

# Proposition d'amélioration de Contenu du Module

## Résumé et table de matières

Module : Complexité et Optimisation

Niveau : M1

### Résumé

L'objectif de ce module est de comparer les performances des algorithmes qui effectuent les mêmes tâches pour trouver un bon algorithme. Le contenu du module est consacré d'une part, à la théorie de la complexité des algorithmes, complexité des problèmes, Grandeur des fonctions et mesure de performance. Et d'autre part, à la présentation des différentes méthodes d'optimisation combinatoire.

La réalisation de cet objectif nécessite des connaissances de base qui concerne le domaine de l'analyse d'algorithmique et les techniques d'optimisation de la complexité présentées par : les méthodes de calcul de complexité des algorithmes (coût uniforme et coût logarithmique) et les classes de complexité des problèmes, l'optimisation, la dichotomie et la stratégie D&C.

Le contenu de ce module permet de présenter la notion d'optimisation combinatoire, les méthodes exactes : Programmation Dynamique, Branch&Bound et Recherche Arborescente ; ainsi que les méthodes approchées de résolution d'un problème d'optimisation, et en particulier, l'étude des heuristiques spécialisées : Algorithme Glouton, les méthodes d'exploration avec information : Algorithmes de Recherche Locale, A\*, Hill Climbing.

# Plan de contenu du module

## Objectifs de cours

### **CHAPITRE 1. Complexité des algorithmes et mesure de performance**

#### **Partie 1. Théorie de la complexité**

1. Critères d'évaluation de performance de l'algorithme
2. Complexité d'un algorithme
3. Complexité d'un problème
4. Type de complexité
5. Notion d'optimalité

#### **Partie 2. Grandeur des fonctions, la Complexité Asymptotique et Calcul de complexité des algorithmes itératifs et récursifs**

1. Optimisation d'exécution de la récursivité
2. C'est quoi les notions de Landau ?
3. Classes de complexité et Notations asymptotique
4. Calcul de complexité des algorithmes itératifs et récursifs
5. Master théorème et Équations de récurrences

### **CHAPITRE 2. Classes et complexité d'un problème**

1. Complexité d'un problème, Présentation et définitions
2. Trois méthodes pour trouver une borne inférieure
3. Classes de complexité d'un problème
  - 3.1. Classes de complexité en temps : Classe P, NP et EXPTIME
  - 3.2. Classes de complexité en espace
4. NP-complétude
5. Réduction d'un problème Q à un problème  $\Pi$

### **CHAPITRE 3. Optimisation de la complexité et la stratégie Diviser pour Régner (D&C)**

1. Méthode Diviser pour Régner
  - 1.1. Diviser pour Régner : tri fusion
  - 1.2. Diviser pour Régner : produit de deux matrices
2. Optimisation de la complexité et la dichotomie
  - 2.1. L'élément majoritaire et la dichotomie
  - 2.2. Optimisation et Algorithmes Avancés, comparaison de complexité
    - 2.2.1. Algorithmes de tri
    - 2.2.2. Algorithmes de recherche

### **CHAPITRE 4. Optimisation combinatoire, méthodes exactes**

1. Définition d'optimisation
2. Définition de problème d'optimisation
3. Programmation dynamique
4. Branch and bound
5. Recherche arborescente, l'exploration sans information

### **CHAPITRE 5. Méthodes approchées, heuristiques spécialisées**

1. Algorithme Glouton
2. L'exploration avec information
  - 2.1. Algorithme Recherche Locale
  - 2.2. Algorithme A\*
  - 2.3. Algorithme Hill Climbing

# Table des matières détaillées

## Objectifs de cours

### **CHAPITRE 1. Complexité des algorithmes et mesure de performance**

#### **Partie 1. Théorie de la complexité**

1. Qualités d'un bon algorithme  
Quelques problèmes algorithmiques classiques  
Qualités d'un bon algorithme
2. Analyse et notions de complexité
  - 2.1. Critères d'évaluation de performance de l'algorithme
    - 2.1.1. Calcul de nombre d'opérations
    - 2.1.2. Mesure d'espace mémoire
  - 2.2. Coût d'un algorithme
  - 2.3. Complexité d'un algorithme
  - 2.4. Complexité d'un problème
  - 2.5. Comment calculer la complexité d'un algorithme ?
  - 2.6. Type de complexité
    - 2.6.1. Complexités temporelle et spatiale d'un algorithme
    - 2.6.2. Complexités pratique et théorique
  - 2.7. Notion d'optimalité

#### **Partie 2. Grandeur des fonctions, la Complexité Asymptotique et Calcul de complexité des algorithmes itératifs et récursifs**

1. Optimisation d'exécution de la récursivité
  - 1.1. Définition et types de récursivité
  - 1.2. Critères de terminaison pour un bon algorithme récursif
  - 1.3. Comment gérer la récursivité ?
    - 1.3.1. Gestion de la récursivité sans la pile : Récursivité Terminale (RT)
    - 1.3.2. Gestion de la récursivité avec la pile : Récursivité Enveloppée (RE)
  - 1.4. Optimisation d'exécution de la récursivité
    - 1.4.1. Comparaison d'algorithmes
2. C'est quoi les notions de Landau ?
  - 2.1. Classes de complexité
  - 2.2. Notations asymptotique
    - 2.2.1. Domination asymptotique
    - 2.2.2. Borne Inferieur, Notation  $\Omega$
    - 2.2.3. Equivalence asymptotique
  - 2.3. Calcul de complexité asymptotique
3. Calcul de complexité des algorithmes itératifs et récursifs
  - 3.1. Coût uniforme et coût logarithmique
    - 3.1.1. A propos de la notation  $O$
  - 3.2. Calcul de complexité des algorithmes itératifs
  - 3.3. Calcul de complexité des algorithmes récursifs
    - 3.3.1. Master théorème et Équations de récurrences

### **CHAPITRE 2. Classes et complexité d'un problème**

6. Complexité d'un problème, Présentation et définitions
7. Trois méthodes pour trouver une borne inférieure
8. Classes de complexité d'un problème
  - 3.1. Classes de complexité en temps
    - 3.1.1. Classe P

3.1.2. NP

3.1.3. EXPTIME

3.2. Classes de complexité en espace

9. NP-complétude

10. Réduction d'un problème Q à un problème  $\Pi$

### **CHAPITRE 3. Optimisation de la complexité et la stratégie Diviser pour Régner (Divide&Conquer)**

2. Méthode Diviser pour Régner

1.1. Diviser pour Régner : tri fusion

1.2. Diviser pour Régner : produit de deux matrices

2. Optimisation de la complexité et la dichotomie

2.1. L'élément majoritaire et la dichotomie

2.2. Optimisation et Algorithmes Avancés, comparaison de complexité

2.2.1. Algorithmes de tri

2.2.2. Algorithmes de recherche

### **CHAPITRE 4. Optimisation combinatoire, méthodes exactes**

1. Définitions

1.1. Définition d'optimisation

1.2. Définition de problème d'optimisation

2. Programmation dynamique

3. Branch and bound

4. Recherche arborescente, l'exploration sans information

1.1. Structures de données pour les graphes

1.2. Parcours de graphe

1.2.1. Parcours en largeur

1.2.2. Parcours en profondeur

1.3. Backtracking (retour sur trace)

1.3.1. Application du backtracking dans les arbres de jeux

1.3.1.1. Principe de MIN-MAX

### **CHAPITRE 5. Méthodes approchées, heuristiques spécialisées**

1. Algorithme Glouton

2. L'exploration avec information

3.1. Définitions

3.2. Algorithme Recherche Locale

3.3. Algorithme A\*

3.4. Algorithme Hill Climbing

### **Références, livre dans la complexité**

1. Sanjeev Arora and Boaz Barak, Computational Complexity: A Modern Approach, January 2007.

2. Olivier Bournez, Fondements de l'informatique Logique, modèles, et calculs, Cours INF412 de l'Ecole Polytechnique, Version du 18 juillet 2018.