istic Informatique Électronique

Université de Rennes

Distributed Data
Neo4J :
Opérateurs avancés

Autin François
Hamon Guillaume
Nolière Léo
Rativel Rémi

Neo4J

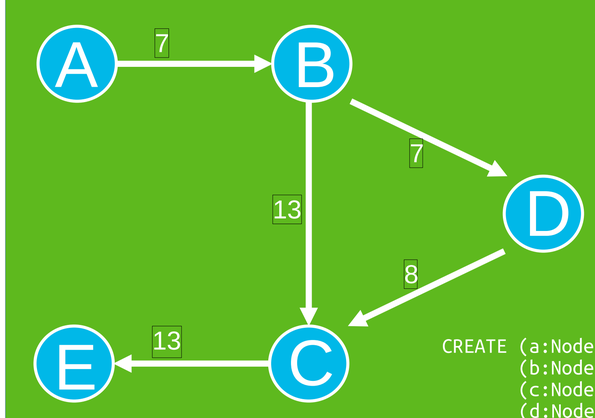
Technologie de **gestion de base de données** parue en Février 2010
- Représentation de **base** sous forme de **nœuds** reliés par des **arcs**
- Outils puissant utilisé pour des requêtes avec des **relations entre objets**

Il est utilisé dans l'application des algorithmes du plus court chemin et l'algorithme PageRank.

Plus court chemin (A*)

- Calcul du plus court chemin entre deux points
- Extension de l'algorithme de Dijkstra par :
  - Peter Elliot Hart
  - Nils John Nilsson
  - Bertram Raphaël
- Réalisé pour permettre le déplacement de robots dans un environnement

Algorithme particulièrement intéressant étant donné la représentation en graphe des bases dans Neo4J



Soit le graphe ci-contre. Neo4J permet de le représenter sous cette forme :

```
CREATE (a:Node {name: 'A'}),
       (b:Node {name: 'B'}),
       (c:Node {name: 'C'}),
       (d:Node {name: 'D'}),
       (e:Node {name: 'E'}),
  (a)-[:CONNECTION {distance: 7}]->(b),
      (b)-[:CONNECTION {distance: 13}]->(c),
      (b)-[:CONNECTION {distance: 7}]->(d),
      (d)-[:CONNECTION {distance: 8}]->(c),
      (c)-[:CONNECTION {distance: 13}]->(e)
```

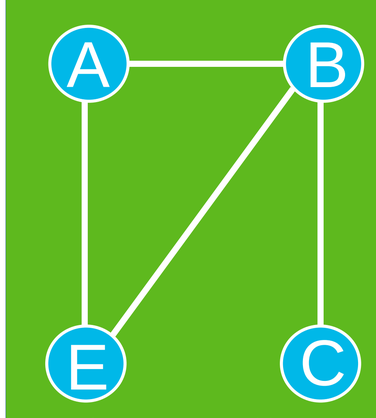
Neo4J nous permet alors d'exécuter l'algorithme A*, ce qui nous donne le résultat :

index	Source Node Name	Target Node Name	Total Cost	Node Names	Costs	path
0	'A'	'E'	33	[A, B, C, E]	[0, 7, 20, 33]	[Node[0], Node[1], Node[2], Node[3]]

Plus court chemin (A*)

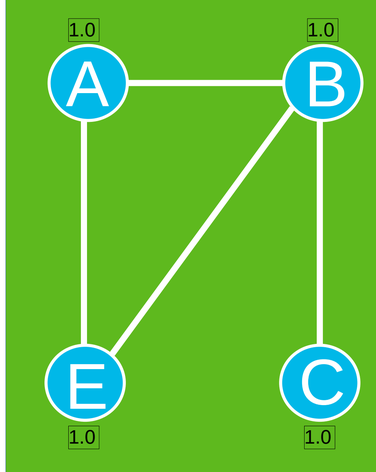
- Détermine l'importance relative d'un nœud dans un graphe
- Se base par le nombre de relation
- Algorithme à la base du moteur de recherche Google
- Développer à l'Université de Standford par Larry Page

Intéressant dans le cadre de Neo4J car il permet d'organiser les données par ordre d'importance.



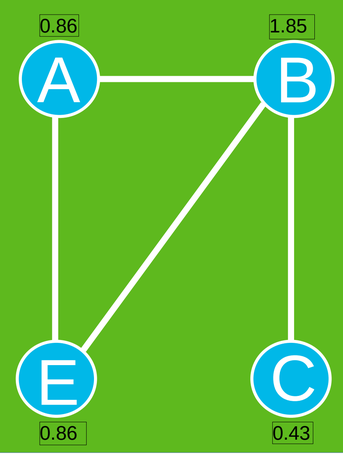
Chaque page est représentée par un nœud, et les liens entre les pages par des arêtes. On utilise PageRank pour évaluer l'importance relative de chaque page dans ce réseau.

Itération 0 : On démarre avec un score initial



Itération n : $PR(u) = (1-d) + d \times \Sigma\, PR(v)/L(v)$
Avec :
- d représente le facteur d'amortissement,
- v parcourt tous les nœuds voisins de u,
- L(v) indique le nombre de liens sortants de v.

Après une itération avec d = 0.85



Sources :
- Neo4j inc. documentation, A* Shortest Path neo4j.com/docs/graph-data-science/current/algorithms/astar/
- Neo4j inc. documentation, PageRank neo4j.com/docs/graph-data-science/current/algorithms/page-rank/