

algorithmique

P. BERLIOUX, Ph. BIZARD

**2 structures de données
et algorithmes de recherche**

DUNOD

informatique

2-005-4-2/1

algorithmique

2 structures de données et algorithmes de recherche



par

Pierre BERLIOUX

Maitre de Conférences à l'Institut
national polytechnique de Grenoble (ENSIMAG)

Philippe BIZARD

Assistant à l'Université
Joseph Fourier de Grenoble

Ouvrage publié avec le concours des ministères
de l'Éducation nationale
et de la Recherche et de la Technologie.
Programme mobilisateur « promotion du français
langue scientifique et diffusion
de la culture scientifique et technique ».

DUNOD

informatique

TABLE DES MATIERES

	<i>page</i>
Première partie. Structures de données	1
1. Programmes. Preuves de programmes. Structures de données	3
1.1. Programmes	3
1.1.1. Variables et instructions	3
1.1.2. Procédures et paramètres	4
1.2. Fonction calculée par un programme	6
1.2.1. Définitions	6
1.2.2. Fonction calculée par une instruction composée	7
1.3. Complexité d'un programme	10
1.3.1. Définitions	10
1.3.2. Analyse de la complexité d'un programme	12
1.4. Conditions (assertions) sur les variables d'un programme	14
1.4.1. Assertions sur les valeurs des variables	14
1.4.2. Syntaxe des assertions	15
1.4.3. Assertions et ensembles des états des variables	17
1.4.4. Substitution	17
1.5. Correction d'un programme	19
1.5.1. Définitions	19
1.5.2. Plus faibles préconditions	20
1.6. La logique de Hoare	22
1.6.1. Présentation	22
1.6.2. Axiomes et règles de déduction	23
1.7. Propriété	24

1.8. Programmes équivalents	26
1.8.1. Programmes équivalents : première définition	26
1.8.2. Programmes équivalents : deuxième définition	26
1.8.3. Programmes équivalents : troisième définition	27
1.8.4. Propositions	28
1.9. Approximation d'un programme par un autre programme	29
1.9.1. Définitions	29
1.9.2. Propositions	30
1.9.3. Programmation d'une opération	31
1.10. Extension aux opérations non déterministes	32
1.10.1. Programmes non déterministes	32
1.10.2. Programmation d'une opération non déterministe	33
1.11. Structures de données	34
1.11.1. Définition	34
1.11.2. Programmation d'une structure de données	35
1.11.3. Les suites et les ensembles	37
2. Les suites	39
2.1. Introduction	39
2.2. La pile	42
2.2.1. Introduction	42
2.2.2. Axiomes pour la correction partielle des opérations de la pile	42
2.2.3. Exemples	43
2.2.4. Programmers de la pile	47
2.3. La file d'attente	52
2.3.1. Introduction	52
2.3.2. Axiomes pour la correction partielle des opérations de la file d'attente	52
2.3.3. Exemple	53
2.3.4. Exemple de programmation de la file d'attente : la file circulaire	54
2.4. Autres structures de données du type suite	57
3. Les ensembles	61
3.1. Introduction	61

Table des matières

3.2. Les dictionnaires	62
3.2.1. Opération de recherche	62
3.2.2. Opération d'insertion	63
3.2.3. Exemple	64
3.2.4. Programmations des opérations du dictionnaire	65
3.3. Dictionnaires avec les opérations d'initialisation et de recherche	66
3.3.1. Programmations avec un tableau	67
3.3.2. Programmations avec un arbre de recherche	69
a. Définitions	69
b. Programmation de l'opération chercher	70
c. Programmation de l'opération initialiser.	
Construction d'un arbre de recherche optimal	72
3.4. Dictionnaires avec insertion	79
3.4.1. Programmations avec un tableau	79
3.4.2. Représentation par un arbre binaire de recherche	81
a. Programmation de l'opération insérer	81
b. Analyse	82
c. Les arbres de recherche équilibrés	84
3.4.3. Programmations par adressage dispersé	89
3.4.4. Exercice : dictionnaires avec suppression	94
3.5. Structures de données avec choix	94
3.5.1. L'opération choisir	94
3.5.2. Structure de données avec les deux opérations choisir et insérer	95
4. Les files d'attente avec priorité	99
4.1. Définition de la structure de données	99
4.1.1. Introduction	99
4.1.2. Axiomes pour la correction partielle des opérations de la file d'attente avec priorité	100
4.1.3. Exemples	101
4.2. Programmations de la file d'attente avec priorité	106
4.2.1. Programmation par listes d'objets de même priorité	107
4.2.2. Programmation à l'aide d'un tas	108
4.2.3. Application : le tri à l'aide d'un tas (heapsort)	111

Deuxième partie. Algorithmes de recherche dans les graphes	115
5. Recherche dans les graphes. Introduction	117
5.1. Les graphes. Définitions	117
5.1.1. Graphe	117
5.1.2. Multigraphe	119
5.1.3. Graphe aux sommets étiquetés	120
5.1.4. Graphe aux arcs étiquetés	120
5.1.5. Graphe non orienté	121
5.1.6. Graphe dont les arcs issus de chaque sommet sont ordonnés	121
5.1.7. Chemins d'un graphe	122
5.1.8. Sommets et arcs accessibles. Sous-graphe	123
5.1.9. Circuits et cycles d'un graphe	123
5.1.10. Arbre	124
5.1.11. Arbres des chemins d'origine donnée d'un graphe	124
5.1.12. Chemins de longueur infinie	126
5.2. Les graphes. Structure de données	127
5.2.1. L'opération successeur	127
5.2.2. Programmation dans le cas des graphes finis	128
5.2.3. Notations pour les valeurs du type suite de sommets	130
5.3. Problèmes de parcours de graphes	131
6. Recherche d'un chemin dans un graphe	137
6.1. Définition d'une procédure générale de parcours de graphe : la procédure chemin	137
6.1.1. Énumération des chemins d'origine donnée d'un graphe	137
6.1.2. Programmation des procédures énumérer et initialiser	139
6.1.3. Transformation de la procédure chemin	142
6.1.4. Propriétés de la procédure chemin	143
6.1.5. Utilisation d'un dictionnaire des sommets extrémités des chemins déjà énumérés : la procédure chemin_dic	144

6.1.6.	Analyse des procédures chemin et chemin_dic	146
--------	---	-----

3 Représentation des chemins

6.1.6.	Analyse des procédures chemin et chemin_dic	146
	a. Représentation des chemins.	

	<i>Types d'algorithmes des procédures</i>	146
	<i>a. Paramètres de l'analyse</i>	146
	<i>b. Diverses implémentations du principe d'algorithmes et de la table de la recherche</i>	149
	<i>c. Étude de graphes particuliers</i>	151
	<i>d. Parcours « profondeur d'abord »</i>	152

6.2.1.	Définition des procédures profondeur et profondeur_dic. Propriétés de correction	152
6.2.2.	Analyse du parcours « profondeur d'abord »	157
6.2.3.	Parcours avec retour arrière	160
6.2.4.	Recherche d'un chemin de longueur minimale	163
	a. La procédure retour_arrière_itéré	163
	b. Analyse de la procédure retour_arrière_itéré	165
6.3.	Parcours « largeur d'abord »	167
6.3.1.	Les procédures largeur et largeur_dic. Propriétés de correction	167
6.3.2.	Analyse du parcours « largeur d'abord »	169
6.3.3.	Comparaison avec les parcours « profondeur d'abord » et avec retour arrière	171
6.4.	Conclusion	172
6.5.	Parcours heuristiques	173
6.5.1.	Introduction	173
6.5.2.	Fonctions heuristiques	175
6.5.3.	Parcours d'un graphe avec minimisation d'une fonction heuristique (les procédures heuristique et heuristique_dic)	178
6.5.4.	Propriétés de la procédure heuristique	179
	a. Correction partielle	179
	b. Equivalence avec la procédure générale chemin	180
	c. Parcours avec l'heuristique parfaite	180
	d. Fonctions heuristiques assurant la correction totale quand il existe un chemin de l'origine au but	182
	e. Fonctions heuristiques permettant de trouver un chemin de longueur presque minimale	183
6.5.5.	Analyse de la procédure heuristique	185
6.5.6.	Retour arrière itéré avec heuristique	185

7. Recherche d'un chemin de poids minimal dans un graphe	187
7.1. Introduction	187
7.1.1 Poids d'un chemin d'un graphe	187
7.1.2. Chemins de poids minimal dans un graphe	188
7.2. Parcours « le plus faible poids d'abord »	193
7.2.1. Les procédures faible_poids et faible_poids_dic. Propriétés de correction	193
7.2.2. Analyse	199
7.3. L'algorithme de Dijkstra	199
7.3.1. Introduction	199
7.3.2. Conditions sur la fonction poids	200
7.3.3. La procédure faible_poids_opt	201
7.3.4. Analyse	203
7.3.5. Utilisation d'un dictionnaire : la procédure faible_poids_opt_dic	204
7.4. Recherche heuristique d'un chemin de poids minimal	205
7.4.1. Principe	205
7.4.2. La procédure heuristique_opt	206
7.4.3. Propriétés de la procédure heuristique_opt	207
7.4.4. Utilisation d'un dictionnaire	210
a. Les algorithmes avec heuristique cohérente	210
b. La procédure heuristique_opt_dic	210
c. Propriétés de la procédure heuristique_opt_dic	211
7.4.5. Analyse des procédures heuristique_opt et heuristique_opt_dic	213
7.4.6. Optimisation de la taille de la mémoire : parcours heuristique avec retour arrière itéré	215
Annexe	221
Bibliographie	231