

Rapport TP4 RCR

BOUROUINA Rania
181831052716

CHIBANE Ilies
181831072041

1 Introduction

Dans ce TP, nous allons nous intéresser à l'implémentation de réseaux sémantiques, et plus précisément à l'implémentation de différents algorithmes des réseaux sémantiques.

2 Réseaux sémantiques

Un réseau sémantique est un graphe marqué destiné à la représentation des connaissances. Pour notre TP, un réseau sémantique est défini dans un fichier JSON récupéré via le site qui a été joint dans le TP ou chaque nœud possède un label et un id qui sont par la suite utilisés pour représenter les différentes relations entre les nœuds. Nous allons réaliser les différents algorithmes avec le langage python étant particulièrement adapté pour travailler avec les fichiers JSON.

Voici l'exemple de réseau sémantique que nous allons utiliser pour le reste du TP.

3 Partie 1 : implémenter l'algorithme de propagation de marqueurs dans les réseaux sémantiques

Dans cette partie nous allons implémenter l'algorithme de propagation de marqueurs vu en cours :

```
1 import json
2
3 def get_label(reseau_semantique, node, relation):
4     node_relation_edges = [edge["from"] for edge in reseau_semantique["edges"] if (edge["to"] == node["id"] and edge["label"] == relation)]
5     node_relation_edges_label = [node["label"] for node in reseau_semantique["nodes"] if node["id"] in node_relation_edges]
6     reponse = "il y a un lien entre les 2 noeuds : " + ", ".join(node_relation_edges_label)
7     return reponse
8
9 def propagation_de_marqueurs(reseau_semantique, node1, node2, relation):
10     nodes = reseau_semantique["nodes"]
11
12     solutions_found = []
13
14     for i in range(min(len(node1), len(node2))):
15         solution_found = False
16
17         try:
18             M1 = [node for node in nodes if node["label"] == node1[i][0]]
19             M2 = [node for node in nodes if node["label"] == node2[i][0]]
20
21             edges = reseau_semantique["edges"]
22             propagation_edges = [edge for edge in edges if (edge["to"] == M1[i]["id"] and edge["label"] == "is a")]
23             while len(propagation_edges) != 0 and not solution_found:
24                 temp_node = propagation_edges.pop()
25                 temp_node_contient_edges = [edge for edge in edges if (edge["from"] == temp_node["from"] and edge["label"] == relation)]
26                 solution_found = any(d["to"] == M2[i]["id"] for d in temp_node_contient_edges)
27                 if not solution_found:
28                     temp_node_is_a_edges = [edge for edge in edges if (edge["to"] == temp_node["from"] and edge["label"] == "is a")]
29                     propagation_edges.extend(temp_node_is_a_edges)
30
31             solutions_found.append(get_label(reseau_semantique, M2, relation) if solution_found else "il n'y a pas un lien entre les 2 noeuds")
32         except IndexError:
33             solutions_found.append("Aucune reponse n'est fournie par manque de connaissances.")
34
35     return(solutions_found)
```

FIGURE 1 – Algorithme de propagation de marqueur

Pour cet Algorithme nous allons utiliser le réseau sémantique fourni par le lien donnée dans la série de TP détaillé dans l'image suivante :

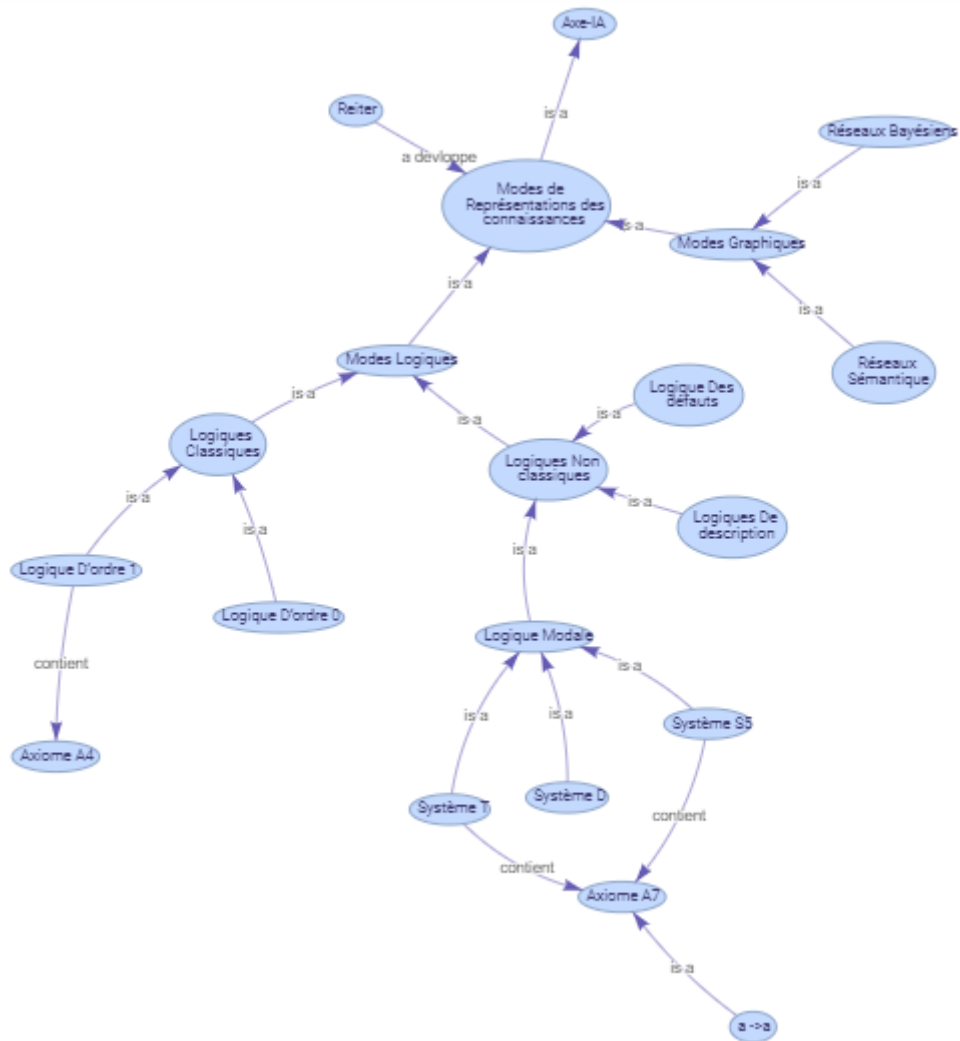


FIGURE 2 – Réseau sémantique utilisé pour la partie 1 du TP

Utilisant plusieurs nœuds marqués en entrée, on obtient les résultats suivants :

```
Partie 1: l'algorithm de propagation de marqueurs
Modes de Représentations des connaissances contient Axiome A7
il y a un lien entre les 2 noeuds : Systeme T, Systeme S5
Modes de Représentations des connaissances contient Axiome A4
il y a un lien entre les 2 noeuds : Logique D ordre 1
Modes de Représentations des connaissances contient Axe-IA
il n'y a pas un lien entre les 2 noeuds
Modes de Représentations des connaissances contient Axiome A9
Aucune reponse n'est fournie par manque de connaissances.
```

FIGURE 3 – Résultat obtenu par l'algorithme de propagation de marqueurs

On voit qu'effectivement l'algorithme à trouver le lien entre ces deux nœuds.

4 Partie 2 : implémenter l'algorithme d'héritage

Dans cette partie, nous allons implémenter l'algorithme d'héritage vu en cours :

```

1 import json
2
3 def get_label(reseau_semantique, node_id):
4     label = [node["label"] for node in reseau_semantique["nodes"] if node["id"] == node_id]
5     return " ,".join(label)
6
7 def heritage(reseau_semantique, name):
8     the_end = False
9
10    nodes = reseau_semantique["nodes"]
11    edges = reseau_semantique["edges"]
12
13    node = [node for node in nodes if node["label"] == name][0]
14    legacy_edges = [edge["to"] for edge in edges if (edge["from"] == node["id"] and edge["label"] == "is_a")]
15    legacy = []
16    properties = []
17    while not the_end:
18        n = legacy_edges.pop()
19        legacy.append(get_label(reseau_semantique, n))
20        legacy_edges.extend([edge["to"] for edge in edges if (edge["from"] == n and edge["label"] == "is_a")])
21    properties_nodes = [edge for edge in edges if (edge["from"] == n and edge["label"] != "is_a")]
22    for pn in properties_nodes:
23        properties.append(": ".join([pn["label"], get_label(reseau_semantique, pn["to"])]))
24    if len(legacy_edges) == 0:
25        the_end = True
26
27    return legacy, properties

```

FIGURE 4 – Algorithme d'héritage

Pour cet Algorithme, nous allons utiliser un autre réseau sémantique exploitant mieux les propriétés de cet algorithme et qui a, lui aussi, été vu en cours et qui est détaillé dans l'image suivante :

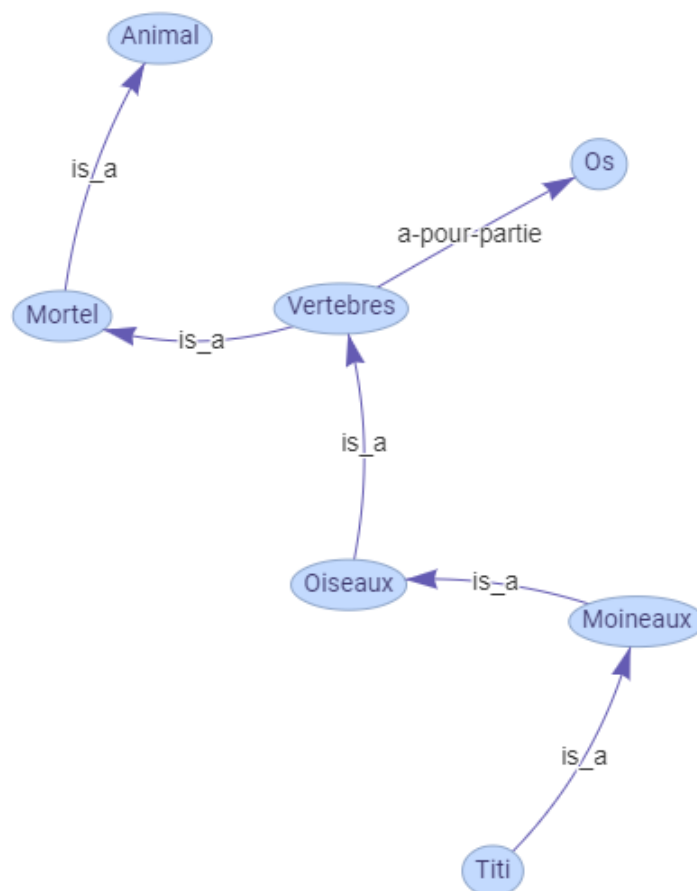


FIGURE 5 – Réseau sémantique utilisé pour la partie 2 du TP

En utilisant le nœud “Titi” en entrée, on obtient le résultat suivant :

```
Partie 2: l'algorithm d'heritage
Resultat de l'inference utiliser:
Titi
Moineaux
Oiseaux
Vertebres
Mortel
Animal
Deduction des priorites:
a-pour-partie: 0s
```

FIGURE 6 – Résultat obtenu par l’algorithme d’héritage

5 Partie 3 : implémentez un algorithme qui permet d’inhiber la propagation dans le cas des liens d’exception

Cet algorithme est très similaire à celui de la partie 1 la seule différence est dans le fait qu’on ne prend pas en compte les arcs de types exception, l’algorithme est plus détaillé ci-dessous :

```
1 import json
2
3 def get_label(reseau_semantique, node, relation):
4     node_relation_edges = [edge["from"] for edge in reseau_semantique["edges"] if (edge["to"] == node["id"] and edge["label"] == relation)]
5     node_relation_edges_label = [node["label"] for node in reseau_semantique["nodes"] if node["id"] in node_relation_edges]
6     reponse = "il y a un lien entre les 2 noeuds : " + ", ".join([node_relation_edges_label[1]])
7     return reponse
8
9 def propagation_de_marqueurs(reseau_semantique, node1, node2, relation):
10     nodes = reseau_semantique["nodes"]
11
12     solutions_found = []
13
14     for i in range(min(len(node1), len(node2))):
15         solution_found = False
16
17         try:
18             M1 = [node for node in nodes if node["label"] == node1[i][0]]
19             M2 = [node for node in nodes if node["label"] == node2[i][0]]
20
21             edges = reseau_semantique["edges"]
22             propagation_edges = [edge for edge in edges if (edge["to"] == M1["id"] and edge["label"] == "is a" and edge["edge_type"] != "exception")]
23             while len(propagation_edges) != 0 and not solution_found:
24                 temp_node = propagation_edges.pop()
25                 temp_node_contient_edges = [edge for edge in edges if (edge["from"] == temp_node["from"] and edge["label"] == relation and edge["edge_type"] != "exception")]
26                 solution_found = any(d["to"] == M2["id"] for d in temp_node_contient_edges)
27                 if not solution_found:
28                     temp_node_is_a_edges = [edge for edge in edges if (edge["to"] == temp_node["from"] and edge["label"] == "is a" and edge["edge_type"] != "exception")]
29                     propagation_edges.extend(temp_node_is_a_edges)
30
31             solutions_found.append(get_label(reseau_semantique, M2, relation) if solution_found else "il n'y a pas un lien entre les 2 noeuds")
32         except IndexError:
33             solutions_found.append("Aucune reponse n'est fournie par manque de connaissances.")
34
35     return(solutions_found)
```

FIGURE 7 – Algorithme d’inhibition de la propagation dans le cas des liens d’exception

Pour cet Algorithme nous allons utiliser le même réseau sémantique avec l’ajout d’un lien d’exception pour voir la différence avec le premier algorithme. Le réseau est détaillé dans l’image suivante :

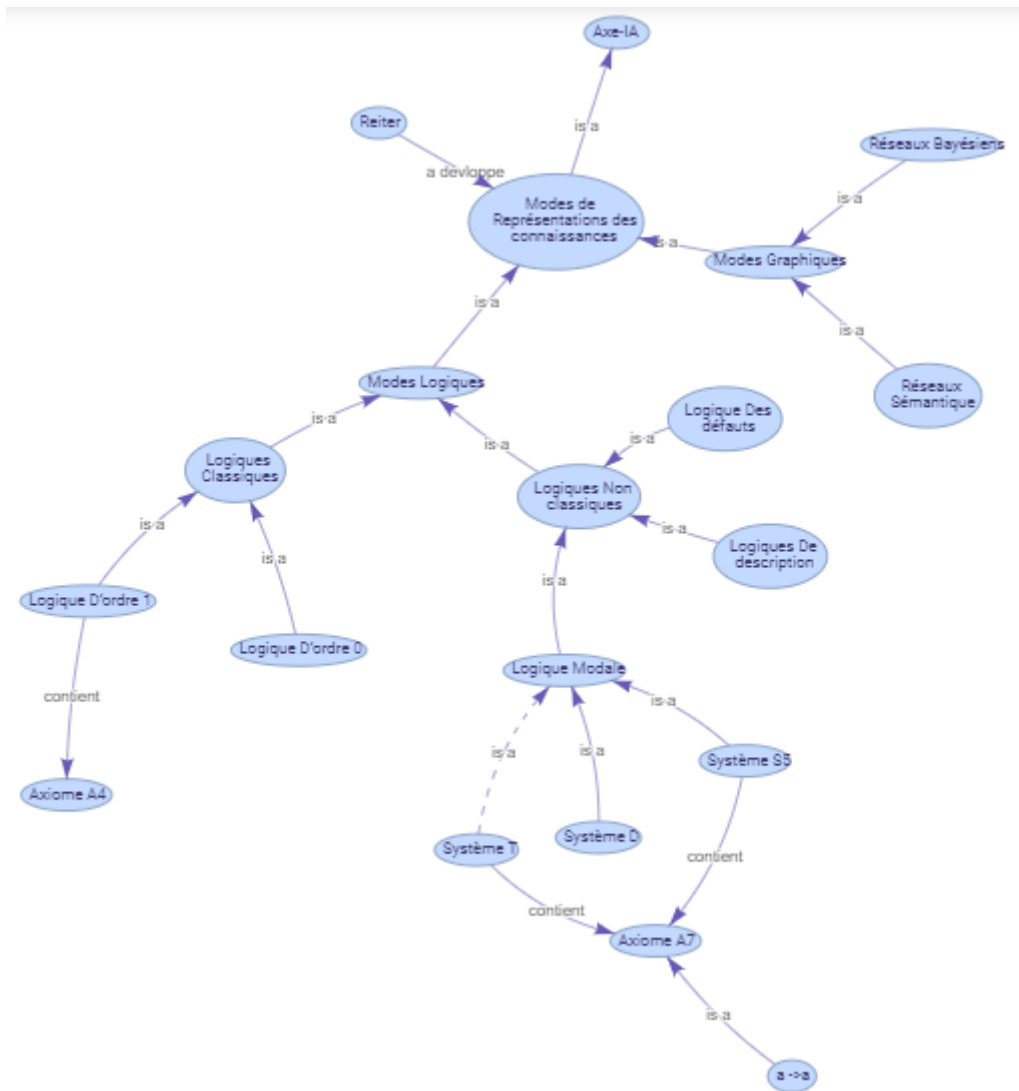


FIGURE 8 – Réseau sémantique utilisé pour la partie 3 du TP

Utilisant la question “Mode de représentation des connaissances contient Axiome A7” on obtient la réponse suivante :

Partie 3: l'algorithme de propagation de marqueurs avec exception
 Modes de Représentations des connaissances contient Axiome A7
 il y a un lien entre les 2 noeuds : Systeme S5

FIGURE 9 – Résultat obtenu par l'algorithme d'inhibition de la propagation dans le cas des liens d'exception