algorithm, (2) select the input file "contextNOSEP.txt", (3) set the minimum length constraint, the maximum length constraint, the minimum gap constraints, the maximum gap, the given minimum support threshold, respectively to 1, 15, 0, 3, 3 and (4) click "Run algorithm".
* If you want to execute this example from the command line, then execute one of following commands:

java -jar spmf.jar run NOSEP\_i contextNOSEP.txt 1, 15, 0, 3, 3 in a folder containing spmf.jar and the example input file contextNOSEP.txt

* If you are using the source code version of SPMF, launch the file "MainTestNOSEP\_i.java" in the package ca.pfv.SPMF.test.

**What is the input of the NOSEP\_i?**

The input is a sequence and five parameters that are set by the user:

the minimum length constraint *minlen*

the maximum length constraint *maxlen*

the minimum gap constraints *mingap*

the maximum gap constraints *maxgap*

the given minimum support threshold *minsup*

For example,we give a file like this:

@CONVERTED\_FROM\_TEXT

@ITEM=1=A

@ITEM=3=C

@ITEM=7=G

@ITEM=20=T

@ITEM=-1=|

1 -1 1 -1 7 -1 20 -1 1 -1 3 -1 7 -1 1 -1 3 -1 7 -1 3 -1 1 -1 20 -1 3 -1 20 -1 1 -1 -2

This sequence is provided as the file context NOSEP.txt in the SPMF distribution.

**What is the output of NOSEP?**

NOSEP is an *Apriori*-based mining algorithm that is able to count the exact occurrence of the patterns.

To explain more formally what is NOSEP, it is necessary to review some definition.

(Pattern With Gap Constraints and Sequence):Pattern **p** with gap constraints can be described as *p*1[ *min*1*, max*1 ]*p*2 ··· [ *minm*−1*, maxm*−1 ]*pm*, where *pj* ∈*∑*, *minj* and *maxj* are two nonnegative integers and represent the minimum gap constraint and maximum gap constraint, respectively, 1 ≤ *j* ≤ *m*, and *∑* is a set of all event items.

(Occurrence): A group of m integers L = <*l*1*, l*2*,... lm*> is called an occurrence of **p** in **s**, if and only if 1 ≤*l*1 < *l*2 < ··· < *lm* ≤ n, *minj* ≤*l*j+1 − *l*j − 1 ≤ *maxj*, *p*1 = *sl*1 ,*p*2 = *sl*2 ,. . . , and *pm* = *slm* .

(Length Constraints):The length constraints can be written as len = [*minlen, maxlen*], where *minlen* and *maxlen* are the minimum length constraint and the maximum length constraint, respectively. If *L* = <*l*1*, l*2*,... lm*> satisfies *minlen* ≤ *lm* − *l*1 + 1 ≤ *maxlen*, then *L* is an occurrence with length constraints.

(Non-Overlapping Occurrence Set and Support)*:* Let *L* = <*l*1*, l*2*,... lm*> and *L*' = <*l*'1*, l*'2*,... l'm*> be two occurrences. If and only if ∀1 ≤ *j* ≤ *m* : *lj* = *l*'*j*, *L* and *L*' are two nooverlapping occurrences. If any two occurrences in a set are nonoverlapping, then the set is called nonoverlapping occurrence set. The support of **p**in **s**under the nonoverlapping condition, which is denoted by *sup*(**p***,* **s**), is the size of the maximum nonoverlapping occurrence set.

(Frequent Pattern and NOS):If the support of pattern **p**in sequence **s**or in sequence database SDBis no less than the given minimum support threshold *minsup*, then pattern **p**is called the frequent pattern. The goal of NOS is to mine all the frequent patterns with the gap and length constraints in sequence **s**or in sequence database SDB.

(Nettree): Nettree is similar to a tree data structure, consisting of root, leaf, level, parent, child, and so on. Nevertheless, Nettree has three characteristics that are evidently different from the tree structure.

1) A Nettree may have *n* roots, where *n* > 1.

2) To describe a node effectively, node *i* in the *jth* level is denoted by *nij* since the same node label can occur on different levels.

3) Any node except the root may have more than one parent and all its parents must be at the same level; that is the nonroot node *nij* (*j* > 1) may have multiple parents{*ni*1*j*−1, *ni*2*j*−1,..., *nimj*−1} (*m* ≥ 1), and thus there may be multiple paths from a node to a root node.

**Input file format:**

The **input file format** is defined as follows. It is a data preprocessed file.

**Output format**

The **output format** is defined as follows.

if set *minlen*=1, *maxlen*=15, *mingap*=0, *maxgap*=3 and *minsup*=3, we find result:

minlen:1 , maxlen:15

mingap:0 , maxgap:3

minsup:3

max: 16 lines: 7

1 3 7 20

1,3 1,7 3,1 7,1 7,3

1,7,3 7,1,3 7,3,1

The number of frequent patterns:12

The time-consuming:7ms.

The number of calculation:25

**Performance**

NOSEP not only meets the Apriori property but also is a complete algorithm. It employs an effective algorithm to completely calculate the support and also adopts an effective pattern growth approach to effectively reduce the candidate patterns. (see paper for more detail on experimental result)