Built the Wikipedia Sub-Graph

Criteria

First Criterion: Multiplicity >1

Da tutti gli hyperlink di una pagina di en.wikipedia elimino:
Main_Page
ripetizioni < 2
links esterni
links non di en.wikipedia

Except[":"] elimina file del tipo:
http://en.wikipedia.org/wiki/Portal:<titolo>
.../File:<immagine>
.../Wikipedia:Contact_Us

```
cleanlinks[links_] := Module[
    {cleanedlinks},
    cleanedlinks = Select[
        links,
        StringMatchQ[#, "http://en.wikipedia.org/wiki/"~~Except[":"]..] &
    ];
    cleanedlinks = Select[ Gather[DeleteCases[cleanedlinks,"http://en.wikipedia.org/wi
    Tally@Flatten[cleanedlinks]
]
(*this function will consider links multiplicity and links with at least 2 repetitions
```

Second Criterion : Multiplicity > 0

Importo anche links singoli

```
cleanlinks2[links_] := Module[
    {cleanedlinks},
    cleanedlinks = Select[
        links,
        StringMatchQ[#, "http://en.wikipedia.org/wiki/"~~Except[":"]..] &
    ];
    cleanedlinks = Select[ Gather[DeleteCases[cleanedlinks,"http://en.wikipedia.org/wi
    Tally@Flatten[cleanedlinks]
]
(*this function will consider links multiplicity and links with at least 2 repetitions
```

Import

Importing first level links

Init_ dev'essere della forma "titolo", dove *titolo* è esattamente la dicitura nel link en.wikipedia

```
importFirstLevelLinks[init_]:= cleanlinks @
    Import["http://en.wikipedia.org/wiki/"<>init, "Hyperlinks"]
(* this output is a list of weighted links with their multiplicity *)
```

```
importFirstLevelLinks2[init_]:= cleanlinks2 @
    Import["http://en.wikipedia.org/wiki/"<>init, "Hyperlinks"]
(* this output is a list of weighted links with their multiplicity *)
```

```
createFirstLevelWeightedEdges[init_,weightedlinks_]:=
{init,Last@StringSplit[#[[1]],"/"],#[[2]]}& /@ weightedlinks
```

Importing second level links

```
importSecondLevelLinks[firstLevelLinks_]:= (cleanlinks @ Import[#,"Hyperlinks"])& /@ f
```

```
importSecondLevelLinks2[firstLevelLinks_]:= (cleanlinks2 @ Import[#,"Hyperlinks"])& /@
```

```
createSecondLevelWeightedEdges[firstNeighbors_,secondLevelLinks_]:=
    Module[{secondNeighbors,secondNeighborsWeighted},
        secondNeighbors=Map[Last[StringSplit[#,"/"]]&,secondLevelLinks[[All,All,1]],{2
        secondNeighborsWeighted=Transpose[{secondNeighbors[[#]],secondLevelLinks[[#,Al
        Flatten[Outer[
            Flatten[{#1,#2}]&,{firstNeighbors[[#]]},{secondNeighborsWeighted[[#]]},2]&
]
```

Create the Sub-Graph until second level

```
findFirstNeighbors[firstLevelLinks_]:= Last@StringSplit[#,"/"]& /@firstLevelLinks[[All
```

```
createTheGraph[init_]:=Module[{firstLevel,secondLevel},
firstLevel=importFirstLevelLinks[init];
secondLevel=importSecondLevelLinks[firstLevel];
Flatten[{createFirstLevelWeightedEdges[init,firstLevel],createSecondLevelWeightedEdges
(* the output is a list of weighted edges *)
]
```

Create the Sub-Graph until second level considering all hyperlinks

```
createTheGraph2[init_]:=Module[{firstLevel,secondLevel},
firstLevel=importFirstLevelLinks2[init];
secondLevel=importSecondLevelLinks2[firstLevel];
Flatten[{createFirstLevelWeightedEdges[init,firstLevel],createSecondLevelWeightedEdges
(* the output is a list of weighted edges *)
]
```

Create the Sub-Graph until third level

```
createTheGraphThirdLevel[init_]:=Module[{firstLevel,secondLevel,thirdLevel},
firstLevel=importFirstLevelLinks[init];
secondLevel=importSecondLevelLinks[firstLevel];
thirdLevel= importSecondLevelLinks[#]& /@ secondLevel;
Flatten[{createFirstLevelWeightedEdges[init,firstLevel],createSecondLevelWeightedEdges
(* the output is a list of weighted edges *)
]
```

Controls

Remove Self Loops

```
NoSelfLoopGraph[EdgesList_]:= DeleteCases[EdgesList,{a_,a_,__}]
```

Correct Multiplicity: remove Multi Edges and update Weights

```
NewGraphWithCorrectMultiplicity[Edges_]:=Module[{unweighted,repetitions,positions,
replaceIndices,deleteIndices,updates,subs,checkedFirst,checkedFinal},
unweighted=Edges[[All,;;2]];
repetitions=Select[Tally[unweighted],(#[[2]]>1)&];
positions=Position[unweighted,#[[;;2]]]&@@@ repetitions;
replaceIndices=positions[[All,1]];
deleteIndices=positions[[All,2]];
updates=Partition[Flatten[{Edges[[#1[[1]],;;2]],Edges[[#1[[1]],3]]+Edges[[#1[[1]],3]]}
subs=Map[#1[[1,1]]→#[[2]]&  ,Transpose[{replaceIndices,updates}]];
checkedFirst=ReplacePart[Edges,subs];
checkedFinal=Delete[checkedFirst,deleteIndices]

(*this function controls and corrects repetitions in the graph, just use it once*)
]
```

Apply both corrections just once:

```
CorrectedGraph[EdgesList_List]:= NewGraphWithCorrectMultiplicity[NoSelfLoopGraph[Edges
```

Print and Read a file

Print the Sub-Graph on a TSV file

As a list of weighted EDGES

```
PrintEdges[EdgesList_List,outputFile_]:= Export["/Users/Levantina/Documents/FISICA/TES
```

Import the Sub-Graph from a TSV file

As a list of weighted EDGES

```
ImportNetwork[filename_]:= Import["/Users/Levantina/Documents/FISICA/TESIPOP/Network/"
```

Operations

List of Pages

```
PagesList[EdgesList_List]:= Union[Flatten[EdgesList[[All,;;2]]]]
```