Ministère de l'enseignement Supérieur et de la Recherche Scientifique
Université des Sciences et de la Technologie Houari Boumediene
Faculté d'électronique et d'informatique
Département d'informatique

Rapport

Module : Rprésentaion de connaissances II

Master 1 SII

TP 5

Comparaisons entre propapagtion graphique et inférence logique

- Réalisé par :

  BENHADDAD Wissam

  BOURAHLA Yasser

10-12-2018

Table des matières

Table des figures

CHAPITRE 1

INTRODUCTION

La théorie des possibilité offre de nombreux outils pour représenter des connaissances , et faire de l'inférence sur la ces bases de connaissances, parmi ces outils, nous allons étudier dans ce TP nous allons nous intéresser à la comparaison des performances de deux méthode d'inférence, par propagation et par inférence logique, nous allons tout d'abord introduire les outils utilisés, puis nous allons expliquer les étapes de transformation d'un graphe possibiliste en une base logique possibiliste, puis nous essayerons les deux algorithmes sur un ensemble de test, en paramétrant à chaque fois le volume de données en entrée(ces données sont générées aléatoirement). d'observations, parmis elles

CHAPITRE 2

RÉALISATION

2.1 Les outils utilisés

2.1.1 Matlab-PNT

Pour ce qui est de la représentation graphique, nous disposons d'un très bon outil, la PNT sous Matlab, nous pouvons donc l'utiliser pour concevoir nos réseaux, et appliquer des algorithme de propagation(basé sur le min par exemple), principalement, nous allons l'utiliser pour la comparaison entre des graphes avec peu(beaucoup) de noeud-connections et son équivalent en base possibiliste, pour faire la conversion l'outil Cygwin est nécessaire.

2.1.2 Cygwin

Il faut savoir que pour transformer un graphe possibiliste en une base possibiliste, nous devons tout d'abord effectuer une conversion des relations entre les noeuds dans le modèle graphique en formules logique pondérées dans la base possibiliste.

Pour ce faire nous avons a disposition un algorithme de transformation ainsi qu'un solveur max-sat qui tournent sous un environnement Unix, d'où l'utilisation de l'outil Cygwin qui permet de simuler un tel environnemt dans Windows.

2.2 Propagation de possibilité sur une évidence

2.2.1 Utilisation du prod1vid.m

Le script **prod1vid.m** se charge en gros, de créer un DAG(Directed Acyclic Graph) de tester si c'est un Poly-arbre ou un graphe multi-connecté (test de présence de cycle), puis génère aléatoirement selon les paramètres nbr-nodes et nbr-parent-max une distribution de possibilité en respectant la contrainte $max(\pi(e_i) = 1$.

Nous avons pris la peine de modifier le code pour le transformer en fonction qu'on pourra appeler depuis un autre script en passant les deux paramètres **nbr-nodes** et **nbr-parent-max**, cela pour automatiser le teste des instances.

2.3 Réglage automatique des paramètres

2.3.1 Jeu de test

Différentes valeurs pour nb-node et diffèrent nb-parent-max ont été testées par le biais de notre mini script **test_any.m**, le but est à chaque fois de fixer le nombre de parent, et faire varier le nombre de noeud, de ce fait nous jouons sur la nature(nb-parent < 2 donc poly-arbre) et la densité (nbr-parent>2 donc multi-connecté) du réseau, et à chaque fois nous récupérons le temps de propagation (éventuellement après construction de l'arbre

de jonction dans le cas du multi-connecté), nous passons ensuite à l'exécution du convertisseur **passage.exe**, qui construit la base possibiliste correspondant au réseau généré ultérieurement.

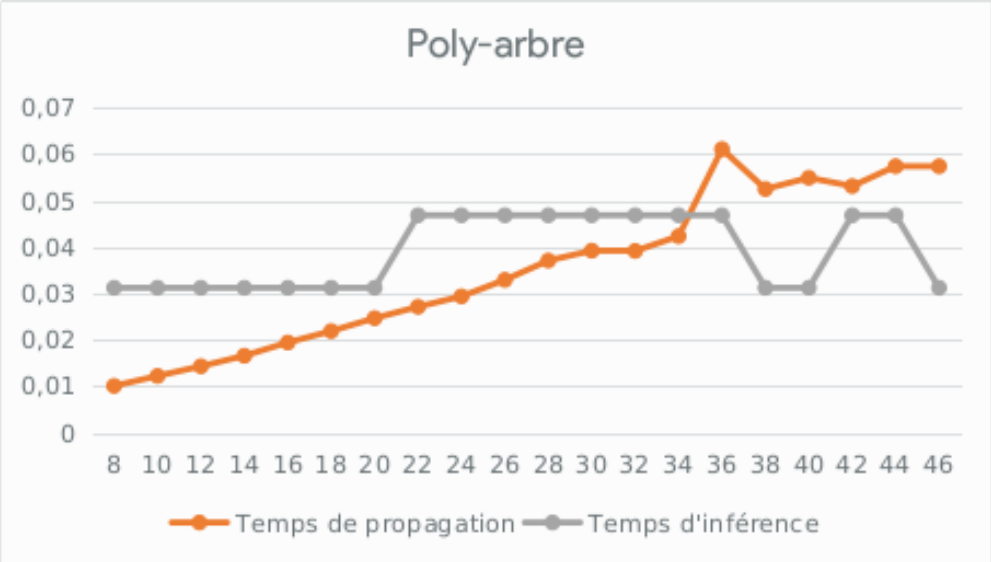
2.3.2 Résultats

Pour résumer le travail effectué, nous avons dressé ces tableaux comparatifs, en fonction des nombres de parents max et du nombre de noeuds, comme demandé, nous avons réalisé les teste sur les 4 catégories suivantes :

Poly-arbre

Pour s'assurer de la génération d'un poly-arbre, nous avons fixé le nombre maximum de parent à 1, nous avons ensuite fait varier le nombre de noeuds dans le réseau. Voici les résultats obtenues :

Nombre noeud	Nombre parent	degré de possibilité	Temps de propagation	Temps d'inférence
8	1	0.00091188	0.010118	0.03125
10	1	0.36788	0.012258	0.03125
12	1	0.00091188	0.014352	0.03125
14	1	0.018316	0.016613	0.03125
16	1	3.06E-07	0.019468	0.03125
18	1	0.13534	0.021925	0.03125
20	1	0.13534	0.024707	0.03125
22	1	0.00091188	0.027159	0.046875
24	1	1	0.029406	0.046875
26	1	0.13534	0.032944	0.046875
28	1	0.13534	0.037111	0.046875
30	1	0.0024788	0.039238	0.046875
32	1	1	0.039181	0.046875
34	1	0.0024788	0.042355	0.046875
36	1	1	0.061121	0.046875
38	1	0.0024788	0.052535	0.03125
40	1	1	0.054939	0.03125
42	1	0.049787	0.05315	0.046875
44	1	1	0.057434	0.046875
46	1	0.13534	0.05742	0.03125



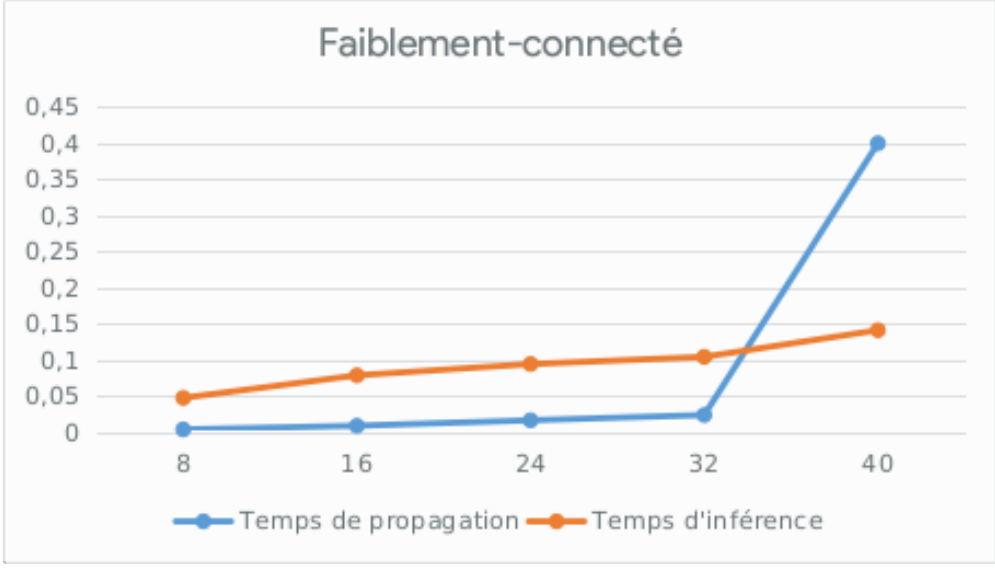
Commentaires : Comme nous pouvions nous y attendre, le cas du polygraphe est aisément simple à traiter, surtout que la phase de construction de l'arbre de jonction n'est pas nécessaire, ce qui explique la rapidité de la propagation, l'inférence logique est aussi très performante, mais puisque c'est un exemple simple d'un polyarbre nous ne pouvons

pas conclure sur la performance des deux, quoique qu'on peut observer une augmentation quasi-linéaire pour la propagation contre une stabilité de l'inférence.

Réseau faiblement multi-connecté

Dans ce cas, nous considérons le cas d'un graphe multi-connecté, par conséquent, le passage par la construction de l'arbre de jonction est nécessaire, cela ralentit considérablement le temps d'exécution, mais ne devrait pas influencer sur le temps de propagation, ce temps dépend surtout de la talle du réseau, les résultats obtenus sont les suivants :

Nombre noeud	Nombre parent	degré de possibilité	Temps de propagation	Temps d'inférence
8	5	0.049787	0.0031867	0.046875
16	5	1.67E-05	0.0082458	0.078125
24	5	0.13534	0.015874	0.09375
32	5	0.049787	0.023162	0.10375
40	5	1	0.39958	0.140625

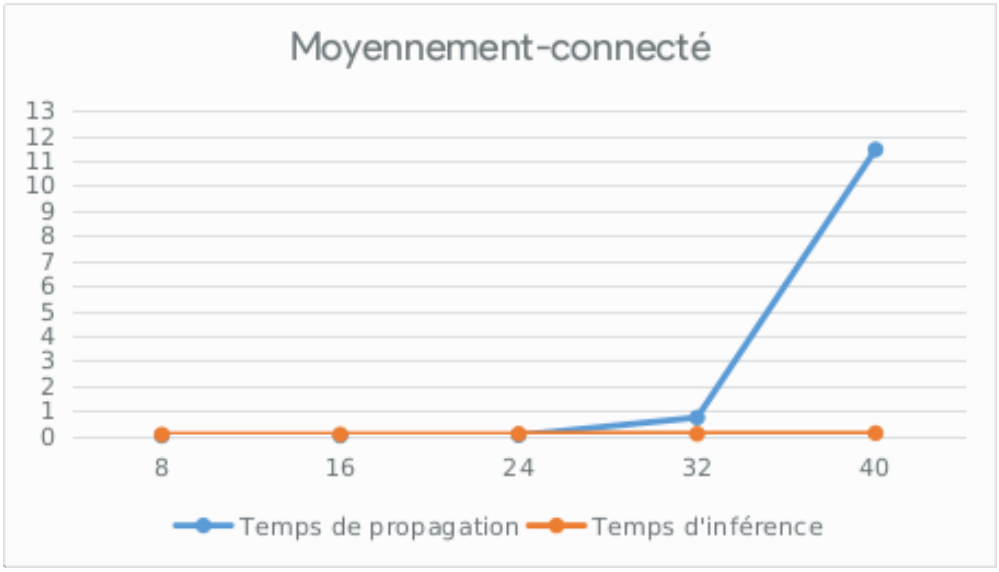


Commentaires On peut remarquer que pour un nombre de noeud petit, le temps de propagation reste très rapide, et cela même comparé au temps d'inférence, mais dès qu'on commence à augmenter ce nombre de noeud, une nette amélioration peut être ressentie, elle reste encore très peu notable, le passage vers la catégorie suivante pourrait nous donner plus de détails.

Réseau moyennement multi-connecté

De même que pour la catégorie précédente, mais de façon plus notable, le temps de construction de l'arbre de jonction commence à se faire sentir, mais le temps de propagation reste encore à être testé.

Nombre noeud	Nombre parent	degré de possibilité	Temps de propagation	Temps d'inférence
8	8	0.36788	0.0054502	0.046875
16	8	0.00091188	0.0080397	0.0525
24	8	1	0.025212	0.078125
32	8	0.049787	0.73254	0.078125
40	8	0.13534	11.451	0.109375

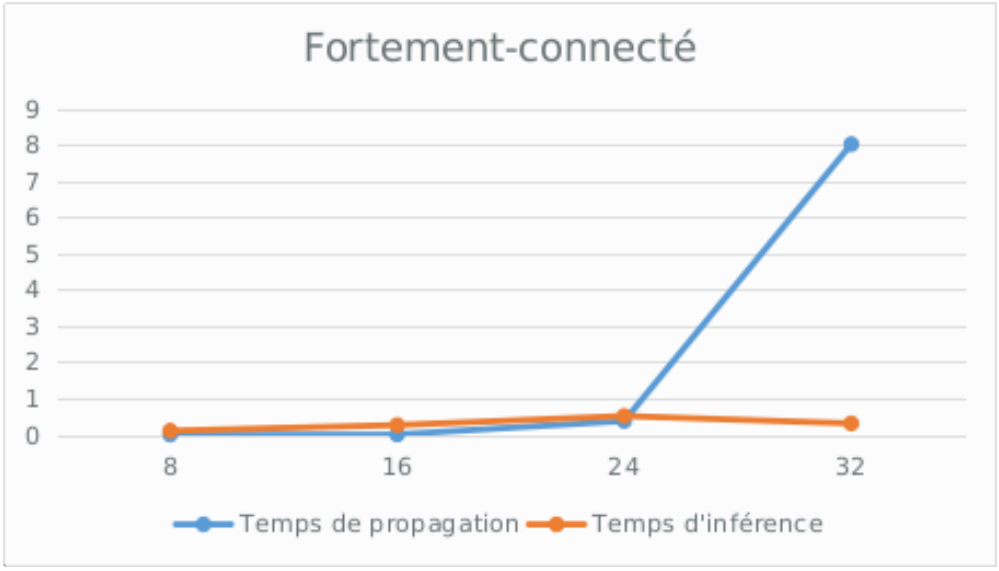


Commentaires Ici nous pouvons finalement apprécier la puissance de l'algorithme d'inférence, en effet pour une grand nombre de noeud, et une densité dans la distribution des relations entre noeud (40 noeuds, 8 parents) le temps de propagation dépasse les 10 secondes(sans compter le temps de construction de l'arbre de jonction), alors que le passage vers la base possibiliste et le lancement de l'inférence se fait dans un temps très raisonnable. C'est ainsi un premier indicateur que la représentation graphique peut avoir des limite comme nous l'avons anticipé.

Réseau fortement multi-connecté

Ici nous poussons le bouchon aux limites, en théorie, pour un grand nombre de noeud, la construction de l'arbre de jonction ne devrait pas se terminer, soit par cause de time-out ou de dépassement de capacité mémoire (c'est cette dernière que nous rencontré durant le test pour les grandes valeurs). Reste a comparé avec les performances de l'inférence logique.

Nombre noeud	Nombre parent	degré de possibilité	Temps de propagation	Temps d'inférence
8	14	0.0067379	0.0027797	0.109375
16	14	0.13534	0.0075522	0.265625
24	14	0.018316	0.37983	0.515625
32	14	0.018316	8.0161	0.3125
40	-	-	-	-



Commentaires Comme anticipé, la propagation graphique montre de grande lacunes, entre le long temps de construction et propagation pour un nombre de noeud égale à 32, et la saturation de la mémoire pour ce même nombre égale à 40, et la vitesse quasi-instantanée d'influencer logique, le choix est vite fait.

CHAPITRE 3

CONCLUSION

Au terme de TP, nous avons pu pu appliquer les aspects théoriques vu en cours, afin de mieux apprécier la puissance des algorithmes d'inférence(que ce soit en terme logique ou par son équivalent graphique, c.à.d la propagation).

Ce que nous pouvons tirer comme conclusion, est qu dans des cas simples c.à.d (petit nombre de variable et faible dépendance entre elles) les deux technique de calculs du degré de possibilité se valent, avec un bonus pour l'inférence logique du point de vue régularité(temps de réponses stable). En revanche dans les cas complexes, du moins ceux que nous avons étudié, l'inférence à de grand avantages par rapport à son homologue graphique, plusieurs raisons peuvent être données comme la complexité temporelle théorique (Complexité exponentielle contre Complexité logarithmique).

En revanche nous n'avons pas pu effectuer d'autre testes plus poussé pour connaître les limites de l'inférence logique, de ce fait décider de quelle méthode est préférable à une autre serait trop prématuré, mais dans les cas que nous avons vu jusqu'à maintenant, l'inférence remporte le match haut la main.