Université Larbi Ben Mhidi Niveau : 1ère année Master AD

Département MI

Année : 2025-2026

Examen en Méthodes algorithmiques

Question à Choix Unique : Choisissez la bonne réponse **(6pts)**

Q1. Quelle condition est indispensable pour appliquer correctement le paradigme diviser pour régner ?
A)Sous-problèmes de taille égale
B) Sous-problèmes indépendants
C)Optimalité locale

Q2. Un algorithme Las Vegas garantit :
A) Un temps polynomial
B) Une solution exacte
C) Une erreur bornée
D) Une approximation contrôlée

Q3. La programmation dynamique est adaptée lorsque :
A) Les décisions locales suffisent à garantir une solution optimale
B) Le problème peut être décomposé en sous-problèmes qui se recouvrent et vérifie le principe d'optimalité
C) L'algorithme repose sur des choix aléatoires pour explorer l'espace des solutions

Q4. Un FPTAS est caractérisé par :
A) Un temps d'exécution polynomial uniquement en fonction de la taille de l'instance
B) Un temps d'exécution polynomial à la fois en la taille de l'instance et en $1/\varepsilon$
C) Temps exponentiel en $1/\varepsilon$

Q5. Un algorithme Monte Carlo est caractérisé par :
A) Temps aléatoire et exactitude garantie
B) Temps borné et erreur possible
D) Temps déterministe et exactitude garantie

Q6. Un algorithme d'approximation de facteur 2 signifie que :
A) L'erreur absolue est ≤ 2
B) La solution est au plus deux fois l'optimum
C) La solution est à distance 2 de l'optimum
D) La solution est exacte dans 50 % des cas

Tournez la page

Exercice 1 : **(8pts)**

Soit un graphe G=(V,E)

Une couverture de domination est un sous-ensemble de sommets $D \subseteq V$ tel que chaque sommet $v \in V$ appartient à D ou est adjacent à au moins un sommet de D

1. Proposez un algorithme d'approximation pour ce problème avec un facteur d'approximation de 2.
2. Expliquez pourquoi l'algorithme proposé a un facteur d'approximation de 2
3. L'algorithme proposé est-il un algorithme glouton ? Justifiez votre réponse

Exercice 2 : **(4pts)**

En appliquant le principe de la programmation dynamique trouver la longueur de la plus longue sous-séquence commune entre deux chaines de caractères.

Exercice 3 : **(2pts)**

Pour chaque méthode algorithmique ci-dessous, cochez : le ou les types de problèmes pour lesquels cette méthode est adaptée :

Méthode algorithmique							
Type de problèmes adaptés		Méthode gloutonne	Diviser pour régner	Programmation dynamique	Backtracking	Méthodes probabilistes	Algorithmes d'approximation
	Fibonacci						
	rendu de monnaie						
	test de primalité						
	l'arbre de Steiner minimum						
	sudoku						
	recherche binaire						

Bonne chance

Université Larbi Ben Mhidi
Département MI
Niveau : 1ère année Master AD
Année : 2025-2026

Corrigé type de l'Examen en Méthodes algorithmiques

Questions A Choix Unique :

Q1. Quelle condition est indispensable pour appliquer correctement le paradigme diviser pour régner ?
A)Sous-problèmes de taille égale
B) Sous-problèmes indépendants
C)Optimalité locale

Q2. Un algorithme Las Vegas garantit :
A) Un temps polynomial
B) Une solution exacte
C) Une erreur bornée
D) Une approximation contrôlée

Q3. La programmation dynamique est adaptée lorsque :
A) Les décisions locales suffisent à garantir une solution optimale
B) Le problème peut être décomposé en sous-problèmes qui se recouvrent et vérifie le principe d'optimalité
C) L'algorithme repose sur des choix aléatoires pour explorer l'espace des solutions
D) Le problème ne possède aucune structure récursive

Q4. Un FPTAS est caractérisé par :
A) Un temps d'exécution polynomial uniquement en fonction de la taille de l'instance
B) Un temps d'exécution polynomial à la fois en la taille de l'instance et en $1/\varepsilon$
C) Temps exponentiel en $1/\varepsilon$

Q5. Un algorithme Monte Carlo est caractérisé par :
A) Temps aléatoire et exactitude garantie
B) Temps borné et erreur possible
D) Temps déterministe et exactitude garantie

Q6. Un algorithme d'approximation de facteur 2 signifie que :
A) L'erreur absolue est ≤ 2
B) La solution est au plus deux fois l'optimum
C) La solution est à distance 2 de l'optimum
D) La solution est exacte dans 50 % des cas

Exercice 2 :

1. L'algorithme **(4pts)**

```
Input : graphe non orienté G=(V,E)
C ← ∅        // couverture construite
E' ← E       // arêtes encore non couvertes

Tant que E' ≠ ∅ faire
   choisir une arête (u,v) ∈ E'
   ajouter u et v à C
   supprimer de E' toutes les arêtes incidentes à u ou à v
Retourner C
```

2. Approximation factorielle **2 : (2pts)**
Chaque arête choisie nécessite au moins un sommet dans la solution optimale D∗. Comme on ajoute 2 sommets pour couvrir une arête, on a au maximum 2 fois la taille optimal

3. Oui, cet algorithme est glouton **(0,5pts)**

Justification :**(1,5pts)**

- À chaque itération, l'algorithme choisit une arête non encore couverte et ajoute les deux sommets qui la composent dans l'ensemble C.
- Ce choix est fait localement, sans tenir compte des conséquences sur le reste du graphe.
- L'algorithme cherche à maximiser la couverture immédiate (couvrir au moins une arête à chaque étape) avec un gain local, ce qui correspond exactement au principe d'un algorithme glouton.

La solution finale peut ne pas être optimale, mais elle garantit un facteur d'approximation de 2, ce qui est typique des algorithmes gloutons avec approximation.

Exercice 1 : (4pts)

En appliquant le principe de la programmation dynamique trouver la longueur de la plus longue sous-séquence commune entre deux chaines de caractères :

```
def LSC(X, Y):
    m = len(X)
    n = len(Y)
    table = [[0] * (n + 1) for _ in range(m + 1)]
    for i in range(1, m + 1):
        for j in range(1, n + 1):
            if X[i - 1] == Y[j - 1]:
                table[i][j] = table[i - 1][j - 1] + 1
            else:
                table[i][j] = max(table[i - 1][j], table[i][j - 1])
    return table[m][n]
```

Exercice 3 : (2 pts)

Pour chaque méthode algorithmique ci-dessous, cochez : le ou les types de problèmes pour lesquels cette méthode est adaptée :

Méthode algorithmique							
Type de problèmes adaptés		Méthode gloutonne	Diviser pour régner	Programmation dynamique	Backtracking	Méthodes probabilistes	Algorithmes d'approximation
	Fibonacci			**X**			
	rendu de monnaie	**X**					
	test de primalité					**X**	
	l'arbre de Steiner minimum						**X**
	sudoku				**X**		
	recherche binaire		**X**				