PROJETS RO 2017-2018 (feuille de route)

Vous allez travailler en équipes pour réaliser vos projets de Recherche Opérationnelle (RO). Nous faisons cela pour vous rapprocher de votre futur métier d'ingénieur. Jouez le jeu, cela en vaut la peine !

Organisation de l'enseignement de la RO : apprentissage par projets d'équipes.

. Séances : 20 séances réparties en 6 CTD et 14 TPs.

. Projets : 5 projets à réaliser :

o Le premier et le deuxième projet portent sur des problèmes qui se posent à un agent évoluant dans un environnement déterministe et non concurrentiel

o Dans le troisième projet, l'environnement est déterministe et concurrentiel

o Dans le quatrième projet l'environnement est stochastique (non déterministe) et non concurrentiel

o Dans le cinquième projet, l'environnement est déterministe et non concurrentiel

o Chaque projet débute par un CTD où l'enseignant donne des informations sur le type de problème à résoudre, sur sa modélisation mathématique et sur sa résolution.

. **Présentation des travaux**

: La sixième et dernière séance (CTD) est consacrée à la présentation orale du travail réalisé par chaque équipe.

. **Equipes** : chaque équipe est composée d'un demi groupe de TP, soit 6 étudiants (en général). Pour chaque projet, l'équipe désigne un responsable de projet et un responsable qualité

o Le responsable du projet dirige l'équipe, répartit les tâches à effectuer, s'assure du bon avancement des travaux. Un bon responsable de projet doit être un fin psychologue qui sait tirer le maximum de ses coéquipiers. Il donne le bon exemple, il sait communiquer.

o Le responsable qualité veille à la qualité des travaux (rapport, logiciel) effectués. Un bon rapport doit être concis et clair. Un bon logiciel doit être bien structuré, bien documenté et testé sur un grand nombre d'exemples.

. **Langage** : sauf exception justifiée, le langage de programmation est MATLAB.

. **Rapport de projet** (format pdf) :

Sur chaque rapport de projet devra être clairement expliquée la part de travail effectué par chaque membre de l'équipe

. Le responsable qualité est chargé de l'application de cette directive.

. **Notation** :

La note finale de Recherche Opérationnelle est la moyenne arithmétique des trois notes de projet (1+2 , 3+4, 5) et de la note de présentation orale des différents projets. Cette note est individualisée.

Premier projet : programmation linéaire en variables continues puis en variables entières

Ce premier projet se divise en deux :

1. Programmation de l'algorithme du simplexe (variables continues)

2. Modélisation mathématique d’un problème d’emploi du temps sous la forme d’un programme linéaire en variables entières et résolution du programme.

Deuxième projet : algorithme de Ford et Fulkerson

Réalisation d'un programme de résolution de problèmes de flot maximum et de tension maximum.

Date limite de remise du rapport des deux premiers projets : vendredi 27 octobre 2017, 18h

Troisième projet : Réalisation d'un programme de calcul d'une « bonne » stratégie dans le duopole de Cournot joué de façon itérée et non coopérative (environnement concurrentiel).

Quatrième projet : Réalisation d'un programme de gestion de stock (contexte stochastique non concurrentiel).

Date limite de remise du rapport des troisième et quatrième projets: XXX 2017, 18h

Cinquième projet (contexte robotique déterministe : Réalisation d'un programme permettant de déformer la trajectoire d’un robot pour éviter les collisions avec des obstacles fixes et mobiles ?

Date limite de remise du rapport du cinquième projet : XXX 2018, 18h

Exposé détaillé des projets et des travaux à effectuer

1. **Premier projet** (3 TPs)

Lors du premier TP, chaque étudiant doit programmer en Matlab, un grand standard de la programmation linéaire en variables continues : l'algorithme du simplexe exposé lors du 1er CTD.

A partir du deuxième TP, les étudiants travaillent en équipe.

L'objectif est de résoudre un problème d'emploi du temps :

Dans ce qui suit, on donne des valeurs numériques mais le problème est générique et devra être modélisé mathématiquement comme tel.

On cherche à construire automatiquement un emploi du temps.

Tous les cours durent 1 heure et 45 minutes. Du lundi au vendredi (d = 5) , il y a t = 4 créneaux par journée : 8h-9h45 , 10h15-12h , 14h-15h45 , 16h15-18h.

Il y a c = 2 promotions d'étudiants. Tous les étudiants d'une même promo suivent exactement les mêmes cours. Les 2 promotions ont les mêmes professeurs sauf pour les cours de maths, d'informatique et de sport. Il y a au total m = 8 professeurs : 2 professeurs de maths : Mme Droite pour la 1ère promotion et Mr Ellips pour la seconde promotion, 1 professeur de physique Mme Proton, 2 professeurs d'informatique Mr Pascal et Mme Dell, 1 professeur d'anglais Mr Young et 2 professeurs de sport Melle Gazelle et Mr Bigceps.

- Mme Droite assure 5 cours par semaine
- Mr Ellips, 4 cours
- Mme Proton, 3 cours pour la 1ère promo et 3 cours pour la 2ème
- Mr Pascal, 6 cours pour la promo 1

• Mme Ada, 6 cours pour la promo 2

• Mr Young, 3 cours pour la promo 1 et 3 cours pour la promo 2

• Les cours de sport ont lieu le jeudi après-midi de 14h à 16h.

Le premier créneau du lundi matin est réservé au partiel. Mr Ellips est indisponible le lundi matin. Mme Proton ne peut pas travailler le mercredi. Chaque promo ne doit pas avoir plus d'un cours d'une même matière dans la même journée à l'exception des cours d'informatique où le nombre de cours dans une même journée ne doit pas dépasser deux.

Enfin, l'EDT doit présenter le moins de trous possibles.

• Modéliser sous la forme d'un Programme Linéaire en variables Entières (PLE) le problème de l'emploi du temps ci-dessus. Attention le problème est générique, le PLE doit l'être aussi

• Résoudre le programme PLE générique sur l'exemple numérique ci-dessus en utilisant le programme intlinprog de MATLAB. Produire une feuille d'EDT sous forme de tableau.

• Evaluer les temps de calcul en fonction des données des problèmes (nombre de variables et de contraintes).

2**. Deuxième projet** (2 TPs)

Le travail consiste à :

1. Programmer en Matlab l'algorithme de Ford et Fulkerson (FF) pour les flots puis pour les tensions

2. Modéliser le problème de trafic maritime (cf. poly) sous la forme d'un problème de flot maximum. Le résoudre par l'algorithme FF pour les flots.

3. Modéliser le problème de plus court chemin sous la forme d’un problème de tension maximum

(cf. poly). Le résoudre par l’algorithme FF pour les tensions.

4. Modéliser le problème d'ordonnancement (cf. poly) sous la forme d'un problème de tension maximum. Le résoudre par l'algorithme FF pour les tensions.

3. **Troisième projet**: duopole de Cournot, théorie des jeux (2 TPs)

L'objectif est de déterminer une stratégie efficace S* (algorithme codé en Matlab) pour jouer au

Duopole de Cournot (cf. poly) dans un cadre non coopératif.

Pour cela on testera S* face aux stratégies suivantes :

– stratégie coopérative

– stratégie non-coopérative

– stratégie de Stackelberg

– toute stratégie élaborée par une autre équipe

– S* elle-même

4. **Quatrième projet**: gestion de stock, chaînes de Markov (2 TPs)

On considère un magasin proposant à la vente un article particulier.

Les commandes des articles au fournisseur s'effectuent à la semaine.

Les nombres d'articles Dk demandés chaque semaine par les clients du magasin sont vus comme des réalisations de variables aléatoires indépendantes et de même loi. Cette loi peut être estimée statistiquement et elle est supposée connue.

Pour une semaine donnée, on note : pk la probabilité que k articles soient demandés et rk = 1 – p0 - … - pk la probabilité que plus de k articles soient demandés.

Le stock maximal d'articles en magasin est de S.

Le coût de stockage est de C1 par article et par semaine (toute semaine commencée est due en entier.

Le coût de pénurie est de C2 par article (souvent C2 = revenu – coût de commande - coût de stockage = v – C4 - C1)

Le coût de la commande de x articles est égal à C3 + C4 x si x > 0 et nul si x = 0.

Le revenu est de v par article vendu (v > C1 + C4) .

A la fin de chaque semaine, le responsable du stock décide – de ne pas commander de nouveaux articles s'il lui en reste au moins s en stock – de reconstituer le stock maximal de S articles s'il lui en reste strictement moins de s.

On note Xn le nombre d'articles restant en stock à la fin de la nième semaine. On remarque que la suite (Xn) est une chaîne de Markov à temps discret homogène et finie à valeurs dans 0,1,...,S.

L'objectif est maintenant de maximiser le revenu moyen par semaine B(s,S) en déterminant le meilleur couple (s,S) possible.

Pour cela :

1. On déterminera la matrice de transition P de Xn

2. On montrera que cette chaîne est convergente et on déterminera la distribution limite (PI(i)i=0...S)

3. On en déduira, en régime permanent, le revenu moyen par semaine B(s,S)

Application numérique

: C1 = 5 C2 = 10 C3 = 10 C4 = 5 v = 20. Tester sur d’autres valeurs.

5. **Cinquième projet**: simulation navigation robotique (5 TPs)

Réalisation d'un programme MATLAB calculant et simulant les trajectoires optimisées d'un robot mobile se déplaçant dans une scène plane encombrée d'obstacles fixes et mobiles. Tester la trajectoire sur plusieurs scènes statiques et dynamiques.

Par trajectoire optimisée, il faut comprendre une trajectoire réalisable, sans collision avec les obstacles fixes et mobiles, qui minimise le temps nécessaire pour que le robot aille d’un point à un autre.

Pour calculer une trajectoire optimisée, on se sert du logiciel PLATINE (fournie)

Le travail va consister :

1. A construire une scène dans laquelle va se déplacer le robot. Une scène est un rectangle à l’intérieur duquel se trouvent des obstacles fixes.

Pour créer une scène, lancer PLATINE2 et taper 2.

2. Une fois la scène construite, on calcule une trajectoire optimisée de notre robot appelé

MICKEY en lançant PLATINE2 et en tapant 3. Le programme calcule un chemin sous la forme d’une courbe NURBS qui est libre d’obstacles fixes. On peut aussi donner une vitesse de parcours de ce chemin.

3. Simulation de la trajectoire : en tapant 6, on obtient une simulation du déplacement du robot MICKEY à partir de la scène et de la trajectoire précédemment définies.

4. On va maintenant introduire de nouveaux robots qui vont se déplacer dans la scène : ce seront nos obstacles mobiles

. Pour cela on utilise le sous-programme de création d'une nouvelle trajectoire en tapant 3.

A noter que les trajectoires de ces nouveaux robots, calculées par PLATINE, fournissent des trajectoires qui ne rencontrent pas les obstacles fixes mais qui peuvent en revanche être en collision avec les autres obstacles mobiles et Mickey !

5. Une fois les obstacles mobiles créées, on lance à nouveau une simulation qui permet de calculer les déplacements de tous les robots : MICKEY et les autres. En général, il y aura des collisions entre MICKEY et les obstacles mobiles. C'est à ce niveau que vous allez intervenir pour calculer une trajectoire optimisée. Vous devez programmer un algorithme d'optimisation (algorithme génétique, recuit simulé, etc.) qui va recalculer la trajectoire de MICKEY. Pour cela, vous devez modifier certaines des variables géométriques et/ou cinématiques suivantes

a. Les variables géométriques sont celles qui définissent le chemin parcouru par MICKEY ; ce sont les points de contrôle de la NURBS et leurs poids.

b. Les variables cinématiques déterminent les valeurs des vitesses de MICKEY lorsqu'il parcourt son chemin. Dans un premier temps, on pourra supposer que MICKEY se déplace à vitesse constante sur sa trajectoire.

L'objectif est de trouver une nouvelle trajectoire optimisée pour MICKEY, sans collision, respectant les contraintes de courbure et qui minimise le temps de parcours. A cet effet, il faudra en général déformer la trajectoire initialement trouvée (celle qui évitait les obstacles fixes) …