L'extension pour LaTeX

dijkstra

v 0.13

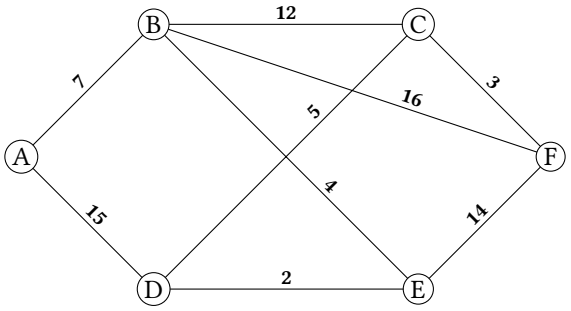
1 octobre 2022

Christian Tellechea
unbonpetit@netc.fr

Cette petite extension met en œuvre l'algorithme de Dijkstra pour des graphes pondérés, orientés ou non : le tableau de recherche du plus court chemin peut être affiché, la distance minimale entre deux sommets et le chemin correspondant sont stockés dans des macros.

1 Un exemple

Dans le graphe *non orienté* suivant, quel est le plus court chemin pour aller de A à F ?



Lire le graphe Pour trouver le plus court chemin pour aller de A à F, il faut d'abord lire le graphe. Comme il est fréquent que les graphes soient peu peuplés, j'ai pris le parti de définir un graphe par une liste d'adjacence. Ainsi, la macro `\readgraph`, qui va lire le graphe, admet comme argument obligatoire une liste d'adjacence :

```
\readgraph{
        A [B=7, D=15],
        B [C=12, E=4, F=16],
        C [D=5, F=3],
        D [E=2],
        E [F=14]
}
```

Les espaces sont ignorés de part et d'autre des noms des sommets, des crochets (ouvrants et fermants), des signes « = » et des virgules. Ainsi, ce n'est que dans les noms des sommets que les espaces ne sont pas ignorés : par exemple, le sommet « `A 1` » est distinct du sommet « `A1` ».

Conditions sur les distances Les distances entre sommets *doivent* être positives, c'est une limitation intrinsèque à l'algorithme de Dijkstra pour qu'il fonctionne sans erreur. La méthode de programmation utilisée dans cette extension exige de plus que ces distances soient des nombres *entiers*.

Une fois que le graphe a été lu, celui-ci est rendu *non orienté* en interne et donc en coulisses, la liste d'adjacence devient

```
A [B=7, D=15],
B [A=7, C=12, E=4, F=16],
C [B=12, D=5, E=3],
D [A=15, C=5, E=2],
E [D=2, B=4, F=14],
F [B=16, C=3, E=14]
```

Par conséquent, la liste d'adjacence entrée par l'utilisateur ne doit pas contenir d'incohérence. Si l'on spécifie la distance entre un sommet A et un sommet B par `A[B=`$\langle x\rangle$`,...]` on peut s'économiser la peine de spécifier cette même distance entre B et A puisque c'est fait par l'extension `dijkstra` automatiquement. En revanche, une erreur sera émise si dans la liste d'adjacence, on trouve `A[B=`$\langle x\rangle$`,...]` puis `B[A=`$\langle y\rangle$`,...]` où $\langle y\rangle$ et $\langle x\rangle$ sont différents.

Lancer l'algorithme Une fois que le graphe est lu par la macro `\readgraph`, on lance l'algorithme avec `\dijkstra{`$\langle A\rangle$`}{`$\langle B\rangle$`}` où $\langle A\rangle$ et $\langle B\rangle$ sont deux sommets du graphe. La distance minimale entre ces deux sommets est stockée dans la macro `\dijkdist` et le chemin correspondant dans `\dijkpath`.

```
\readgraph{
  A [B=7, D=15],
  B [C=12, E=4, F=16],
  C [D=5, F=3],
  D [E=2],
  E [F=14]}
Tableau : \dijkstra{A}{F}\par
Distance A–F = \dijkdist\par
Chemin = \dijkpath
```

Tableau :

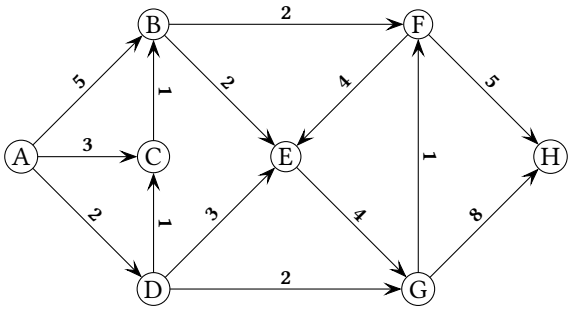
A	B	C	D	E	F
0	∞	∞	∞	∞	∞
—	$\mathbf{7_A}$	∞	15_A	∞	∞
—	—	19_B	15_A	$\mathbf{11_B}$	23_B
—	—	19_B	$\mathbf{13_E}$	—	23_B
—	—	$\mathbf{18_D}$	—	—	23_B
—	—	—	—	—	$\mathbf{21_C}$

Distance A-F = 21
Chemin = A-B-E-D-C-F

Dans le tableau, les colonnes sont disposées dans le *même ordre* que celui des sommets dans la liste d'adjacence lue par `\readgraph`.

2 Graphe orienté

Pour spécifier à `\readgraph` que la liste d'adjacence est celle d'un graphe *orienté*, la macro doit être suivie d'une étoile.



Cela donne

```
\readgraph*{
  A[B=5, C=3, D=2],
  B[E=2, F=2],
  C[B=1],
  D[C=1, E=3, G=2],
  E[G=4],
  F[E=4, H=5],
  G[F=1, H=8]}
Tableau : \dijkstra{A}{H}\par
Distance A-H = \dijkdist\par
Chemin = \dijkpath
```

Tableau :

A	B	C	D	E	F	G	H
0	∞	∞	∞	∞	∞	∞	∞
—	5_{A}	3_{A}	$\mathbf{2_{A}}$	∞	∞	∞	∞
—	5_{A}	$\mathbf{3_{A}}$	—	5_{D}	∞	4_{D}	∞
—	$\mathbf{4_{C}}$	—	—	5_{D}	∞	4_{D}	∞
—	—	—	—	5_{D}	6_{B}	$\mathbf{4_{D}}$	∞
—	—	—	—	$\mathbf{5_{D}}$	5_{G}	—	12_{G}
—	—	—	—	—	$\mathbf{5_{G}}$	—	12_{G}
—	—	—	—	—	—	—	$\mathbf{10_{F}}$

Distance A-H = 10
Chemin = A-D-G-F-H

3 Paramètres

Paramètres de `\dijkstra` Des ⟨*paramètres*⟩ peuvent être passés à la macro `\dijkstra` dans son argument optionnel qui prend la forme d'une liste de ⟨*clé*⟩=⟨*valeur*⟩.
On peut également régler des ⟨*paramètres*⟩ pour toutes les exécutions de la macro `\dijkstra` à venir avec

`\setdijk{`⟨*paramètres*⟩`}`

mais aussi modifier des ⟨*paramètres*⟩ *par défaut* avec

`\setdijkdefault{`⟨*paramètres*⟩`}`

Pour réinitialiser toutes les ⟨*clés*⟩ à leur ⟨*valeur*⟩ par défaut, il faut exécuter la macro `\initdijk`.

Voici toutes les ⟨*clés*⟩, leur ⟨*valeur*⟩ par défaut et leur description.

show-tab=⟨booléen⟩ (*Défaut* : "true")
Lorsque cette ⟨*clé*⟩ est true, le tableau est affiché par la macro `\dijkstra`. Il ne l'est pas dans le cas contraire.

v-position=⟨texte⟩ (*Défaut* : "c")
Ce paramètre est placé dans l'argument optionnel de `\begin{tabular}[`⟨*v-position*⟩`]` pour spécifier la position que doit avoir le tableau par rapport à la ligne de base.

pre-tab=⟨code⟩ (*Défaut* : ⟨vide⟩)
Ce ⟨*code*⟩ arbitraire est exécuté juste avant le `\begin{tabular}`.

post-tab=⟨code⟩ (*Défaut* : ⟨vide⟩)
Ce ⟨*code*⟩ arbitraire est exécuté juste après le `\end{tabular}`.

col-type=⟨code⟩ (*Défaut* : "c")
Ce ⟨*code*⟩ est le descripteur des colonnes contenant les sommets.

infinity-code=⟨code⟩ (*Défaut* : "$\infty $")
Ce ⟨*code*⟩ est exécuté pour exprimer une distance infinie dans le tableau et dans la macro `\dijkdist`.

norevisit-code=⟨code⟩ (*Défaut* : "–")
Ce ⟨*code*⟩ est exécuté dans le tableau pour exprimer qu'un sommet a déjà été fixé.

h-rules=⟨booléen⟩ (*Défaut* : "false")
Lorsque ce booléen est true, les réglures horizontales entre les étapes sont tracées dans le tableau.

```
\readgraph{
  A [B=7, D=15],
  B [C=12, E=4, F=16],
  C [D=5, F=3],
  D [E=2],
  E [F=14]}
Tableau :
\dijkstra[h-rules=true,
          v-position=b]{A}{F}
```

Tableau :

A	B	C	D	E	F
0	∞	∞	∞	∞	∞
–	**7$_{A}$**	∞	15$_{A}$	∞	∞
–	–	19$_{B}$	15$_{A}$	**11$_{B}$**	23$_{B}$
–	–	19$_{B}$	**13$_{E}$**	–	23$_{B}$
–	–	**18$_{D}$**	–	–	23$_{B}$
–	–	–	–	–	**21$_{C}$**

show-lastcol=⟨booléen⟩ (*Défaut* : "false")
Lorsque ce booléen est true, une colonne supplémentaire est affichée dans le tableau ; cette colonne correspond au sommet fixé.

lastcol-type=⟨code⟩ (*Défaut* : "c|")
Ce ⟨*code*⟩ est le descripteur de la colonne correspondant au sommets fixés.

lastcol-label=⟨code⟩ (*Défaut* : "sommet fix\'e")
Ce ⟨*code*⟩ contient le nom de la colonne correspondant aux sommets fixés.

```
\readgraph{
  A [B=7, D=15],
  B [C=12, E=4, F=16],
  C [D=5, F=3],
  D [E=2],
  E [F=14]}
Tableau :
\dijkstra[show-lastcol]{A}{F}
```

Tableau :

A	B	C	D	E	F	sommet fixé
0	∞	∞	∞	∞	∞	A
–	**7$_{A}$**	∞	15$_{A}$	∞	∞	B
–	–	19$_{B}$	15$_{A}$	**11$_{B}$**	23$_{B}$	E
–	–	19$_{B}$	**13$_{E}$**	–	23$_{B}$	D
–	–	**18$_{D}$**	–	–	23$_{B}$	C
–	–	–	–	–	**21$_{C}$**	F

nopath-string=⟨code⟩ (*Défaut* : "Pas de chemin possible")
Ce ⟨*code*⟩ est placé dans la macro \dijkpath dans le cas où aucun chemin n'a pu être trouvé, comme cela peut être le cas si le graphe est non connexe.

```
\readgraph{
  A [B=2],
  B [C=3],
  D [E=5]}
\dijkstra[show-tab=false]{A}{E}
Chemin = \dijkpath\par
Distance A-E= \dijkdist
```

Chemin = Pas de chemin possible
Distance A-E= ∞

path-sep=⟨code⟩ (*Défaut* : "-")
Ce ⟨*code*⟩ est inséré entre chaque sommet dans la macro \dijkpath.

Formatage distance/sommet Lorsqu'un sommet a un prédécesseur, la macro \formatnodewithprev se charge d'afficher la distance et le sommet. Cette macro prend deux arguments (la ⟨*distance*⟩ et le ⟨*sommet*⟩) et sa définition par défaut est

```
\newcommand*\formatnodewithprev[2]%
{% #1=distance, #2=nom du noeud de provenance
    $#1_{\mathrm{#2}}$%
}
```

ce qui a pour effet de mettre le sommet de provenance en indice de la distance. On peut redéfinir cette macro pour choisir une autre mise en forme comme ci-dessous où le sommet est placé entre parenthèses.

```
\renewcommand*\formatnodewithprev[2]%
{%
  #1 (#2)%
}
\readgraph{
  A [B=7, D=15],
  B [C=12, E=4, F=16],
  C [D=5, F=3],
  D [E=2],
  E [F=14]}
Tableau : \dijkstra{A}{F}
```

Tableau :

A	B	C	D	E	F
0	∞	∞	∞	∞	∞
–	**7$_{A}$**	∞	15 (A)	∞	∞
–	–	19 (B)	15 (A)	**11$_{B}$**	23 (B)
–	–	19 (B)	**13$_{E}$**	–	23 (B)
–	–	**18$_{D}$**	–	–	23 (B)
–	–	–	–	–	**21$_{C}$**

Mise en évidence du sommet fixé Le premier sommet fixé est celui de départ et sa distance est toujours 0.
La macro `\highlightfistnode` prend comme argument la distance (qui est 0) et le traite pour effectuer sa mise en forme. Sa définition par défaut, qui compose cette distance en gras, est :

```
\newcommand*\highlightfirstnode[1]{$\mathbf{#1}$}
```

Les autres sommets, lorsqu'ils sont fixés, apparaissent dans le tableau avec leur distance et leur nom et sont traités par la macro `\highlightnode` qui rend deux arguments. Sa définition permet une mise en forme similaire à ce que fait `\formatnodewithprev`, sauf que la distance et le sommet sont en gras :

```
\newcommand*\highlightnode[2]%
{% #1=distance, #2=nom du noeud de provenance
    $\mathbf{#1}_{\mathrm{\mathbf{#2}}}$%
}
```

Pour obtenir d'autre effets, on peut redéfinir ces macros. L'exemple donné n'est pas réaliste tant les effets sont incohérents, c'est simplement un aperçu de ce qu'il est possible de faire :

```
\renewcommand*\highlightfirstnode[1]%
{%
  \fboxsep=1pt
  \fbox{\color{blue}$\mathbf{#1}$}%
}%
\renewcommand*\highlightnode[2]%
{% #1=distance,
 %  #2=nom du noeud de provenance
  \color{red}$#1_\mathrm{#2}$%
}
\readgraph{
  A [B=7, D=15],
  B [C=12, E=4, F=16],
  C [D=5, F=3],
  D [E=2],
  E [F=14]}
Tableau : \dijkstra{A}{F}
```

Tableau :

A	B	C	D	E	F
$\boxed{\mathbf{0}}$	∞	∞	∞	∞	∞
—	7_A	∞	15_A	∞	∞
—	—	19_B	15_A	11_B	23_B
—	—	19_B	13_E	—	23_B
—	—	18_D	—	—	23_B
—	—	—	—	—	21_C

4 Code

Le code ci-dessous est l'exact verbatim du fichier `dijkstra.sty` :

```
% !TeX encoding = UTF-8
% Ce fichier contient le code de l'extension "dijkstra"
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                                                                    %
\def\dijkname                   {dijkstra}                           %
\def\dijkver                      {0.13}                             %
%                                                                    %
\def\dijkdate                  {2022/10/01}                          %
%                                                                    %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
% --------------------------------------------------------------------
% This work may be distributed and/or modified under the
% conditions of the LaTeX Project Public License, either version 1.3c
% of this license or (at your option) any later version.
% The latest version of this license is in
%
% %     http://www.latex-project.org/lppl.txt
%
% and version 1.3 or later is part of all distributions of LaTeX
% version 2005/12/01 or later.
% --------------------------------------------------------------------
% This work has the LPPL maintenance status `maintained'.
%
% The Current Maintainer of this work is Christian Tellechea
% Copyright : Christian Tellechea 2017-2022
% email: unbonpetit@netc.fr
%        Commentaires, suggestions et signalement de bugs bienvenus !
%        Comments, bug reports and suggestions are welcome.
% --------------------------------------------------------------------
% L'extension dijkstra est composée des 4 fichiers suivants :
```

```
%   - code               : dijkstra.sty
%   - manuel en français : dijkstra-fr.tex & dijkstra-fr.pdf
%   - fichier lisezmoi   : README
% ------------------------------------------------------------------------
%
\csname dijkloadonce\endcsname
\let\dijkloadonce\endinput
\NeedsTeXFormat{LaTeX2e}
\ProvidesPackage{dijkstra}[\dijkdate\space v\dijkver\space Dijkstra Algorithm (CT)]
\RequirePackage{simplekv}

\expandafter\edef\csname dijk_restorecatcode\endcsname{\expandafter\catcode\number`\_=\number\catcode`\_\relax}
\catcode`\_=11

\newcount\dijk_nest
\newcount\dijk_cnt
\newif\ifdijk_oriented

\def\dijk_maxint{1073741823}
\def\dijk_quark{\dijk_quark}
\def\dijk_cscmd#1#2{\expandafter#1\csname#2\endcsname}
\def\dijk_gobarg#1{}
\long\def\dijk_first#1#2{#1}
\long\def\dijk_second#1#2{#2}
\long\def\dijk_earg#1#2{\expandafter\dijk_earg_i\expandafter{#2}{#1}}\let\dijk_exparg\dijk_earg
\long\def\dijk_eearg#1#2{\expandafter\expandafter\expandafter\dijk_earg_i\expandafter\expandafter\expandafter{#2}{#1}}
\long\def\dijk_earg_i#1#2{#2{#1}}
\long\def\dijk_ifx#1{\ifx#1\expandafter\dijk_first\else\expandafter\dijk_second\fi}
\long\def\dijk_ifempty#1{\dijk_ifempty_i#1\_nil\_nil\dijk_second\dijk_first\__nil}%
\long\def\dijk_ifempty_i#1#2\_nil#3#4#5\__nil{#4}
\def\dijk_ifcsname#1{\ifcsname#1\endcsname\expandafter\skv_first\else\expandafter\skv_second\fi}
\def\dijk_addtomacro#1#2{\expandafter\def\expandafter#1\expandafter{#1#2}}
\def\dijk_eaddtomacro#1#2{\dijk_exparg{\dijk_addtomacro#1}{#2}}
\def\dijk_eeaddtomacro#1#2{\dijk_eearg{\dijk_addtomacro#1}{#2}}
\long\def\dijk_exptwoargs#1#2#3{\dijk_exparg{\dijk_exparg{#1}{#2}}{#3}}
\def\dijk_ifnum#1{\ifnum#1\expandafter\dijk_first\else\expandafter\dijk_second\fi}
\def\dijk_swapargs#1#2#3{#1{#3}{#2}}
\def\dijk_ifstar#1#2{\def\dijk_ifstar_i{\dijk_ifx{*\dijk_nxttok}{\dijk_first{#1}}{#2}}\futurelet\dijk_nxttok\↩
    dijk_ifstar_i}
\def\dijk_ifopt#1#2{\def\dijk_ifopt_i{\dijk_ifx{[\dijk_nxttok}{#1}{#2}}\futurelet\dijk_nxttok\dijk_ifopt_i}
\def\dijk_stripsp#1%
{%
  \long\def\dijk_stripsp##1{\expanded{\dijk_stripsp_i\_marksp##1\__nil\_marksp#1\_marksp\_nil}}%
  \long\def\dijk_stripsp_i##1\_marksp#1##2\_marksp##3\_nil{\dijk_stripsp_ii##3##1##2\__nil#1\__nil\_nil}%
  \long\def\dijk_stripsp_ii##1#1\__nil##2\_nil{\dijk_stripsp_iii##1##2\_nil}%
  \long\def\dijk_stripsp_iii##1##2\__nil##3\_nil{\unexpanded{##2}}%
}
\dijk_stripsp{ }

\def\dijk_foreach#1\in#2#3%
{%
  \global\advance\dijk_nest1
  \dijk_cscmd\def{dijk_loopcode_\number\dijk_nest}{#3}%
  \dijk_foreach_i#1#2,\dijk_quark,%
  \dijk_cscmd\let{dijk_loopcode_\number\dijk_nest}\empty
  \global\advance\dijk_nest-1
}%

\def\dijk_foreach_i#1#2,%
{%
  \def#1{#2}%
  \dijk_ifx{\dijk_quark#1}
  {%
  }
  {%
    \dijk_ifx{#1\empty}{}{\csname dijk_loopcode_\number\dijk_nest\endcsname}%
    \dijk_foreach_i#1%
  }%
}%

\def\dijk_ifinst#1#2%
```

```
{% #2 est-il dans #1 ?
  \def\dijk_ifinst_i##1#2##2\_nil{\dijk_swapargs{\dijk_ifempty{##2}}}%
  \dijk_ifinst_i#1#2\_nil
}

\def\readgraph
{%
  \dijk_ifstar{\dijk_orientedtrue\readgraph_a}{\dijk_orientedfalse\readgraph_a}%
}

\def\readgraph_a#1%
{%
  \let\dijk_initlistofnodes\empty% liste des sommets
  \let\dijk_graph\empty% argument #1 où l'on va enlever les espaces
  \dijk_sanitizegraph#1,\dijk_quark[],% enlever tous les espaces indésirables et évaluer les nombres dans l'argument #1
  \expandafter\readgraph_b\dijk_graph,\dijk_quark[],%
}

\def\dijk_sanitizegraph#1,%
{%
  \expandafter\expandafter\expandafter\dijk_sanitizegraph_i\dijk_stripsp{#1},% bugfix 0.12
}

\def\dijk_sanitizegraph_i#1[#2],%
{%
  \dijk_ifx{\dijk_quark#1}
  {%
    \dijk_removelastcommainmacro\dijk_graph
  }
  {%
    \dijk_eearg{\def\dijk_childnodes}{\dijk_stripsp{#1}[]%
    \dijk_foreach\dijk_temp\in{#2}{\expandafter\dijk_sanitizegraph_ii\dijk_temp\_nil}%
    \dijk_removelastcommainmacro\dijk_childnodes
    \dijk_eaddtomacro\dijk_graph{\dijk_childnodes],}%
    \dijk_sanitizegraph
  }%
}

\def\dijk_sanitizegraph_ii#1=#2\_nil
{%
  \dijk_eeaddtomacro\dijk_childnodes{\dijk_stripsp{#1}=}%
  \dijk_eaddtomacro\dijk_childnodes{\the\numexpr#2\relax,}%
}

\def\dijk_removelastcommainmacro#1%
{%
  \expandafter\dijk_removelastcommainmacro_i#1\_nil#1%
}

\def\dijk_removelastcommainmacro_i#1,\_nil#2%
{%
  \def#2{#1}%
}

\def\readgraph_b#1#2[#3]#4,%
{%
  \dijk_ifx{\dijk_quark#1}
  {%
    \dijk_exparg{\dijk_foreach\dijk_tempnodename\in}{\dijk_initlistofnodes}
    {% pour chaque sommet
      \dijk_eearg{\dijk_foreach\dijk_tempnodechild\in}{\csname dijknode\dijk_tempnodename\endcsname}
      {% pour chaque enfant
        \expandafter\readgraph_c\dijk_tempnodechild\_nil\dijk_currentnodechildname\dijk_currentnodechilddist% capturer ↩
            nom et distance de l'enfant
        \dijk_exptwoargs\dijk_ifinst\dijk_initlistofnodes{\dijk_currentnodechildname,}% si l'enfant n'est pas dans la ↩
            liste des sommets
        {%
        }%
        {%
          \dijk_eaddtomacro\dijk_initlistofnodes{\dijk_currentnodechildname,}% l'y mettre
          \dijk_cscmd\let{dijknode\dijk_currentnodechildname}\empty% et initialiser la liste de ses enfants
```

```
        }%
        \unless\ifdijk_oriented% si graphe non orienté, ajouter les distances inverses
          \dijk_exparg{\dijk_eearg\dijk_ifinst{\csname dijknode\dijk_currentnodechildname\endcsname}}{\dijk_tempnodename=}% si le parent est dans déjà un des enfants de l'enfant
          {%
            \expandafter\def\expandafter\readgraph_d\expandafter########\expandafter1\dijk_tempnodename=########2,########3\_nil{%
              \unless\ifnum########2=\dijk_currentnodechilddist\relax% si distance différente  : erreur, c'est pas normal
                \errmessage{Distance "\dijk_tempnodename=########2" incorrecte dans \dijk_currentnodechildname{} comprise comme "\dijk_tempnodename=\dijk_currentnodechilddist"}%
                \dijk_cscmd\edef{dijknode\dijk_currentnodechildname}{########1\dijk_tempnodename=\dijk_currentnodechilddist,########3}%
              \fi
              }%
            \expandafter\expandafter\expandafter\readgraph_d\csname dijknode\dijk_currentnodechildname\endcsname\_nil
          }%
          {% sinon, l'y mettre
            \dijk_cscmd\edef{dijknode\dijk_currentnodechildname}{\dijk_tempnodename=\dijk_currentnodechilddist,\csname dijknode\dijk_currentnodechildname\endcsname}%
          }%
        \fi
      }%
    }%
    \dijk_cnt0
    \dijk_exparg{\dijk_foreach\dijk_tempnodename\in}{\dijk_initlistofnodes}
    {% pour chaque sommet, construire la liste de ses enfants
      \advance\dijk_cnt1
      \dijk_cscmd\let{listofchilds_\dijk_tempnodename}\empty
      \dijk_eearg{\dijk_foreach\dijk_tempnodechild\in}{\csname dijknode\dijk_tempnodename\endcsname}
      {%
        \expandafter\readgraph_c\dijk_tempnodechild\_nil\dijk_currentnodechildname\dijk_currentnodechilddist
        \expandafter\dijk_eaddtomacro\csname listofchilds_\dijk_tempnodename\endcsname{\dijk_currentnodechildname,}%
      }%
    }%
    \edef\dijk_numberofnodes{\the\dijk_cnt}%
  }%
  {%
    \def\dijk_currentnodename{#1}%
    \dijk_eaddtomacro\dijk_initlistofnodes{\dijk_currentnodename,}%
    \dijk_cscmd\def{dijknode\dijk_currentnodename}{#3,}%
    \readgraph_b
  }%
}%

\def\readgraph_c#1=#2\_nil#3#4%
{%
  \def#3{#1}\edef#4{\number\numexpr#2\relax}%
}

\def\dijk_nodedist#1#2#3%
{% renvoit la distance du sommet #1 vers #2 dans la macro #3
  \def\dijk_nodedist_i##1#2=##2,##3\_nil{\def#3{##2}}%
  \expandafter\expandafter\expandafter\dijk_nodedist_i\csname dijknode#1\endcsname,#2=1073741823,\_nil%
}

\def\dijk_removenode#1%
{% enlève le sommet #1 de la liste des sommets non vus
  \dijk_exparg{\dijk_ifinst}{\expandafter,\dijk_nodestoexplore}{,#1,}
  {%
    \def\dijk_removenode_i##1,#1,##2\_nil{\dijk_exparg{\def\dijk_nodestoexplore}{\dijk_gobarg##1,##2}}%
    \expandafter\dijk_removenode_i\expandafter,\dijk_nodestoexplore\_nil
  }
  {%
  }%
}

\def\dijkstra
{%
  \dijk_ifopt{\dijkstra_i}{\dijkstra_i[]}%
}
\def\dijkstra_i[#1]#2#3%
```

```
{% #1=sommet départ   #2=sommet arrivée
  \begingroup
    \dijk_ifempty{#1}{}{\setdijk{#1}}%
    \let\dijk_listofnodes\dijk_initlistofnodes
    \let\dijk_nodestoexplore\dijk_initlistofnodes
    \dijk_cnt0
    \dijk_eearg{\def\dijk_currentnode}{\dijk_stripsp{#2}}%
    \dijk_eearg{\def\dijk_endnode}{\dijk_stripsp{#3}}%
    \edef\dijk_tab
    {%
      \noexpand\dijk_pre_tab
      \noexpand\begin{tabular}[\dijk_v_position]{%
            *{\dijk_numberofnodes}{|\dijk_col_type}|%
            \ifboolKV[\dijkname]{show-lastcol}
              {\noexpand\dijk_last_col_type}
              {}%
            }%
          \noexpand\hline
    }%
    \def\dijk_autoamp{\def\dijk_autoamp{\dijk_addtomacro\dijk_tab&}}%
    \dijk_exparg{\dijk_foreach\dijk_tempnodename\in}\dijk_listofnodes
    {% pour tous le sommets du graphe
      \dijk_autoamp% ajouter "&", sauf la première fois
      \dijk_cscmd\let{dist_\dijk_tempnodename}\dijk_maxint% toutes les distances à +inf
      \dijk_cscmd\let{prev_\dijk_tempnodename}\dijk_quark% tous les prédecesseurs à <quark>
      \dijk_eaddtomacro\dijk_tab{\dijk_tempnodename}% peupler 1re ligne du tableau
    }%
    \ifboolKV[\dijkname]{show-lastcol}
      {\dijk_eaddtomacro\dijk_tab{\expandafter&\dijk_lastcol_label}}
      {}%
    \dijk_addtomacro\dijk_tab{\\\hline}%
    \dijk_cscmd\def{dist_\dijk_currentnode}{0}% distance sommet de départ = 0
    \dijk_whilenotempty\dijk_nodestoexplore
    {%
      \dijk_findmindist\dijk_currentnode% retourne \dijk_currentnode : le sommet enfant ayant la distance la plus faible
      \dijk_ifx{\dijk_quark\dijk_currentnode}
      {% si le sommet n'est pas trouvé (graphe non connexe)
        \global\let\dijkdist\dijk_infinity_code
        \let\dijk_nodestoexplore\empty% sortir de la boucle
      }
      {%
        \xdef\dijkdist{\csname dist_\dijk_currentnode\endcsname}%
        \unless\ifx\dijk_nodestoexplore\empty
          \dijk_addstep
        \fi
        \dijk_ifx{\dijk_currentnode\dijk_endnode}
        {% si le sommet de sortie est atteint
          \let\dijk_nodestoexplore\empty% sortir de la boucle
        }
        {% sinon
          \dijk_exparg\dijk_removenode\dijk_currentnode% enlever ce sommet du graphe à explorer
          \dijk_eearg{\dijk_foreach\dijk_temp\in}{\csname listofchilds_\dijk_currentnode\endcsname}
          {%
            \dijk_exptwoargs\dijk_ifinst\dijk_nodestoexplore{\dijk_temp,}
              {\dijk_exptwoargs\dijk_updatedist\dijk_currentnode\dijk_temp}%
              {}%
          }%
          \advance\dijk_cnt1
        }%
      }%
    }%
    \ifboolKV[\dijkname]{h-rules}
      {}
      {\dijk_addtomacro\dijk_tab\hline}%
    \dijk_addtomacro\dijk_tab{\end{tabular}}%
    \dijk_eaddtomacro\dijk_tab{\dijk_post_tab}%
    \dijk_ifx{\dijk_quark\dijk_currentnode}
      {\global\let\dijkpath\dijk_nopath_string}
      {\dijk_exparg\dijk_createpath\dijk_currentnode}% calculer le chemin sauf s'il est impossible à trouver
    \ifboolKV[\dijkname]{show-tab}\dijk_tab{}% afficher le tableau
  \endgroup
```

```
}

\def\dijk_createpath
{%
  \global\let\dijkpath\dijk_currentnode
  \dijk_createpathi
}
\def\dijk_createpathi#1%
{% #1=sommet en cours
  \dijk_eearg{\def\dijk_temp}{\csname prev_#1\endcsname}%
  \dijk_ifx{\dijk_quark\dijk_temp}
  {%
  }
  {%
    \xdef\dijkpath{\dijk_temp\dijk_path_sep\dijkpath}%
    \dijk_exparg\dijk_createpathi\dijk_temp
  }%
}

\def\dijk_findmindist#1%
{% trouve dans "sommets à explorer" celui ayant la distance mini
  \let\dijk_mindist\dijk_maxint
  \let#1\dijk_quark
  \dijk_exparg{\dijk_foreach\dijk_currentnodechildname\in}\dijk_nodestoexplore
  {%
    \ifnum\csname dist_\dijk_currentnodechildname\endcsname<\dijk_mindist\relax
      \expandafter\let\expandafter\dijk_mindist\csname dist_\dijk_currentnodechildname\endcsname
      \let#1\dijk_currentnodechildname
    \fi
  }%
}

\def\dijk_whilenotempty#1#2%
{% tant que la macro #1 n'est pas \ifx-vide, exécuter #2
  \dijk_ifx{#1\empty}{}{#2\dijk_whilenotempty#1{#2}}%
}

\def\dijk_updatedist#1#2%
{%
  \dijk_nodedist{#1}{#2}\tempdist
  \ifnum\numexpr\csname dist_#1\endcsname+\tempdist\relax<\csname dist_#2\endcsname\relax
    \dijk_cscmd\edef{dist_#2}{\the\numexpr\csname dist_#1\endcsname+\tempdist\relax}%
    \dijk_cscmd\edef{distwithprev_#2}{\noexpand\formatnodewithprev{\the\numexpr\csname dist_#1\endcsname+\tempdist\relax↩
        }{\unexpanded{#1}}}%
    \dijk_cscmd\def{prev_#2}{#1}%
  \fi
}

\def\dijk_addstep
{%
  \def\dijk_autoamp{\def\dijk_autoamp{\dijk_addtomacro\dijk_tab&}}%
  \dijk_exparg{\dijk_foreach\dijk_temp\in}\dijk_listofnodes
  {%
    \dijk_autoamp
    \dijk_exptwoargs\dijk_ifinst\dijk_nodestoexplore\dijk_temp
    {%
      \ifnum\csname dist_\dijk_temp\endcsname=\dijk_maxint\relax
        \dijk_eaddtomacro\dijk_tab{\dijk_infinity_code}%
      \else
        \dijk_ifx{\dijk_temp\dijk_currentnode}% si c'est le sommet fixé, le mettre en valeur
        {%
          \dijk_ifcsname{distwithprev_\dijk_temp}
          {%
            \dijk_eeaddtomacro\dijk_tab{\expandafter\expandafter\expandafter\dijk_highlightnode
              \csname distwithprev_\dijk_temp\endcsname}% forme \dijk_highlightnode\formatnodewithprev{<dist>}{<sommet>}
          }
          {%
            \dijk_eeaddtomacro\dijk_tab{\expandafter\expandafter\expandafter
            \highlightfirstnode\expandafter\expandafter\expandafter
            {\csname dist_\dijk_temp\endcsname}}% forme \highlightfirstnode{0}
          }%
```

```
        }
        {% sinon, afficher normalement (forme \formatnodewithprev{<dist>}{<sommet>})
          \dijk_eeaddtomacro\dijk_tab{\csname dist\ifcsname distwithprev_\dijk_temp\endcsname withprev\fi _\dijk_temp\↩
              endcsname}%
        }%
      \fi
    }%
    {%
      \dijk_eaddtomacro\dijk_tab{\dijk_no_revisit_code}% sommet déjà fixé
    }%
  }%
  \ifboolKV[\dijkname]{show-lastcol}
    {\dijk_eaddtomacro\dijk_tab{\expandafter&\detokenize\expandafter{\dijk_currentnode}}}% ajout du sommet fixé
    {}%
  \dijk_addtomacro\dijk_tab{\\}%
  \ifboolKV[\dijkname]{h-rules}
    {\dijk_addtomacro\dijk_tab\hline}
    {}%
}

\def\dijk_highlightnode\formatnodewithprev{\highlightnode}

\defKV[\dijkname]{%
  v-position     = \def\dijk_v_position     {#1},
  pre-tab        = \def\dijk_pre_tab        {#1},
  post-tab       = \def\dijk_post_tab       {#1},
  col-type       = \def\dijk_col_type       {#1},
  infinity-code  = \def\dijk_infinity_code  {#1},
  norevisit-code = \def\dijk_no_revisit_code{#1},
  lastcol-type   = \def\dijk_last_col_type  {#1},
  lastcol-label  = \def\dijk_lastcol_label  {#1},
  nopath-string  = \def\dijk_nopath_string  {#1},
  path-sep       = \def\dijk_path_sep       {#1}
}

\dijk_restorecatcode

\def\initdijk{\restoreKV[\dijkname]}

% Macros permettant de modifier les <valeurs> des <clés>
\def\setdijk#{\setKV[\dijkname]}

% ... ainsi que les <valeurs> par défaut
\def\setdijkdefault#{\setKVdefault[\dijkname]}

\newcommand*\formatnodewithprev[2]%
{% #1=distance, #2=nom du noeud de provenance
  $#1_{\mathrm{#2}}$%
}

\newcommand*\highlightnode[2]%
{% #1=distance, #2=nom du noeud de provenance
  $\mathbf{#1}_{\mathrm{\mathbf{#2}}}$%
}

\newcommand*\highlightfirstnode[1]%
{%
  $\mathbf{#1}$%
}

\setdijkdefault{
  show-tab       = true,% afficher le tableau
  v-position     = c,% argument optionnel de \begin{tabular}[<arg>]
  pre-tab        = {},% juste avant le \begin{tabular}
  post-tab       = {},% juste après le \end{tabular}
  col-type       = c,% colonnes de type "c" pour les colonnes de distances
  infinity-code  = $\infty$,% pour distance infinie
  norevisit-code = ---,% pour les sommets préalablement fixés
  h-rules        = false,% pas de filets entre les lignes des étapes
  show-lastcol   = false,% si vrai : mettre en plus la colonne "sommet fixé"
  lastcol-type   =  c|,% dernière colonne
```

```
  lastcol-label  = sommet fix\'e,
  nopath-string  = Pas de chemin possible,% si chemin impossible
  path-sep       = -,% séparateur entre sommets dans le chemin
}

\endinput

Versions :
 ____________________________________________________________________
| Version |    Date    | Changements                                 |
|---------+------------+---------------------------------------------|
|   0.1   | 06/09/2017 | Première version                            |
|---------+------------+---------------------------------------------|
|   0.11  | 09/09/2017 | - retrait d'un \show, laissé par oubli après les     |
|         |            |   phases de débogage                        |
|         |            | - petit nettoyage du code                   |
|---------+------------+---------------------------------------------|
|   0.12  | 25/06/2020 | - bugfix : le package est rendu compatible avec la   |
|         |            |   version 0.2 de simplekv                   |
|         |            | - bugfix : mauvaise gestion des espaces dans la macro|
|         |            |   \dijk_sanitizegraph                       |
|---------+------------+---------------------------------------------|
|   0.13  | 01/10/2022 | - le package est rendu compatible avec la version    |
|         |            |   0.2a de simplekv                          |
|         |            | - code en UFT8                              |
|---------+------------+---------------------------------------------|
```