SAE S2.02 – Rapport pour la ressource Graphes

SAE S2.02 – Rapport pour la ressource Graphes

SAE S2.02 – Rapport pour la ressource Graphes

KIECKEN, SADY, GAIENIER; Groupe : D2

Version 1

Étude d’un premier exemple

Énumérer tous les appariements acceptables (c’est à dire qui asocient des adolescents compatibles) pour les données de l’Exemple 1, en supposant que les français rendent visite aux italiens.

Appariements acceptables :

A avec X	A avec Y	A avec z	B avec X	B avec Z	C avec X	C avec Y	C avec Z

Justifier pourquoi l’appariement optimal est Bellatrix–Xolag, Adonia–Zander, et Callista–Yak; cette explication ne doit pas

parler de graphes, mais se baser uniquement sur les données du problème.

Bellatrix–Xolag ont un point commun “culture” et Bellatrix–Zander n’en ont aucun.

Adonia–Zander ont un point commun “technology”, Adonia–Xolag ont aussi un point commun mais cela n’est pas possible car Xolag déjà affecté et Adonia–Yak n’ont aucun point commun.

Callista–Yak ont deux points commun “science,reading” et Callista n’a de point commun avec personnes d’autres

Modélisation de l’exemple

Donner un graphe qui modélise l’Exemple 1, plus précisément, la matrice d’adjacence de ce graphe. Expliquez comment vous avez choisi le poids pour chacune des arêtes.

	X	Y	Z
A	-1.0	0.0	-1.0
B	-1.0	100.0	0.0
C	0.0	-2.0	0.0

Poids initial pour chaque : 0.0

Pour A-X, on enlève 1 car ils ont un un point commun. Pour A-Y, on y touche pas car pas de points commun. Pour A-Z, on enlève 1 car ils ont un un point commun.

Pour B-X, on enlève 1 car ils ont un point commun. Pour B-Y, on ajoute 100 car B est allergique aux animaux. Pour B-Z, on y touche pas car pas de points commun.

Pour C-X, on y touche pas car pas de points commun. Pour C-Y, on enlève 2 car ils ont deux points commun. Pour C-Z, on y touche pas car pas de points commun.

Modélisation pour la Version 1

Décrire une modélisation générale pour la Version 1. C'est à dire, donner une formule ou une description précise qui décrit comment, étant donné un adolescent hôte et un adolescent visiteur, on détermine le poids de l'arête entre ces deux adolescents en fonction des critères considérés dans la Version 1.

Le poids est initialisé à 0.0. Le poids de l'arête entre deux adolescents est calculé en fonction des critères suivants : - Nous testons si les deux adolescents sont compatibles (Uniquement en se basant sur l'allergie aux animaux), si non, nous ajoutons 99 au poids de l'arête. - Nous testons si les deux adolescents ont un point commun, si oui, nous retirons 1 au poids de l'arête.

Implémentation de la Version 1

Cette partie du travail sera évaluée à partir du code. Implémenter la fonction weight de la classe AffectationUtil en suivant la modélisation proposée. Puis, implémenter une classe TestAffectationVersion1 qui montre que votre implémentation est correcte. La classe de test doit contenir au moins une méthode de test comme ceci: - créer les adolescents de l'Exemple 1 - construire le graphe modèle pour ces adolescents; le graphe sera de type fr.ulille.but.GrahpeNonOrienteValue - calculer l'affectation optimale en utilisant la classe fr.ulille.but.CalculAffectation - écrire des assertions (assertEquals ...) qui vérifient que le résultat de l'affectation calculé à l'étape précédente est bien celui attendu

Si vous n’êtes pas à l’aise avec les tests unitaires, votre classe TestAffectationVersion1 peut contenir une méthode main à la palce de la méthode de test, dans ce cas vous afficherez dans le terminal l’appariement résultat.

Exemple de vérification de l’incompatibilité

*Cet exemple va mettre au défi votre modèle vis à vis de la prise en compte de l’incompatibilité entre adolescents

Récupérez sur Moodle le fichier de données `compatibilityVsHobbies.csv`. Expliquez quelle est sa particularité de cet exemple. Écrire la méthode de test qui test qui change cet exemple, construit le graphe modèle, calcule l’affectation, et finalement vérifie qu’aucune paire d’adolescents non compatibles n’a été construite par l’algorithme.*

Version 2

Sera évaluée à partir du tag git `Graphes-v2`

Exemple minimal pour la gestion de l’historique

Présenter un exemple minimal qui est pertinent pour tester l’historique. L’exemple contiendra: - huit adolescents de deux pays différents tels que - certains des adolescents expriment des préférences d’historique (critère HISTORY). Toutes les valeurs possibles pour ce critère doivent être présentes - aucun des adolescents n’est allergique aux animaux en aucun n’a exprimé de passe-temps, ainsi pour l’instant on peut se concentrer uniquement sur la gestion de l’historique - un

historique, c’est à dire une collection de paires d’adolescents qui étaient correspondants l’année passée. Ces paires doivent permettre d’illustrer les différents cas de figure qui peuvent se présenter par rapport aux contraintes d’historique et les huit adolescents

Liste des adolescents : FORENAME; NAME; COUNTRY; BIRTH_DATE; GUEST_ANIMAL_ALLERGY; HOST_HAS_ANIMAL; GUEST_FOOD; HOST_FOOD; HOBBIES; GENDER; PAIR_GENDER; HISTORY ItaTeen; A; ITALY; ; ; ; ; ; ; ; other; ItaTeen; B; ITALY; ; ; ; ; ; ; ; ; ItaTeen; C; ITALY; ; ; ; ; ; ; ; same; ItaTeen; D; ITALY; ; ; ; ; ; ; ; same; GerTeen; E; GERMANY; ; ; ; ; ; ; ; ; GerTeen; F; GERMANY; ; ; ; ; ; ; ; other; GerTeen; G; GERMANY; ; ; ; ; ; ; ; same; GerTeen; H; GERMANY; ; ; ; ; ; ; ; other;

Historique : A;E B;F C;G D;H

Puis, donner un appariement optimal qui tient compte des données d’historique, et expliquer pourquoi il est optimal. L’explication ne doit pas parler des graphes, mais uniquement des adolescents et les critères exprimés.

L’appariement optimal est le suivant : - A;F A possède comme critère HISTORY : other, il ne peux donc pas être en pair avec E F possède comme critère HISTORY : other, il ne peux donc pas être en pair avec B

B;H B n’a pas de critère HISTORY H possède comme critère HISTORY : other, il ne peux donc pas être en pair avec D	C;G C possède comme critère HISTORY : same, G aussi, et ils étaient déjà en pair l’année dernière.	D;E D possède comme critère HISTORY : same, seulement H possède comme critère HISTORY : other, il ne peuvent donc pas être en pair ensemble E n’a pas de critère HISTORY

Deuxième exemple pour la gestion d’historique

Modifiez l’exemple précédent en ajoutant des préférences liées aux passe-temps. Donnez l’appariement que vous considérez optimal dans ce cas. En particulier, expliquez comment vous comptez combiner une éventuelle affinité liée à l’historique avec l’affinité liée aux passe-temps. Rappelons que l’historique peut compter comme une contrainte rédhibitoire ou comme une préférence, voir le sujet pour plus de précisions.

FORENAME; NAME; COUNTRY; BIRTH_DATE; GUEST_ANIMAL_ALLERGY; HOST_HAS_ANIMAL; GUEST_FOOD; HOST_FOOD; HOBBIES; GENDER; PAIR_GENDER; HISTORY
ItaTeen; A; ITALY; ; ; ; ; ; science,culture,reading; ; ; other;
ItaTeen; B; ITALY; ; ; ; ; ; ; ; ; ; ItaTeen; C; ITALY; ; ; ; ; ; technology; ; ; same; ItaTeen; D; ITALY; ; ; ; ; ; science,language; ; ; same; GerTeen; E; GERMANY; ; ; ; ; ; technology,language;; ; ; GerTeen; F; GERMANY; ; ; ; ; ; culture,technology;; ; other; GerTeen; G; GERMANY; ; ; ; ; ; culture; ; ; same; GerTeen; H; GERMANY; ; ; ; ; ; science,culture,reading; ; ; other;

Donner l’appariement que vous considérez optimal dans ce deuxième exemple, toujours sans parler de graphes.

L’appariement optimal est le suivant : - A;H A possède comme critère HISTORY : other, il ne peux donc pas être en pair avec E H possède comme critère HISTORY : other, il ne peux donc pas être en pair avec D A possède comme critère HOBBIES : science,culture,reading, H possède comme critère HOBBIES : science,culture,reading, ils ont donc trois critères en commun.

B;E B n’a pas de critère HISTORY E n’a pas de critère

C;G C possède comme critère HISTORY : same, G

D;F D possède comme critère HISTORY : same, seulement H possède comme critère HISTORY : other, il ne peuvent donc pas être en pair ensemble F possède comme critère HISTORY : other, il ne peux

HISTORY B et E n’ont pas de critère HOBBIES en commun.	aussi, et ils étaient déjà en pair l’année dernière. Ils doivent donc être en pair ensemble.	donc pas être en pair avec B D possède comme critère HOBBIES : science,language, F possède comme critère HOBBIES : culture,technology, ils n’ont donc pas de critère en commun.

Gestion de l’historique : - Si les deux adolescents étaient déjà en pair l’année dernière et qu’ils ont tous les deux le critère HISTORY : same, alors ils doivent être en pair ensemble. - Si les deux adolescents étaient déjà en pair l’année dernière et que l’un des deux a le critère HISTORY : other, alors ils ne peuvent pas être en pair ensemble. - Si les deux adolescents n’étaient pas en pair l’année dernière, alors ils peuvent être en pair ensemble. - Si les deux adolescents étaient déjà en pair l’année dernière et qu’ils ne possèdent ni l’un ni l’autre de critère HISTORY, alors ils peuvent être en pair ensemble.

Modélisation pour les exemples

Pour chacun des deux exemples précédents, donnez un graphe (donné par sa matrice d’adjacence) tel que l’affectation minimale dans ce graphe correspond à l’appariement optimal identifié plus haut. Expliquez comment vous avez choisi le poids pour chacune des arêtes.

Exemple 1 :

	E	F	G	H
A	100.0	0.0	0.0	0.0
B	0.0	100.0	0.0	0.0
C	0.0	0.0	-100.0	0.0
D	0.0	0.0	0.0	100.0

A et E sont incompatible due à l’historique, donc on met un poids de 100.0 sur l’arête A-E. A et F sont compatible et n’ont aucune affinité, donc on met un poids de 0.0 sur l’arête A-F. A et G sont compatible et n’ont aucune affinité, donc on met un

poids de 0.0 sur l’arête A-G. A et H sont compatible et n’ont aucune affinité, donc on met un poids de 0.0 sur l’arête A-H.

B et E sont compatible et n’ont aucune affinité, donc on met un poids de 0.0 sur l’arête B-E. B et F sont incompatible due à l’historique, donc on met un poids de 100.0 sur l’arête B-F. B et G sont compatible et n’ont aucune affinité, donc on met un poids de 0.0 sur l’arête B-G. B et H sont compatible et n’ont aucune affinité, donc on met un poids de 0.0 sur l’arête B-H.

C et E sont compatible et n’ont aucune affinité, donc on met un poids de 0.0 sur l’arête C-E. C et F sont compatible et n’ont aucune affinité, donc on met un poids de 0.0 sur l’arête C-F. C et G sont compatible et doivent être en pair due à l’historique, donc on met un poids de -100.0 sur l’arête C-G. C et H sont compatible et n’ont aucune affinité, donc on met un poids de 0.0 sur l’arête C-H.

D et E sont compatible et n’ont aucune affinité, donc on met un poids de 0.0 sur l’arête D-E. D et F sont compatible et n’ont aucune affinité, donc on met un poids de 0.0 sur l’arête D-F. D et G sont compatible et n’ont aucune affinité, donc on met un poids de 0.0 sur l’arête D-G. D et H sont incompatible due à l’historique, donc on met un poids de 100.0 sur l’arête D-H.

Exemple 2 :

	E	F	G	H
A	100.0	-1.0	-1.0	0.0
B	0.0	100.0	-1.0	0.0
C	-1.0	-1.0	-100.0	0.0
D	-1.0	0.0	0.0	100.0

Modélisation pour l’historique de la Version 2

Décrire une modélisation générale pour la Version 1. C’est à dire, donner une formule ou une description précise qui décrit comment, étant donné un adolescent hôte et un adolescent

visiteur, on détermine le poids de l'arête entre ces deux adolescents en fonction des critères considérés dans la Version 1. Décrire également comment vous construisez le graphe modèle à partir des données en entrée.

On peut modéliser l'historique de la manière suivante : - Si les deux adolescents étaient déjà en pair l'année dernière et qu'ils ont tous les deux le critère HISTORY : same, alors on met un poids de -100.0 sur l'arête entre ces deux adolescents. - Si les deux adolescents étaient déjà en pair l'année dernière et que l'un des deux a le critère HISTORY : other, alors on met un poids de 100.0 sur l'arête entre ces deux adolescents. - Si les deux adolescents n'étaient pas en pair l'année dernière, alors on ne touche pas au poids de l'arête entre ces deux adolescents. - Si les deux adolescents étaient déjà en pair l'année dernière et qu'ils ne possèdent ni l'un ni l'autre de critère HISTORY, alors on ne touche pas au poids de l'arête entre ces deux adolescents.

Puis on vérifie le nombre de passe temps commun entre chaque adolescents, et on enlève 1.0 au poids de l'arête entre ces deux adolescents pour chaque passe temps commun. (Selon les règles de la Version 1)

Implémentation de l'historique de la Version 2

Quelles fonctions de votre code avez-vous modifié pour prendre en compte le critère historique ? Donnez ici les noms des méthodes (et leur classe), à quoi elles servent, et quelles modifications vous avez apportées. Essayez d'être synthétique.

Nous allons modifier la fonction weight de la classe AffectationUtil.

Nous vérifions si les deux adolescents étaient déjà en pair l'année dernière, si c'est le cas, nous vérifions si les deux

adolescents ont une valeur dans le critère HISTORY si c’est le cas, nous appelons la focntion historyCompatibility de la classe Teenager qui renvoie true ou false. Si la fonction historyCompatibility renvoie true, alors on met un poids de -100.0 sur l’arête entre ces deux adolescents. Si la fonction historyCompatibility renvoie false, alors on met un poids de 100.0 sur l’arête entre ces deux adolescents.

Test pour l’historique de la Version 2

Créer la classe de TestAffectationVersion2 qui contiendra deux méthodes de test, une pour chacun des exemples. Chacune de ces méthodes doit avoir la même structure que pour TestAffectationVersion1, c’est à dire créer les données d’entrée (adolescents, historique), créer le graphe, calculer l’affectation, et tester que le résultat est comme attendu.

Prendre en compte les autres préférences

Pour chacun des autres critères d’affinité que vous décidez de prendre en compte, décrire comment vous changez la fonction weight de la classe AffectationUtil.

Pour chaque critère nous allons vérifier si le critère est bien défini pour chaque adolescent, si c’est le cas, nous appelons la fonction correspondante de la classe Teenager qui renvoie true ou false. Si la fonction renvoie true, alors on ne touche pas au poids de l’arête entre ces deux adolescents. Si la fonction renvoie false, alors on met un poids de 100.0 sur l’arête entre ces deux adolescents.

Quand aux affinités : - Pour la préférence de genre, nous vérifions si les deux adolescents ont le même genre, si c’est le cas, nous retirons 10.0 au poids de l’arête entre ces deux adolescents. - Pour la préférence de la différence d’âge,

L’incompatibilité en tant que malus

Proposer une formule ou une description précise qui explique comment calculer le poids d’une arête en considérant les incompatibilités comme des malus et les critères satisfaits comme des bonus. Implémenter cette formule dans une seconde méthode appelée `weightAdvanced`, ceci pour éviter de casser votre code. Puis, écrire une méthode de test qui permet d’illustrer le calcul d’affectation basé sur `weightAdvanced`. Vous pouvez également tester l’affectation en utilisant le fichier de données `incompatibilityVsBonus.csv`.