Utiliser wincopper pour extraire des motifs séquentiels

L'objectif de ce notebook est de montrer l'utilisation de wincopper. Les algorithmes et structures sont décrites dans *Alatrista-Salas, H., Guevara-Cogorno, A., Maehara, Y. Nunez-del-Prado, M. (2020). Efficiently Mining Gapped and Window Constraint Frequent Sequential Patterns. Proceedings 17th International Conference on Modeling Decisions for Artificial Intelligence (https://link.springer.com/chapter/10.1007/978-3-030-57524-3_20 (https://link.springer.com/chapter/10.1007/978-3-030-57524-3_20))*.

Wincopper offre la possibilité outre d'extraire les motifs, de pouvoir prendre en compte des contraintes de temps. Il propose également une implémentation de prefixspan. Le lien github est ici : https://github.com/bitmapup/prefixspanr/ (https://github.com/bitmapup/prefixspanr/)

Attention: la version actuelle est en Python 2. Une version en Python 3 devrait bientôt être mise en place.

Installation

Avant de commencer, il est nécessaire de déjà posséder dans son environnement toutes les librairies utiles. Dans la seconde cellule nous importons toutes les librairies qui seront utiles à ce notebook. Il se peut que, lorsque vous lanciez l'éxecution de cette cellule, une soit absente. Dans ce cas il est nécessaire de l'installer. Pour cela dans la cellule suivante utiliser la commande :

! pip install nom_librairie

Attention : il est fortement conseillé lorsque l'une des librairies doit être installer de relancer le kernel de votre notebook.

Remarque : même si toutes les librairies sont importées dès le début, les librairies utiles pour des fonctions présentées au cours de ce notebook sont ré-importées de manière à indiquer d'où elles viennent et ainsi faciliter la réutilisation de la fonction dans un autre projet.

Attention : ici il faut impérativement installer les librairies associées à wincopper

In [3]:
```
# utiliser cette cellule pour installer les librairies manquantes
# pour cela il suffit de taper dans cette cellule : !pip install no
m_librairie_manquante
# d'exécuter la cellule et de relancer la cellule suivante pour voi
r si tout se passe bien
# recommencer tant que toutes les librairies ne sont pas installées
...


!pip install git+https://github.com/bitmapup/prefixspanr.git


# eventuellement ne pas oublier de relancer le kernel du notebook
```

```
Collecting git+https://github.com/bitmapup/prefixspanr.git
  Cloning https://github.com/bitmapup/prefixspanr.git to /tmp/pip-
req-build-ebomUQ
  Running command git clone -q https://github.com/bitmapup/prefixs
panr.git /tmp/pip-req-build-ebomUQ
Requirement already satisfied: wheel in /usr/local/lib/python2.7/d
ist-packages (from wincopper==1.0.2) (0.37.0)
Requirement already satisfied: pandas in /usr/local/lib/python2.7/
dist-packages (from wincopper==1.0.2) (0.24.2)
Requirement already satisfied: psutil in /usr/local/lib/python2.7/
dist-packages (from wincopper==1.0.2) (5.4.8)
Requirement already satisfied: numpy in /usr/local/lib/python2.7/d
ist-packages (from wincopper==1.0.2) (1.16.4)
Collecting resource
  Downloading https://files.pythonhosted.org/packages/34/ad/9cd037
c01c075f9a273c23557f8e71195d773d59d3881bbb26011d396c8b/Resource-0.
2.1-py2.py3-none-any.whl
Requirement already satisfied: pytz>=2011k in /usr/local/lib/pytho
n2.7/dist-packages (from pandas->wincopper==1.0.2) (2018.9)
Requirement already satisfied: python-dateutil>=2.5.0 in /usr/loca
l/lib/python2.7/dist-packages (from pandas->wincopper==1.0.2) (2.5
.3)
Collecting JsonForm>=0.0.2
  Downloading https://files.pythonhosted.org/packages/e0/bf/33b12c
4e1382804b2a1ffa3ed4a28b1c0208dbd2321665bc6bbf7007e8f9/JsonForm-0.
0.2-py2-none-any.whl
Collecting JsonSir>=0.0.2
  Downloading https://files.pythonhosted.org/packages/5d/3a/61e8f6
7d968d80327375e504996f6b05d4f61864e8b4c6ef80b690e23da7/JsonSir-0.0
.2-py2-none-any.whl
Collecting python-easyconfig>=0.1.0
  Downloading https://files.pythonhosted.org/packages/b1/86/113808
1cca360a02066eedaf301d0f358c35e0e0d67572acf9d6354edca9/Python_Easy
Config-0.1.7-py2.py3-none-any.whl
Requirement already satisfied: six>=1.5 in /usr/local/lib/python2.
7/dist-packages (from python-dateutil>=2.5.0->pandas->wincopper==1
.0.2) (1.15.0)
Requirement already satisfied: jsonschema in /usr/local/lib/python
2.7/dist-packages (from JsonForm>=0.0.2->resource->wincopper==1.0.
```

```
2) (2.6.0)
Requirement already satisfied: PyYAML in /usr/local/lib/python2.7/
dist-packages (from python-easyconfig>=0.1.0->resource->wincopper=
=1.0.2) (3.13)
Requirement already satisfied: functools32; python_version == "2.7
" in /usr/local/lib/python2.7/dist-packages (from jsonschema->Json
Form>=0.0.2->resource->wincopper==1.0.2) (3.2.3.post2)
Building wheels for collected packages: wincopper
  Building wheel for wincopper (setup.py) ... done
  Created wheel for wincopper: filename=wincopper-1.0.2-cp27-none-
any.whl size=27962 sha256=ec9db1e543a50285115c6364459bb6ca9cd9e4db
f451f14933c0ab5c39218eaf
  Stored in directory: /tmp/pip-ephem-wheel-cache-76PSUw/wheels/6c
/9a/3a/94618f57b0781fe88835ffb90acac13e247b83367f0c3ebc75
Successfully built wincopper
Installing collected packages: JsonForm, JsonSir, python-easyconfi
g, resource, wincopper
Successfully installed JsonForm-0.0.2 JsonSir-0.0.2 python-easycon
fig-0.1.7 resource-0.2.1 wincopper-1.0.2
```

In [4]:
```
# Importation des différentes librairies utiles pour le notebook

#Sickit learn met régulièrement à jour des versions et
#indique des futurs warnings.
#ces deux lignes permettent de ne pas les afficher.
import warnings
warnings.filterwarnings("ignore", category=FutureWarning)

# librairies générales

import pandas as pd
import numpy as np
import re
import sys
import psutil

from ast import literal_eval

# pour wincopper
import wincopper as wc
```

Pour pouvoir sauvegarder sur votre répertoire Google Drive, il est nécessaire de fournir une autorisation. Pour cela il suffit d'éxecuter la ligne suivante et de saisir le code donné par Google.

In [ ]:
```
# pour monter son drive Google Drive local
from google.colab import drive
drive.mount('/content/gdrive')
```

```
Mounted at /content/gdrive
```

Corriger éventuellement la ligne ci-dessous pour mettre le chemin vers un répertoire spécifique dans votre répertoire Google Drive :

In [ ]:
```
my_local_drive='/content/gdrive/My Drive/Colab Notebooks/ML_FDS'
# Ajout du path pour les librairies, fonctions et données
sys.path.append(my_local_drive)
# Se positionner sur le répertoire associé
%cd $my_local_drive

%pwd
```

Utilisation de wincopper

Télécharger le fichier dataset_wincopper.csv dans votre répertoire courant. Pour cela, décommenter la cellule suivante :

In [5]:
```
#!wget http://www.lirmm.fr/~poncelet/Ressources/dataset_wincopper.csv
```

```
--2021-09-23 09:23:28--  http://www.lirmm.fr/~poncelet/Ressources/
dataset_wincopper.csv
Resolving www.lirmm.fr (www.lirmm.fr)... 193.49.104.251
Connecting to www.lirmm.fr (www.lirmm.fr)|193.49.104.251|:80... co
nnected.
HTTP request sent, awaiting response... 200 OK
Length: 214 [text/csv]
Saving to: ‘dataset_wincopper.csv’

dataset_wincopper.c 100%[===================>]     214  --.-KB/s
in 0s

2021-09-23 09:23:29 (25.9 MB/s) - ‘dataset_wincopper.csv’ saved [2
14/214]
```

Lecture et conversion du fichier :

In [12]:
```
# les items dans le fichier exemple sont sous la forme de string, i
.e. 'a', il faut les convertir en literal
data = pd.read_csv("dataset_wincopper.csv", sep=",", header=0, conv
erters={"sequence": literal_eval})
display(data)
```

	sid	sequence
0	10	[a, [a, b, c], [a, c], d, [c, f]]
1	20	[[a, d], c, [b, c], [a, e]]
2	30	[[e, f], [a, b], [d, f], c, b]
3	40	[e, g, [a, f], c, b, c]

Récupération des identifiants de séquences (*sid*) et des séquences dans des listes.

In [45]:
```
sids = list(data["sid"])
sequences = list(data["sequence"])
```

Utilisation de PrefixSpan

Prefixspan manipule des listes d'itemsets. Il est possible de la spécifier dans les options.

Pour utiliser PrefixSpan, il existe différentes options :

- '*threshold*' (support minimal) avec '*threshold*' *: int or float*. Si threshold est un entier Wincopper considère qu'il s'agit d'un support absolu alors que pour un float il s'agit du support relatif.
- '*items_separated*' avec '*items_separated*'= *False or True*. Si *itemsSeparated* = *True* les patterns ne peuvent contenir seulement que des 1-itemsets. Si *itemsSeparated* = *False* les patterns seront composés de k-itemsets (valeur par défaut).
- '*window*': taille de fenêtre maximale entre itemsets
- '*gap*': gap maximum entre itemsets

Attention: Prefixspan considère des listes d'itemsets donc *itemsSeparated* = *False*

In [41]:
```
# spécification du support minimal en entier (donc par rapport aux
nombres de séquences de la base)
threshold = 3
# contient k-itemsets séparés par des virgules
items_separated = False

options = {'threshold': threshold, 'itemsSeparated': items_separate
d}
```

Il suffit alors d'appeler la méthode *prefixspan* avec la liste des options.

In [42]:
```
result_mining = wc.prefixspan(sequences, options)

for pat in result_mining:
    print(pat)
```

```
['<a>', 4, 1.0]
['<a><b>', 4, 1.0]
['<a><c>', 4, 1.0]
['<a><c><b>', 3, 0.75]
['<a><c><c>', 3, 0.75]
['<b>', 4, 1.0]
['<b><c>', 3, 0.75]
['<c>', 4, 1.0]
['<c><b>', 3, 0.75]
['<c><c>', 3, 0.75]
['<d>', 3, 0.75]
['<d><c>', 3, 0.75]
['<e>', 3, 0.75]
['<f>', 3, 0.75]
```

Avec d'autres contraintes :

In [31]:
```
# spécification du support minimal en entier (donc par rapport aux
nombres de séquences de la base)
threshold = 3
# contient k-itemsets séparés par des virgules
items_separated = False
# max window de 1, il faut 1 au max entre les itemsets
maxwindow=1
options = {'threshold': threshold, 'itemsSeparated': items_separate
d, 'window':maxwindow}
result_mining = wc.prefixspan(sequences, options)

for pat in result_mining:
    print(pat)
```

```
['<a>', 4, 1.0]
['<a><c>', 3, 0.75]
['<b>', 4, 1.0]
['<c>', 4, 1.0]
['<c><b>', 3, 0.75]
['<d>', 3, 0.75]
['<d><c>', 3, 0.75]
['<e>', 3, 0.75]
['<f>', 3, 0.75]
```

Utilisation de wincopper

Wincopper propose de pouvoir utiliser différentes contraintes qui peuvent être précisées dans les options :

- '*threshold*': support minimal en entier (support absolu) ou en float (support relatif)
- '*minSseq*': constrainte de taille minimale minimale des itemsets (itemset size)
- '*maxSseq*': contrainte de taille maximale des itemsets (itemset size)
- '*minSize*': contrainte de taille minimale des patterns (subsequence size)
- '*maxSize*': contrainte de taille maximale des patterns (subsequence size)
- '*logic*': contraite d'inclusion souple
  OR relation '(s1 | s2)'
  AND relation '(s1 & s2)'

In [57]:
```
# spécification du support minimal en entier (donc par rapport aux
nombres de séquences de la base)
threshold = 2
# contient k-itemsets séparés par des virgules
items_separated = False

# taille minimale des sous séquences
minseq = 2
# taille maximale des sous séquences
maxseq = 2
options = {'threshold': threshold, 'itemsSeparated': items_separate
d, 'maxSseq':maxseq,'minSseq':minseq}

result_mining = wc.prefixspan(sequences, options)

for pat in result_mining:
    print(pat)
```

```
['<a,b>', 2, 0.5]
['<b,c>', 2, 0.5]
```

Il existe d'autres options qui peuvent être utilisées :

- '*dataDesc*': permet de donner un nom au fichier résultat
- '*resultFile*': drapeau pour générer un fichier résultat. Si *resultFile=True* un fichier avec les patterns extrait est créé (défaut). Si *resultFile=False* aucun fichier n'est généré.
- '*test*': pour générer un résumé de fichier de tests (dans un but d'expérimentation). Si *test=True* le résumé est généré. Si *test=False* aucun fichier n'est généré (défaut).

In [64]:
```
# spécification du support minimal en entier (donc par rapport aux
nombres de séquences de la base)
threshold = 2
# contient k-itemsets séparés par des virgules
items_separated = False

# taille minimale des sous séquences
minseq = 2
# taille maximale des sous séquences
maxseq = 2
#
test=True
options = {'threshold': threshold, 'itemsSeparated': items_separate
d, 'maxSseq':maxseq,'minSseq':minseq,
          'test':True, 'dataDesc':'toto.txt'}

result_mining = wc.prefixspan(sequences, options)

for pat in result_mining:
    print(pat)
```

```
['<a,b>', 2, 0.5]
['<b,c>', 2, 0.5]
```