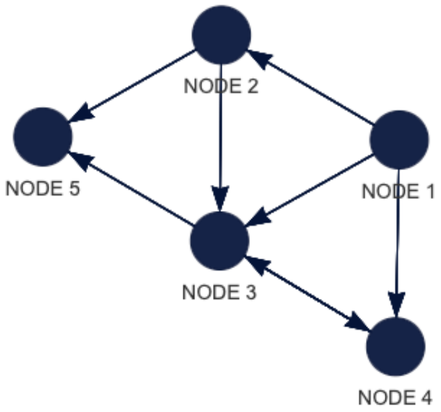
Page Rank

MRJOB

Courant Fanny & Massimi Camille
M2 DS

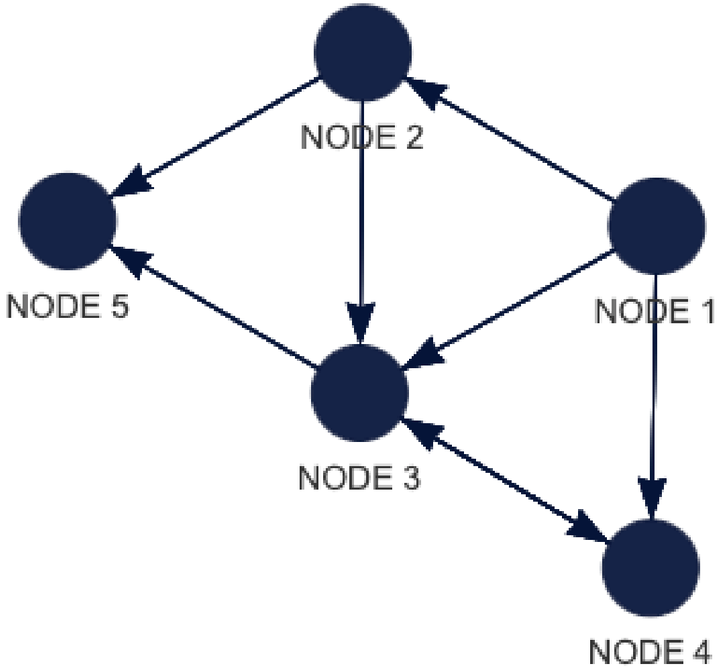


Page Rank

EXEMPLE

test.txt

1	2
1	3
1	4
2	3
4	3
3	4
3	5
2	5



Initialisation

$$PR_0(i) = \frac{1}{N}$$

Itérations

$$PR_{k+1}(i) = \frac{c}{N} + (1 - c) \sum_{j \to i} \frac{PR_k(j)}{n_j}$$

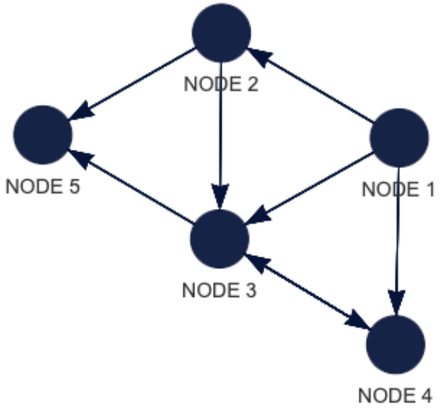
$PR_{k+1}(i)$: PageRank de la page i

c : coefficient (= 0.15)

N : nombre total de pages

n_j : nombre de pages citées par j

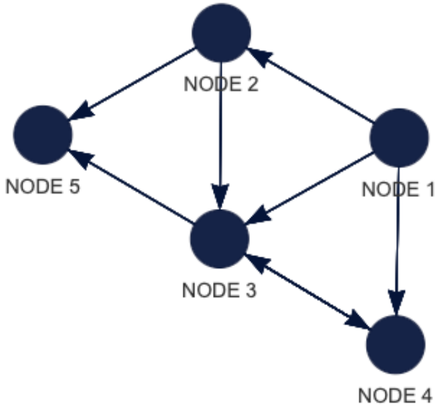
$j \to i$: ensemble de pages qui citent i



Script python

APPLICATION

```
class PageRank(MRJob):
    c = 0.15
    nb_pages = 0  # pour compter le nb de pages
    liste_nb_page = []  # la liste des pages
    dico = {}# dico avec en clé la page source (=la page i) et en valeurs les liens sortants c'est-à-dire les pages j que la page i cite (i -> j)
    pages_citeuses = []  # récupère les pages "citeuses" et supprime celles qui sont citées -> pour garder les pages juste "citeuses"

    # ETAPES :
    def steps(self):

        # JOB1 : initialisation du Page Rank
        # JOB2 : itérations
        return [MRStep(mapper=self.mapper1, reducer=self.reducer1)] +\
               [MRStep(mapper=self.mapper2, combiner=self.combiner, reducer=self.reducer2)] * 10
```



Mapper1

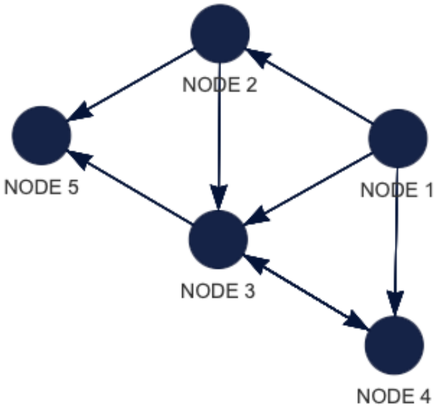
JOB 1

```
def mapper1(self, _, line):
    for elem in motif.findall(line):
        # calcul du nombre de pages
        if elem[0] not in PageRank.liste_nb_page:
            PageRank.liste_nb_page.append(elem[0])
            PageRank.pages_citeuses.append(elem[0])  # ajout de la page citeuse
        if elem[1] not in PageRank.liste_nb_page:
            PageRank.liste_nb_page.append(elem[1])
        yield elem[0], elem[1]
```

test.txt

1	2
1	3
1	4
2	3
4	3
3	4
3	5
2	5

Key page citant	**Value** page citée
"3"	"4"
"1"	"4"
"2"	"3"
"4"	"3"
"1"	"3"
"3"	"5"
"2"	"5"
"1"	"2"



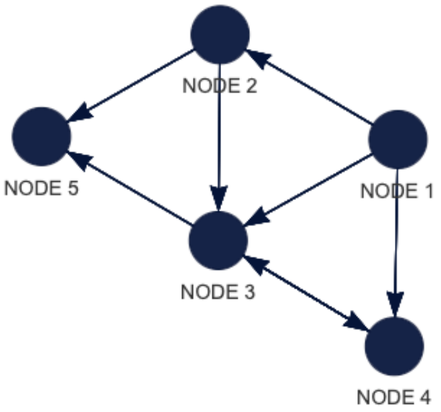
Reducer1

JOB 1

```python
def reducer1(self, page_i, val1):

    liens_sortant = list(val1)
    PageRank.nb_pages = len(PageRank.liste_nb_page)

    # dico avec en clé la page source (=la page i) et en valeurs les
    # liens sortants : les pages j que la page i cite (i -> j)
    if page_i not in PageRank.dico or PageRank.dico[page_i] == None:
        PageRank.dico[page_i] = liens_sortant
    else:
        PageRank.dico[page_i].extend(liens_sortant)

    # exception pour traiter les pages juste citées ou les pages "citeuses"
    for l in liens_sortant:
        # suppression des pages citées dans les pages citeuses
        if l in PageRank.pages_citeuses:
            PageRank.pages_citeuses.remove(l)
        if l not in PageRank.dico:
            PageRank.dico[l] = None

    yield page_i, 1 / PageRank.nb_pages
```

Key page citant	**Value** page citée
"3"	"4"
"1"	"4"
"2"	"3"
"4"	"3"
"1"	"3"
"3"	"5"
"2"	"5"
"1"	"2"

Key page citeuse	**Value** PageRank
"3"	0.2
"4"	0.2
"2"	0.2
"1"	0.2



Mapper2

JOB 2

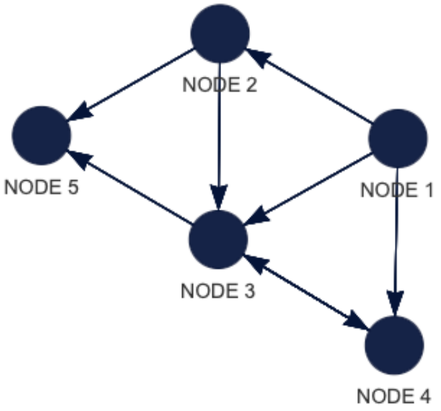
Key page citeuse	Value PageRank
"3"	0.2
"4"	0.2
"2"	0.2
"1"	0.2

```python
def mapper2(self, key, PR):
    if PageRank.dico[key] != None:
        for page_i in PageRank.dico[key]:  # pages i de la somme
            yield key, (((1 - PageRank.c) * PR / sum(collections.Counter(PageRank.dico[key]).values()),
                        page_i))  # key = page j (les pages j de la somme j -> i)
```

$$PR_{k+1}(i) = \frac{c}{N} + (1-c)\sum_{j \to i} \frac{PR_k(j)}{n_j}$$

$$\text{Res1} = (1-c) \times \frac{PR_k(j)}{n_j}$$

Key page citeuse	Value **res1(j=page citeuse)**, page citée
"2"	[0.085, "3"]
"2"	[0.085, "5"]
"1"	[0.056666, "2"]
"1"	[0.056666, "3"]
"1"	[0.056666, "4"]
"4"	[0.17, "3"]
"3"	[0.085, "4"]
"3"	[0.085, "5"]



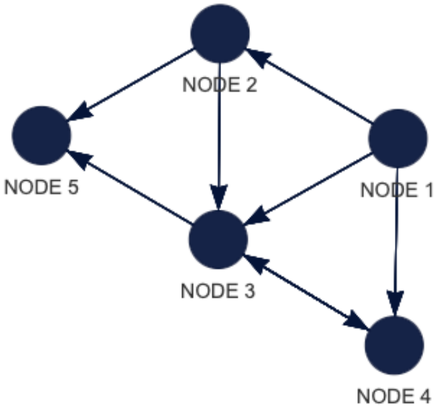
Combiner

JOB 2

Key page citeuse	**Value** **res1(j=page citeuse)**, page citée
"2"	[0.085, "3"]
"2"	[0.085, "5"]
"1"	[0.056666, "2"]
"1"	[0.056666, "3"]
"1"	[0.056666, "4"]
"4"	[0.17, "3"]
"3"	[0.085, "4"]
"3"	[0.085, "5"]

```python
def combiner(self, key, valu4):
    # exception des pages justes "citeuses"
    if key in PageRank.pages_citeuses:
        yield key, 0  # on renvoie 0 pour que reduce prenne en compte cette page
    for val in valu4:
        PR, page_i = val
        yield page_i, PR  # key=page j
```

Key page citée	**Value** res1
"3"	0.085
"5"	0.085
"1"	0
"2"	0.056666
"3"	0.056666
"4"	0.056666
"3"	0.17
"4"	0.085
"5"	0.085



Reducer2

JOB 2

```
def reducer2(self, page_i, val5):
    yield page_i, (PageRank.c * 1 / PageRank.nb_pages) + sum(val5)
```

Key page citée	**Value** res1
"3"	0.085
"5"	0.085
"1"	0
"2"	0.056666
"3"	0.056666
"4"	0.056666
"3"	0.17
"4"	0.085
"5"	0.085

Key page	**Value** PageRank
"4"	0.171666
"5"	0.2
"3"	0.341666
"1"	0.03
"2"	0.086666