Décomposition de matroïdes orientés:

Compte rendu semaine 1:

MARIN Yann

Introduction:

Au cours de cette semaine, j'ai développé ma compréhension du problème demandé. De plus je me suis intéressé à un cadre théorique potentiellement intéressant qui rejoint la fin de mon stage de math-info (Matroïde de collection). De plus, j'ai programmé une première tentative pour trouver l'ensemble des sous-modèles fixes à partir des bases fixes d'un ensemble de modèle.

Sommaire:

1) Formalisation du problème posé.
2) Cadre théorique via le matroïde de collection de sous modèle.
3) Premier programme.

Glossaire:

Nous allons essayer d'être constant dans l'utilisation des signes pour désigner tel ou tel objet, voici la liste de ces signes et leur utilisation:

Pour tout signe X, nX désignera la taille de X.

-d une dimension.
-M un modèle, désignant généralement un couple (E,B).
-**M** un ensemble de modèles.
-B un ensemble de base signé d'un modèle / un ensemble de base fixe sur **M**.
-E un ensemble de point (ou sommet) avec en général |E|=n.
-b \in B une base. b peut soit désigner un ensemble de d points soit un indice. De plus, b possède un signe (+,0,-) qui ne sera en général pas utilisé pour le moment.
-P désignera un sous-ensemble de points de E. On dira parfois que P est un "sous-modèle" de E.
-S désignera l'ensemble des (d parmi nP) bases que l'on peut faire à partir des points de P. On dira parfois que S "désigne un sous-modèle de " E.
-Sp désignera l'ensemble S qui désigne P.
-C désignera l'ensemble des S de E.

Un sous modèle fixe est un ensemble de points P contenu dans E tel que pour toute base b de S désignant P, b appartient a l'ensemble des bases fixes B de l'ensemble de modèles **M**. On peut alors soit désigner un sous-modèle fixe par P son ensemble de points, soit par S l'ensemble des bases entièrement contenues dans P.

I) Formalisation du problème posé:

Etant donné un ensemble de modèle **M** sur les points E et un ensemble B de base fixe sur **M**, nous voulons trouver:

i) Les ensembles S contenu dans B (Les sous modèles fixes de E) et maximaux pour cette propriété.
ii) Etant donnés ces sous-modèles maximaux, trouver les partitions ou les recouvrements des n points en sous-modèles maximaux (avec le moins de parties possibles)
(Question: Pourquoi en sous-modèles maximaux ?)
iii) Étant donné une partition ou un recouvrement par sous-modèles fixes, trouver comment mesurer certaines caractéristiques entre les sous-parties.

Cette semaine nous allons surtout traiter de la question (i).

Nous allons montrer quelque propriétés importantes avant de passer à la suite:

Proposition: Soit P un sous-modèle. Alors si nous avons (d parmi nP) bases contenus dans Sp, alors nous avons précisément Sp.
Cela veut dire que si nous avons (d parmi nP) bases entièrement contenue dans un ensemble de sommet P, alors ces bases désignent le sous-modèle P.

Proposition: Soit P un ensemble de points de E et d une dimension. Pour couvrir P il faut nP-d+1 bases de Sp.

Démonstration: Par récurrence,
-Pour couvrir d points, il suffit de d-d+1=1 bases.
-Soit u un point de P et supposons que l'on puisse couvrir P/u avec nP-1-d+1=nP-d bases, alors pour couvrir P il faut rajouter une base qui contient u. Il nous faut donc nP-d+1 bases.

Proposition:
(Hérédité) Soit P' un sous-ensemble de P un sous-modèle fixe de **M**, alors P' est aussi un sous-modèle fixe de **M**
(Intersection) Soit P1 et P2 des sous-modèles fixe de **M**, alors P3 = P1 inter P2 est aussi un sous-modèle fixe de **M**.

Démonstration: l'hérédité est triviale et l'intersection découle de l'hérédité.

Définition: Matroïde de collection
Tout ce qui suit dépend de la véracité de ma démonstration dans mon stage de M1 math info, mais cette partie est indépendante du reste du travail.
On garde ici les notation du glossaire mais cela marche dans un cadre général.

Soit C une collection d'ensemble S sur E={1::n} tel que E \in C et chaque ensemble S est muni d'un poids p(S) tel que:
i) Si S1 et S2 sont dans C avec nS1 < nS2 alors p(S1)<p(S2).
ii) Si S1 et S2 sont dans S alors S3 = S1 inter S2 est dans C.
iii) Si S1 et S2 sont dans C et S1 inter S2 est non vide, et S3 le plus petit ensemble de C qui contient S1US2 alors p(S3)<=p(S1)+p(S2)-p(S1 inter S2).

Soit X un sous ensemble de E de taille p(E) tel que pour tout S dans C, |S inter X| <= p(S).
L'ensemble de ces X forment les bases d'un matroïde. (que j'appelle matroïde de collection).

Je fais la démonstration dans mon papier de stage de math-info mais rien ne dit que ma démonstration est bonne, mais ce n'est pas tellement important pour la suite.

II) Cadre théorique:

Soit M un modèle sur E={1::n} et B={1::(d parmi n)} l'ensemble des indices des parties de taille d de E rangées par ordre lexicographique ("les bases possibles sur E").
Soit M(C,B) le matroïde de collection C sur l'ensemble B tel que:
-C est l'ensemble des Sp pour P dans E.
-Pour tout Sp dans C, p(Sp)=nP-d+1

Alors les bases de M(C,B) sont précisément les ensembles X de B de taille nE-d+1 qui recouvrent E.
En d'autre termes, un ensemble de base X couvrent tous les points de E si et seulement si pour tout sous modèles P défini par Sp |X inter Sp| <=nP-d+1.

A partir de maintenant, on désignera C muni du poids p comme "la table des sous-modèles" ou "la table des recouvrements" de E.

Cette table n'est pas calculable en pratique. Elle ne dépend que du nombre de points de E et non pas des bases fixes mais elle donnerait des informations intéressantes sur nos modèles. Par exemple, si l'on voulait savoir si un ensemble de bases fixes couvrent le modèle il suffirait de regarder si notre ensemble contient une base du matroïde non orienté M(C,B).
D'autre propriété peuvent ressortir, par exemple les sous-modèles fixes sont les ensembles S dans C tel que |S inter B|=|S|. Cela peut aussi être un bon point de départ théorique pour les questions (ii) et (iii):

-On pourrait trouver une partition en regardant tous les ensembles Sp tels que |Sp inter B|=nP-d+1 puis en gardant un sous ensemble de Sp qui couvrent E sans s'intersecter.
-Etant donnée une partition, pour mesurer l'interaction entre deux sous modèles Sp1 et Sp2 on pourrait regarder le nombre de bases non fixes dans Sp le plus petit ensemble de C contenant Sp1 et Sp2. (ie: P=P1UP2).

Cette table est théoriquement très riche mais en pratique elle demanderait de calculer tous les sous-modèles faisable pour une taille n. Pour illustrer la difficulté, sur 10 points en dimension 4 il faudrait:

1) Calculer les (4 parmi 10)= 210 bases
2) Calculer les différentes (4 parmi 5)=5 bases dans les différentes façons de prendre (5 parmi 10)= 252 points.
3) Calculer les différentes (4 parmi 6)=15 bases dans les différentes façons de prendre (6 parmi 10)= 210 points.
4) Calculer les différentes (4 parmi i) dans les différentes façons de prendre (d+i-1 parmi 10)... pour i de 1 à 10.

Il y a probablement un moyen de rendre cela plus efficace, voir de partir de la table de n-1 sommet pour avoir celle de n sommets, mais dans tout les cas on ne peut calculer et manipuler cette table que pour de très petit n en pratique.

Néanmoins, cette table pourrait servir pour étudier des sous-modèles de très petite taille.

Par exemple on peut se poser la question suivante:

Question (iv):

Étant donné **M** un ensemble de modèle et B un ensemble de bases fixes. Peut-on partitionner **M** en ensemble de sous-modèles eux-mêmes partitionnables en sous-modèles fixes ?

III) Un premier algorithme pour répondre à la question 1:

On rappelle qu'il s'agit de trouver étant donné un ensemble de bases fixes l'ensemble des sous-modèles fixes (maximum).

Dans le TER, l'étudiant avait choisi de partir des sous-modèles fixes de taille d (les bases) pour les augmenter en ajoutant des bases de B jusqu'à trouver les sous-modèles fixes de taille maximum.

On va donc faire l'inverse et partir de l'ensemble B et retirer des bases jusqu'à avoir les sous-modèles fixes maximaux.

Dans le TER, l'étudiant pour savoir si un sous-modèle P était fixe regardait si Sp contenait une base qui ne soit pas fixe via l'ensemble des bases non fixes. Nous n'utilisons pas ce critère mais à la place, nous regarderons le nombre de bases fixes contenues dans Sp. Si ce nombre est inférieur à nSp, alors c'est que P n'est pas un sous-modèles fixe.

Nous allons pour cela faire l'algorithme récursif suivant:

TROUVER_SOUS_MODELES_FIXES(B,E,nB,nE):
Entrées:
-B un ensemble de base fixe
-E un ensemble de points. (que l'on suppose couvert par B).
-nB la taille de B
-nE la taille de E

Si nB=nE:
    On rajoute E dans l'ensemble des sous-modèles fixes;
Si nB<nE:
    Pour tout sous ensemble Bi de B de taille nBi=nB-1:
        On calcul Ei l'ensemble des points couvert par Bi;
        On calcul nEi la taille de Ei
        TROUVER_SOUS_MODELES_FIXES(Bi,Ei,nBi,nEi)

Et on initialise la récursion sur TROUVER_SOUS_MODELES_FIXES(B,E,nB,nE).

En fait, cet algorithme va bêtement tester pour tout sous ensemble Bi de B si Bi respecte le critère mentionné au dessus pour voir si Bi donne un sous-modèles fixe.
Si c'est le cas, il ajoute les points couverts par Bi dans la solution et ne calcule pas les sous-ensembles de B contenu dans Bi.
Si ce n'est pas le cas, il cherche dans les sous-ensemble de Bi.

Cet algorithme trouve tous les sous-modèles fixes puisqu'il test tout les sous ensembles de Bi qui ne sont pas contenus dans des sous-modèles fixes.
On sait de plus qu'il s'arrête puisqu'à chaque récursion B perd un sommet.

Par contre, dans le pire des cas, (càd s'il n'y a pas de sous-modeles fixes en dehors des bases) il test les 2^|B| sous parties de B. De plus, pour chaque sous-ensemble il fait une récursion, ce qui est problématique sur python. (Je crois ?)

J'ai essayé une autre version pour pour réduire le nombre de récursion, qui pour le moment ne fonctionne pas (en tout cas le code), et ne réduit pas le nombre de calcul:

TROUVER_SOUS_MODELES_FIXES_v2(B,E,nB,nE):
Entrées:
-B un ensemble de base fixe
-E un ensemble de points. (que l'on suppose couvert par B).
-nB la taille de B
-nE la taille de E

Si nB=nE:
    On ajoute E dans l'ensemble des sous-modèles fixes;

Si nB<nE:

Pour tout sous ensemble Bi de B de taille nBi=nB-1:

On calcul Ei l'ensemble des points couvert par Bi;

On calcul nEi la taille de Ei

Si nEI < nE:

TROUVER_SOUS_MODELES_FIXES(Bi,Ei,nBi,nEi)

Une autre version (non implémentée) serait :

TROUVER_SOUS_MODELES_FIXES_v3(B,E,nB,nE):
Entrées:
-B un ensemble de base fixe
-E un ensemble de points. (que l'on suppose couvert par B).
-nB la taille de B
-nE la taille de E

Si nB=nE:

On ajoute E dans l'ensemble des sous-modèles fixes;

Si nB<nE:

Pour tout sous ensemble Pi de points de E calculer les sous-ensembles de Bi maximaux qui les couvrent (et aucun autre)

On calcul nPi la taille de Pi

TROUVER_SOUS_MODELES_FIXES(Bi,Pi,nBi,nPi)

et alors le nombre de récursions serait le nombre de sous ensemble de E exactement couvert par des sous-ensembles de B, mais le temps de calcul dépend alors de l'étape "Pour tout sous ensemble Pi de points de E calculer les sous-ensembles de Bi maximaux qui les couvrent (et aucun autre)"

En python, le code de la première version est le suivant:

```python
def trouver_sous_modèles_fixesv2(B,r,tab): #B un ensemble d'indice, nB l'ensemble des points couvert par B,r indice de récursion, tab tableau du résultat
    tailleB = len(B)
    nB = from_bases_indices_to_point(B)
    taillenB = len(nB)
    combination = combination_tab[taillenB-d]
    if tailleB == combination_: #on termine la récursion
        tab.append(nB)
    elif tailleB > combination_: #Normalement on entre jamais dans cette boucle.
        print("Qu'est-ce qu'il se passe ????")
    else: #cas tailleB < combination #TODO si on peut améliorer l'algorithme c'est ici
        #print("Cas 3!")
        for i in range(r,tailleB):
            Bi = [x for x in B]
            del Bi[i]
            trouver_sous_modèles_fixesv2(Bi, i, tab)
            '''On test tous les sous-ensemble de B (sauf ceux correspondants à des sous-modèles non maximaux) mais est-ce bien nécessaire ?
                -Si on ne prend que les sous ensemble de degré inférieur ?
                A tester, je crois que ça revient au même en fait, on ferait moins de récursion mais autant de test
                Mais est-ce qu'il ne faut pas diminuer au maximum le nombre de récursion ?
                '''
```