Manuel d'utilisation

Manuel d'utilisation

0) Prérequis sur le système

Disposer des packages « libreadline » (partie runtime uniquement)

à titre d'exemple, pour vérifier sous debian (ou ubuntu)

```
# dpkg -l libreadline8
(…)
||/ Nom                  Version      Architecture Description
+++-====================-============-============-
=================================================
ii  libreadline8:amd64    8.1-1        amd64        GNU readline and
history libraries, run-time libraries
```

Si vous souhaitez pouvoir utiliser l'affichage X11 pour afficher le graphe au format DOT, il vous faut

a. un affichage X opérationnel

➔ plusieurs solutions possibles

=> installer un serveur « X » quelque part (par exemple sur windows « vcXsrv » ou Xming)et autoriser son utilisation depuis le serveur d'exécution de « automate_ui »

=> installer « VNC » sur le serveur d'exécution, et un client « vncviewer » pour visualiser le'afiichage X11 à distance

=> etc.

à titre d'exemple, avec vnc (sur un serveur Linux/Debian)

Sur le serveur Débian, on peut installer « tighvncserver »

 # apt-get install tightvncserver »

Depuis l'utilisateur d'exécution

on lance le serveur X

$ vncserver :0

on entre un mot de passe pour l'accès « VNC »

on positionne la variable DISPLAY

$ export DISPLAY=:0

Vérifier que le serveur X est opérationne !

$ xdpyinfo

(vous obtenez des informations sur le type d'affichage)

En cas d'erreur, vous obtenez un mesqage dy type « **xdpyinfo: unable to open display ":0".** »

(NB le port d'écoute vnc est indiqué dans le fichier « $HOME/.vnc/*

$ grep 'Listening' ${HOME}/.vnc/*${DISPLAY}.log

```
$ grep 'Listening' ${HOME}/.vnc/*${DISPLAY}.log
05/01/22 10:54:30 Listening for VNC connections
on TCP port 5900 )
```

Sur un poste quelconque (windows par exemple), on peut utiliser « vncviewer.exe » par exemple pour visualiser le contenu de ce qui s'affiche sur le serveur X (en indiquant le port précédemment identifié, et en fournissant le mot de passe entré précédemment)

https://uvnc.com/component/jdownloads/send/0-/420-ultravnc-1-3-60-bin-zip.html?Itemid=0

b. Le produit « GraphViz » qui fournit l'exécutable « dot » utilisé pour affichage les fcihier au format « .dot »

```
# apt-get install graphviz
```

1) Mis en place du binaire « automate_ui ». Il est autonome (il dépend que de la librairie « libreadline8 » et de la commande « dot » pour la partei !xDot), il peut donc être installé/déployé où on le souhaite.

2) Exécution de « automate_ui » (on supposera l'installation dans un sous –répertoire « bin », pour l'exemple mais ce n'est pas une obligation),

→ un paramètre peut être passé pour charger dès le lancement un fichier automate…

```
$ ./bin/automaton_ui aut/word.aut
>>> Allowed characters =
[ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz]
LR1[current=aut/word.aut]:
```

Un prompt commençant par « LR1 » invite à entrer une chaîne (à vérifier dans l'automate) ou une commande.

Les commandes disponibles peuvent être obtenues en entrant directement « enter » (soumission d'une commande/chaîne vide).

```
$ ./bin/automaton_ui aut/word.aut
>>> Allowed characters =
[ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz]
LR1[current=aut/word.aut]:
!cd             Change to directory DIR.
!load           Load Automate from file.
!loadFile               Synonym for `load'.
!gDOT           Show Automate DOT graph.
!xDOT           Show Automate DOT graph (graphviz)².
!getDOT         Synonym for `gDOT'.
!chk            Check string to automate.
!check          Synonym for `chk'.
!help           Display this text.
!?              Synonym for `help'.
!list           List files in DIR.
!ls             Synonym for `list'.
!pwd            Print the current working directory.
!quit           Quit using LR1.
!rename         Rename FILE to NEWNAME.
!stat           Print out statistics on FILE.
!view           View the contents of FILE.
!LR1version             Printf the current version of LR1 automaton
program.
!version                Synonym of LR1version.
>>> Allowed characters =
[ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz]
LR1[current=aut/word.aut]:
```

Il est donc possible (en se limitant au fonction propre à l'automate de

i. Charger « !load 'fichier.aut' » ou « !loadFile 'fichier.aut' » un fichier automate (il viendra remplacer l'automate courant repris sur le prompt)

```
$ ./bin/automaton_ui aut/word.aut
>>> Allowed characters =
[ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz]
```

```
LR1[current=aut/word.aut]: !load aut/arith.aut
Load automate from `aut/arith.aut'
>>> Allowed characters = [ ()*+-/0123456789]
LR1[current=aut/arith.aut]:
```

ii. « !gDOT » ou « !getDOT » permettent d'obtenir un graphe au format « DOT » de l'automate courant

```
$ ./bin/automaton_ui aut/word.aut
>>> Allowed characters =
[ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz]
LR1[current=aut/word.aut]: !gDOT
digraph DOTaut {
start [ style=invis ];
start -> Q0 [ color=black];
Accepted [shape=none, fontcolor=green];
  "(0, S)" [shape=none];
  Q0 -> "(0, S)" [ color=royalblue1, fontcolor=royalblue1, label = "'\\n'"];
  Q0 -> Q1 [ color=black,  fontcolor=black,label = "A-Z,a-z"];
  "(0, S)" [shape=none];
  Q1 -> "(0, S)" [ color=royalblue1, fontcolor=royalblue1, label = "'\\n'"];
  Q1 -> Q1 [ color=black,  fontcolor=black,label = "A-Z,a-z"];
  "(2, S)" [shape=none];
  Q2 -> "(2, S)" [ color=royalblue1, fontcolor=royalblue1, label = "'\\n'"];
  Q3 -> Accepted [ color=green, fontcolor=green, label = "'\\n'"];
  Q0 -> Q3 [ color=red, fontcolor=red, label = "S"];
  Q1 -> Q2 [ color=red, fontcolor=red, label = "S"];
}
>>> Allowed characters =
[ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz]
```

iii. « !xDOT » permet d'afficher le graphe au format « DOT » de l'automate courant dans une fenêtre x11

```
$ ./bin/automaton_ui aut/word.aut
>>> Allowed characters =
[ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz]
LR1[current=aut/word.aut]: !xDOT
Execute : ( dot -Tx11 /tmp/.xdot.word.aut.2392; rm
/tmp/.xdot.word.aut.2392 )&
>>> Allowed characters =
[ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz]
LR1[current=aut/word.aut]:
```

ð Une fenêtre X11 est ouverte en tâche de fond sur un fichier temporaire au format « .dot » de l'automate

iv. « !chk chaîne » ou « !check chaîne » ou « chaîne » soumettent la chaîne à l'automate courant (inutile d'indiquer le caractère '\n' de fin de ligne il est implicitement ajouté)

```
$ ./bin/automaton_ui aut/word.aut
>>> Allowed characters =
[ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz]
LR1[current=aut/word.aut]: chaineatester
  Word accepted => "chaineatester"
```

```
>>> Allowed characters =
[ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz]
LR1[current=aut/word.aut]: !chk chaineatester
  Word accepted => "chaineatester"
>>> Allowed characters =
[ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz]
LR1[current=aut/word.aut]: !check chaineatester
  Word accepted => "chaineatester"
>>> Allowed characters =
[ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz]
LR1[current=aut/word.aut]: chaine2tester
  Word rejected => "chaine2tester\n"
               _______^
>>> Allowed characters =
[ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz]
```

v. Des commandes annexes (« cd » « pwd » « ls » « stat » ou « view ») permettent de se déplacer dans l'arborescence comme dans un « shell »…

3) Mis en place du binaire « get_DOT ». Il est lui aussi autonome , et peut être déployé où on le souhaite.

4) Exécution de « get_DOT ». Il pren en paramètre un fichier décrivant l'automate, et renvoie un contenu au format '.dot'.

```
$ ./bin/get_DOT aut/word.aut
digraph DOTaut {
start [ style=invis ];
start -> Q0 [ color=black];
Accepted [shape=none, fontcolor=green];
  "(0, S)" [shape=none];
  Q0 -> "(0, S)" [ color=royalblue1, fontcolor=royalblue1, label = "'\\n'"];
  Q0 -> Q1 [ color=black,  fontcolor=black,label = "A-Z,a-z"];
  "(0, S)" [shape=none];
  Q1 -> "(0, S)" [ color=royalblue1, fontcolor=royalblue1, label = "'\\n'"];
  Q1 -> Q1 [ color=black,  fontcolor=black,label = "A-Z,a-z"];
  "(2, S)" [shape=none];
  Q2 -> "(2, S)" [ color=royalblue1, fontcolor=royalblue1, label = "'\\n'"];
  Q3 -> Accepted [ color=green, fontcolor=green, label = "'\\n'"];
  Q0 -> Q3 [ color=red, fontcolor=red, label = "S"];
  Q1 -> Q2 [ color=red, fontcolor=red, label = "S"];
}
```