Master Parisien de Recherche Opérationnelle

ENSTA, Institut Polytechnique de Paris

Protection de la Biodiversité

Rapport pour Recherche Opérationnelle et Développement Durable (RODD)

Auteurs:

Ling Ma

Changmin Wu

Encadremant:

Amélie Lambert

TABLE OF CONTENTS

Page

LIST OF TABLES

LIST OF FIGURES

ABSTRACT

Dans ce projet, nous avons traité quatre problèmes sous le thème de la protection de la biodiversité: la sélection de réserves naturelles, la maîtrise des effets néfastes de engendrés par la fragmentation du paysage, le maintient de la diversité génétique et la exploitation écologique des forêts. Différentes techniques de programmation mathématique sont introduites pour simplifier les modélisations, notamment la linéarisation, l'optimisation combinatoire fractionnaire, approximation d'une fonction concave (logarithmique) par une fonction linéaire par morceaux, etc. Nous présentons donc dans ce rapport la modélisation de chaque problème, l'implémentation[1], le test sur des instances variées, l'analyse de la solution et de la sensibilité de la taille (limitation du modèle).

[1] `https://github.com/ChangminWu/rodd`

1. SÉLECTION DE RÉSERVE NATURELLES

Dans ce projet, nous étudions le problème de la sélection de réserve naturelles afin de stopper la perte de bio-diversité, mais en limitant son effet sur activités humaines.

1.1 Pré-modélisation

Supposons que l'on ait un ensemble d'espèces à protéger, $E = \{e_1, e_2, \ldots, e_p\}$, vivant sur un ensemble de parcelles $S = \{s_1, s_2, \ldots, s_n\}$. L'objet est de trouver un réserve optimal, qui est un sous-ensemble de parcelles, soit minimiser son aire tel que son effet sur activités humaines est le plus petit, soit maximiser le nombre d'espèces qui vivent dans le réserve. Donc on a deux modélisations suivants:

$$(P1)\begin{cases} \min & \sum_{i\in N} x_i \\ \text{s.t.} & \sum_{i\in S_k} x_i \geq 1 \quad k \in P \\ & x_i \in \{0,1\} \quad i \in N \end{cases}$$

$$(P2)\begin{cases} \max & \sum_{k\in P} y_k \\ \text{s.t.} & y_k \leq \sum_{i\in S_k} x_i \quad k \in P \\ & \sum_{i\in S_k} a_i x_i \leq B \quad k \in P \\ & x_i \in \{0,1\} \quad i \in N \\ & y_k \in \{0,1\} \quad k \in P \end{cases}$$

1.2 Modélisation

1.2.1 Description du problème

Voir le fichier donné.

1.2.2 Modélisation par un programme linéaire

$$(P)\begin{cases} \min & \sum_{ij\in N} c_{ij}y_{ij} & \\ \text{s.t.} & 9x_{ij} \leq \sum_{i'j'\in Nbr(ij)} y_{i'j'} & ij \in S \\ & 1-\prod_{ij\in S}(1-p_{k,ij}x_{ij}) \geq \alpha_k & k \in \text{Rare} \\ & 1-\prod_{ij\in S}(1-p_{l,ij}y_{ij}) \geq \alpha_l & l \in \text{Commune} \\ & x_{ij} \in \{0,1\} & ij \in S \\ & y_{ij} \in \{0,1\} & ij \in S \end{cases}$$

où c_{ij} est le coût de protection de la parcelle ij, $Nbr(ij)$ sont l'ensemble de la parcelle ij et les parcelles autour d'elle. $p_{k,ij}$ est la probabilité que l'espèce e_k présente dans la parcelle s_{ij}, survive dans cette parcelle si celle-ci est protégée. α_k est le contraint sur la probabilité de présence dans la réserve pour l'espèce e_k. y_{ij} représente si une parcelle est protégée et x_{ij} représente si elle est zone centrales.

On linéarise P par prendre logarithme sur les produits et on a

$$(PL)\begin{cases} \min & \sum_{ij\in N} c_{ij}y_{ij} & \\ \text{s.t.} & 9x_{ij} \le \sum_{i'j'\in Nbr(ij)} y_{i'j'} & ij \in S \\ & \sum_{ij\in S} \log(1-p_{k,ij}x_{ij}) \le \log(1-\alpha_k) & k \in \text{Rare} \\ & \sum_{ij\in S} \log(1-p_{l,ij}y_{ij}) \le \log(1-\alpha_l) & l \in \text{Commune} \\ & x_{ij} \in \{0,1\} & ij \in S \\ & y_{ij} \in \{0,1\} & ij \in S \end{cases}$$

1.3 Solution

On a étudié une instance de la taille 10×10 et 6 espèces (3 rares et 3 communes).

6	6	6	4	4	4	4	8	8	8
6	6	6	4	4	4	4	8	8	8
6	6	6	4	4	4	4	8	8	8
5	5	5	3	3	3	3	7	7	7
5	5	5	3	3	3	3	7	7	7
5	5	5	3	3	3	3	7	7	7
5	5	5	3	3	3	3	7	7	7
4	4	4	6	6	6	6	5	5	5
4	4	4	6	6	6	6	5	5	5
4	4	4	6	6	6	6	5	5	5

Figure 1.1.: Coût de protection de chaque parcelle

4 cas sont considérés:

- cas 1: $\alpha_k = 0.5$ $(k = 1, \ldots, 6)$

- cas 2: $\alpha_k = 0.9$ $(k = 1, \ldots, 3)$ et $\alpha_k = 0.5$ $(k = 4, \ldots, 6)$
- cas 3: $\alpha_k = 0.5$ $(k = 1, \ldots, 3)$ et $\alpha_k = 0.9$ $(k = 4, \ldots, 6)$
- cas 4: $\alpha_k = 0.8$ $(k = 1, \ldots, 3)$ et $\alpha_k = 0.6$ $(k = 4, \ldots, 6)$

Pour cas 1, on a la solution en Figure 1.2.

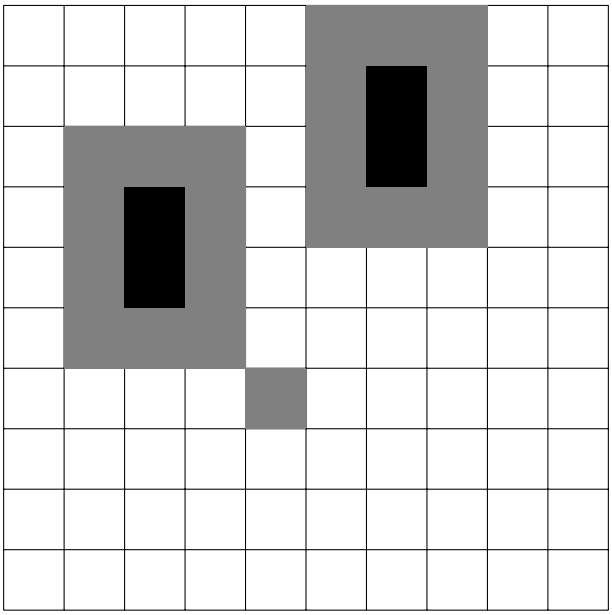

Figure 1.2.: solution de cas 1-les parcelles noires sont les zones centrales protégées et les parcelles gris sont zones tampons

Pour cas 2, on a la solution en Figure 1.3.

Pour cas 3, on a la solution en Figure 1.4.

Pour cas 4, on a la solution en Figure 1.5.

1.4 Performance

Voir tableau 1.1.

	Temps	Nbr Noeuds	Coût	probabilité de survie
Cas1	0.05s	0	119	$(0.92, 0.91, 0.92, 0.98, 0.89, 0.98)$
Cas2	0.02s	0	327	$(0.58, 0.52, 0.64, 0.92, 0.64, 0.76)$
Cas3	0.08s	0	130	$(0.58, 0.52, 0.64, 0.93, 0.91, 0.91)$
Cas4	0.06s	0	211	$(0.82, 0.81, 0.82, 0.97, 0.78, 0.88)$

Table 1.1: performance et solution de cas différents

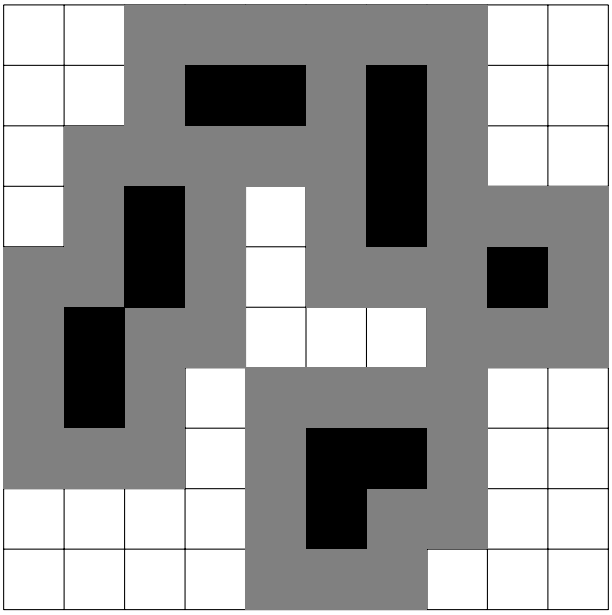

Figure 1.3.: solution de cas 2-les parcelles noires sont les zones centrales protégées et les parcelles gris sont zones tampons

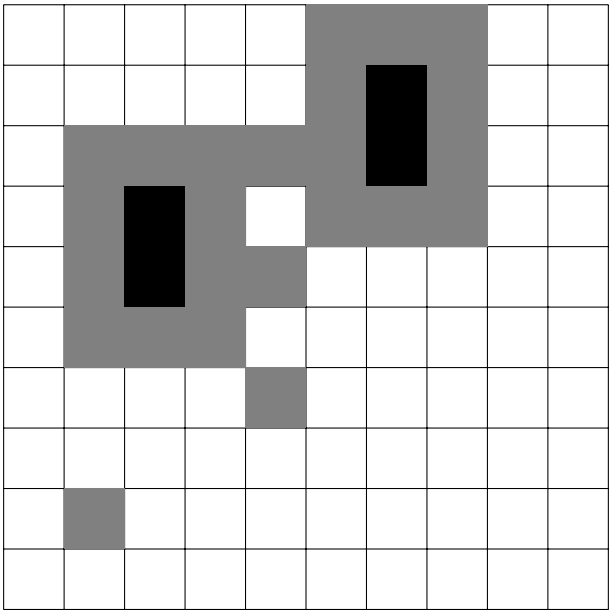

Figure 1.4.: solution de cas 3-les parcelles noires sont les zones centrales protégées et les parcelles gris sont zones tampons

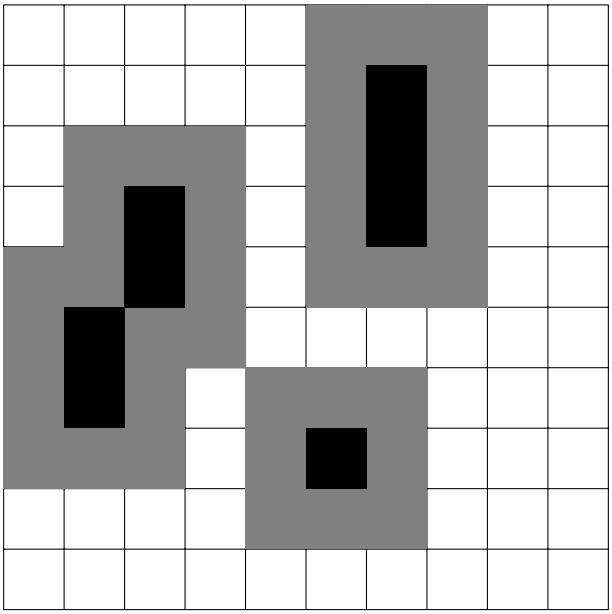

Figure 1.5.: solution de cas 4-les parcelles noires sont les zones centrales protégées et les parcelles gris sont zones tampons

1.5 Sensibilité de la taille d'instance

Afin d'étudier la sensibilité de la taille, nous générons des instances aléatoires avec les règles suivantes:

- le coût de chaque parcelle est choisi aléatoirement entre $[1, 2, \ldots, 10]$
- la probabilité qu'une espèce rare puisse survivre sur une parcelle est 0.33, et après sa probabilité de survie est choisi aléatoirement entre $[0.1, 0.2, 0.3, 0.4, 0.5]$
- la probabilité qu'une espèce rare puisse survivre sur une parcelle est 0.5, et après sa probabilité de survie est choisi aléatoirement entre $[0.2, 0.3, 0.4, 0.5, 0.6]$

On a étudié 10 différent taille de 10×10 à 100×100. Les résultats se trouvent en tableau 1.2.

Car les instances sont générées aléatoirement et la difficulté du problème se varie de différentes instances, on ne trouve pas un règle explicite à partir de ces résultats. Mais quand-même on peut voir que, en générale, la difficulté du problème augmente avec

Taille	Temps	Nbr Noeuds
10×10	0.06s	0
20×20	0.25s	0
30×30	0.64s	23
40×40	206.61s	28479
50×50	3.47s	39
60×60	1.17s	0
70×70	19.53s	87
80×80	29.31s	91
90×90	18.31s	34
100×100	119s	269

Table 1.2: performance et solution de cas différents

l'augmentation de la taille: plus des noeuds dans l'arbre de recherche sont développés et plus de temps de calcul sont pris.

1.6 Un autre modèle

Le modèle peut s'écrire comme:

$$(P3)\begin{cases} \max \quad P - \sum_{k\in \text{Rare}} \prod_{ij\in S}(1-p_{k,ij}x_{ij}) - \sum_{l\in \text{Commune}} \prod_{ij\in S}(1-p_{l,ij}x_{ij}) & \\ \text{s.t.} \quad 9x_{ij} \leq \sum_{i'j'\in Nbr(ij)} y_{i'j'} & ij \in S \\ \quad \sum_{ij\in S]} c_{ij}x_{ij} \leq B & \\ \quad x_{ij} \in \{0,1\} & ij \in S \\ \quad y_{ij} \in \{0,1\} & ij \in S \end{cases}$$

On ne peut pas le formuler par un programme linéaire en variables mixtes à cause de cette forme $\sum\prod(\cdot)$.

2. MAÎTRISE DES EFFETS DE LA FRAGMENTATION DU PAYSAGE

Dans ce projet, nous étudions le problème de la fragmentation des espaces. L'indicateur que l'on veut optimiser est *la distance moyenne au plus proche voisin (DMPPV)*, qui est

$$\text{DMPPV} = \frac{1}{n}\sum_{i=1}^{n} \min_{j}\{d_{ij} : j = 1, \ldots, n; j \neq i\}$$

. Une méthode de optimisation combinatoire fractionnaire est implémentée.

2.1 Pré-modélisation

Un problème d'optimisation combinatoire fractionnaire est un programme de la forme:

$$(P)\begin{cases} \max & \dfrac{f(x)}{g(x)} \\ \text{s.t.} & x \in X \subseteq [0,1]^n \end{cases}$$

Il peut être résolu par une problème paramétrique associée:

$$(P_\lambda)\begin{cases} \max & f(x) - \lambda g(x) \\ \text{s.t.} & x \in X \subseteq [0,1]^n \end{cases}$$

Un algorithme de *fixed-point* est propose à le résoudre:

Algorithme 1 : Algorithme de Dinkelbach

initial : $\lambda = \lambda_0, \nu = \nu_0$

tant que $\nu_0 > 0$ faire

  $\nu = \max(f(x) - \lambda g(x) : x \in X \subseteq [0,1]^n)$;

  soit x^* la solution trouvée ($\nu = f(x^*) - \lambda g(x^*)$) ;

  $\lambda = \frac{f(x^*)}{g(x^*)}$;

fin

retourner x^*

2.2 Modélisation

2.2.1 Description du problème

Voir le fichier donné.

2.2.2 Modélisation mathématiques

$$(P1)\begin{cases} \min & \dfrac{\sum_{i,j} d_{ij} y_{ij}}{\sum_{i=1}^{n} x_i} & \\ \text{s.t.} & y_{ij} \leq x_i & \forall i,j \\ & y_{ij} \leq x_j & \forall i,j \\ & \displaystyle\sum_{j=1}^{n} y_{ij} == x_i & \forall i \\ & A_{\min} \leq \displaystyle\sum_{i=1}^{n} x_i \leq A_{\max} & \\ & \displaystyle\sum_{i=1}^{n} c_i x_i \leq B & \\ & x \in \{0,1\}^n & \\ & y \in \{0,1\}^{n \times n} & \end{cases}$$

où x_i désigne si une parcelle est choisie et y_{ij} désigne si la parcelle s_j est la plus proche voisine de s_i.

2.2.3 Modélisation par un modèle d'optimisation combinatoire fractionnaire

Supposons que $f(x) = \sum_{i,j} d_{ij} y_{ij}$ et $g(x) = \sum_{i=1}^{n} x_i$, on a la problème paramétrique associé sous cette forme:

$$(P1_\lambda) \begin{cases} \min & f(x) - \lambda g(x) & \\ \text{s.t.} & y_{ij} \leq x_i & \forall i, j \\ & y_{ij} \leq x_j & \forall i, j \\ & \sum_{j=1}^{n} y_{ij} == x_i & \forall i \\ & A_{\min} \leq \sum_{i=1}^{n} x_i \leq A_{\max} & \\ & \sum_{i=1}^{n} c_i x_i \leq B & \\ & x \in \{0,1\}^n & \\ & y \in \{0,1\}^{n \times n} & \end{cases}$$

Prenant $\lambda_0 = 20$ et $\nu_0 = 10$, on peut résoudre le problème $P1$ itérativement par l'algorithme de Dinkelbach.

2.3 Solution

On a étudié une instance de la taille 10×10 (Voir Figure 2.1).

3 cas sont considérés:

- cas 1: $A_{\min} = 30$, $A_{\min} = 35$, $B = 920$
- cas 2: $A_{\min} = 20$, $A_{\min} = 21$, $B = 520$
- cas 3: $A_{\min} = 70$, $A_{\min} = 75$, $B = 3500$

Pour cas 1, on a la solution en Figure 2.2.

Pour cas 2, on a la solution en Figure 2.3.

Pour cas 3, on a la solution en Figure 2.4.

7	3	10	10	2	8	6	4	5	5
7	7	10	5	2	8	6	3	9	9
7	3	4	6	3	2	4	9	7	8
6	2	7	6	4	7	5	10	7	8
2	4	3	4	9	6	4	9	8	4
7	5	2	9	8	9	5	6	10	10
5	2	3	7	9	9	4	9	6	3
5	2	9	4	2	8	6	9	3	4
9	6	5	4	5	6	8	9	6	6
8	8	7	7	3	5	8	3	9	9

Figure 2.1.: Coût de sélection de chaque parcelle avec unité de 10

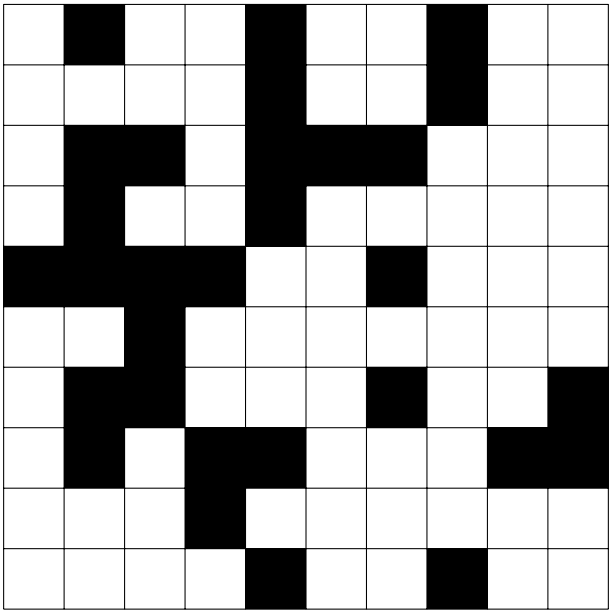

Figure 2.2.: solution de cas 1-les parcelles noires sont les parcelles sélectionnées: DMPPV $=$ 1.155009, Nombre de parcelles sélectionnées est 30

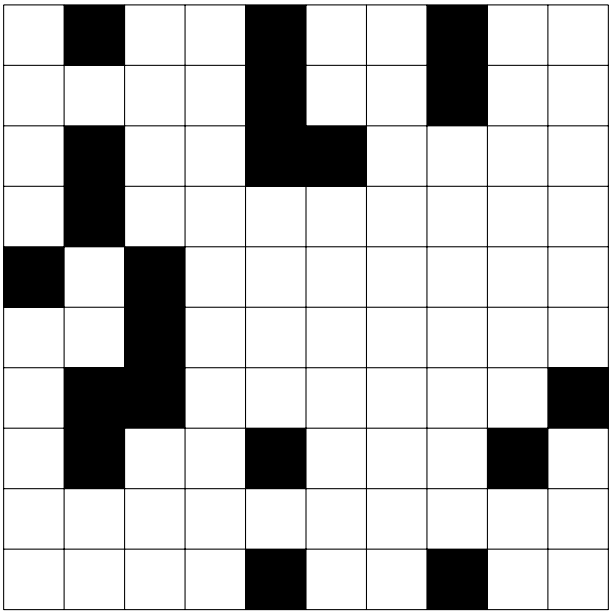

Figure 2.3.: solution de cas 2-les parcelles noires sont les parcelles sélectionnées: DMPPV $=$ 1.2739354, Nombre de parcelles sélectionnées est 20

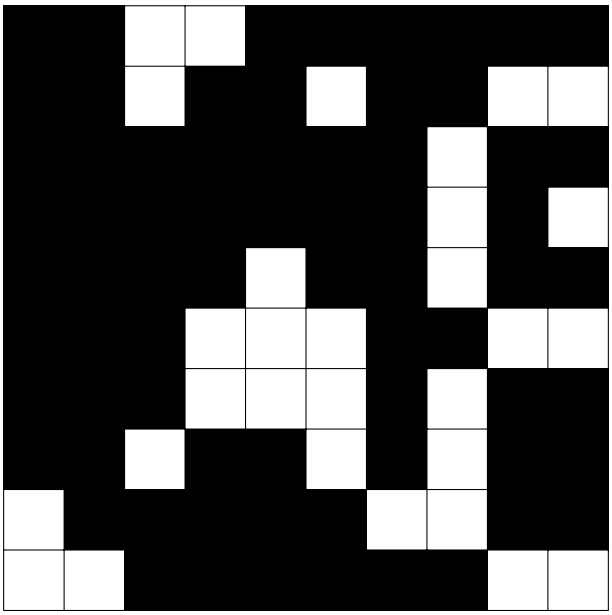

Figure 2.4.: solution de cas 3-les parcelles noires sont les parcelles sélectionnées: DMPPV $= 1$, Nombre de parcelles sélectionnées est 71

	Temps	Nbr Noeuds	Nbr Itérations	DMPPV
Cas1	0,36s	0	2	1.155009385
Cas2	0,31s	0	2	1.273935433
Cas3	0.36s	0	2	1

Table 2.1: performance et solution de cas différents

2.4 Performance

Voir tableau 2.1.

2.5 Sensibilité de la taille d'instance

Afin d'étudier la sensibilité de la taille, nous générons des instances aléatoires avec les règles suivantes:

- le coût de la sélection de chaque parcelle est choisi aléatoirement entre $[1, 2, \ldots, 10]$ (avec une unité de 10)
- $A_{\min}$ est choisi aléatoirement entre $[0.2 * M \times N, \ldots, 0.4 * M \times N]$ et $A_{\max} - A_{\min}$ est choisi aléatoirement entre $[1, \ldots, 5]$.
- B est fixé à $11 * M \times N$

On a étudié 7 différent taille de 5×5 à 35×35. Les résultats se trouvent en tableau 2.2.

On trouve que le temps de calcul augmente exponentiellement avec la taille d'instance. Les autres caractéristiques, comme le nombre de noeuds développés, nombre d'itération de l'algorithme Dinkelbach et le DMPPV, ne sont pas changé beaucoup par rapport à la taille, mais ils sont tous dépendant de l'instance, par exemple, la valeur B que l'on a choisi.

Taille	Temps	Nbr Noeuds	Nbr Itérations	DMPPV
5×5	0.03s	0	2	1
10×10	0.42s	0	2	1
15×15	4.91s	0	2	1
20×20	24.42s	0	2	1
25×25	87.64s	0	2	1.001670216
30×30	184.48s	0	2	1
35×35	505.42s	0	2	1

Table 2.2: L'effet de la taille sur le performance

3. PROTECTION DE LA DIVERSITÉ GÉNÉTIQUE

Dans ce projet, nous étudions le problème de la conservation génétique. L'objective est de minimiser la perte d'allèles dans la population engendrée en déterminant une contribution optimale de chaque individu. Une méthode d'approximation de la fonction logarithmique est implémentée pour faciliter la solution et donner une borne inférieure.

3.1 Pré-modélisation

Si une contrainte est de la forme:

$$\log(y) \geq f(x_1, \ldots, x_n)$$

où $0 < y \leq 1$ $x \in \mathbb{R}^n$.

Elle peut être approchée par un ensemble de contrainte de la forme

$$\log(\theta_r) + \frac{1}{\theta_r}(y - \theta_r) \geq f(x_1, \ldots, x_n)$$

où $r \in \{1, 2, \ldots, h\}$ et θ un vecteur donné de $\mathbb{R}^h$ tel que $0 < \theta_1 < \theta_2 < \ldots < \theta_h = 1$

3.2 Modélisation

3.2.1 Description du problème

Voir le fichier donné.

3.2.2 Modélisation mathématiques

$$(P)\begin{cases} \min & \sum_{g\in G}\sum_{a\in A} y_{ga} & \\ \text{s.t.} & z_{ga} \geq \prod_{i\in N, p_{iga}\neq 0} p_{iga}^{x_i} & \forall g\in G, a\in A \\ & y_{ga} \geq z_{ga} - \sum_{i\in N, p_{iga}=0} x_i & \forall g\in G, a\in A \\ & \sum_{i\in N_m} x_i = N & \\ & \sum_{i\in N_m} x_i = \sum_{j\in N_f} x_j & \\ & 0\leq x_i\leq 3 & \forall i\in N \\ & y_{ga}\in[0,1] & \forall g\in G, a\in A \\ & z_{ga}\in[0,1] & \forall g\in G, a\in A \end{cases}$$

où x_i désigne le nombre d'enfant d'individu i, y_{ga} désigne la probabilité de disparition d'allèle a de locus g, $p_{i,ga}$ la probabilité de disparition d'allèle a de locus g sur l'individu i, et z_{ga} est une variable intermédiaire introduite pour éviter l'indétermination de la forme 0^0. Quand un individu, dont la probabilité de disparition de ga est 0, a des enfants de nombre non nul, on a $z_{ga} - \sum_{i\in N, p_{iga}=0} x_i$ négatif, qui signifie la deuxième contrainte est toujours satisfit par y_{ga}. Dans la solution optimale, on a $y_{ga} = 0$. c'est d'à dire, tant qu'un tel individu a un enfant, l'allèle ga sera jamais perdu dans le reproduction.

On peut re-écrire la première contrainte sous la forme de logarithmique:

$$(P1)\begin{cases} \min & \sum_{g\in G}\sum_{a\in A} y_{ga} & \\ \text{s.t.} & \log(z_{ga}) \geq \sum_{i\in N, p_{iga}\neq 0} \log(p_{i,ga}) * x_i & \forall g \in G, a \in A \\ & y_{ga} \geq z_{ga} - \sum_{i\in N, p_{iga}=0} x_i & \forall g \in G, a \in A \\ & \sum_{i\in N_m} x_i = N & \\ & \sum_{i\in N_m} x_i = \sum_{j\in N_f} x_j & \\ & 0 \leq x_i \leq 3 & \forall i \in N \\ & y_{ga} \in [0,1] & \forall g \in G, a \in A \\ & z_{ga} \in [0,1] & \forall g \in G, a \in A \end{cases}$$

3.2.3 Approximation par une fonction linéaire par morceaux

Il suffit de remplacer la contrainte

$$\log(z_{ga}) \geq \sum_{i\in N, p_{iga}\neq 0} \log(p_{iga}) * x_i \quad \forall g \in G, a \in A$$

par une suite des contraintes:

$$\log(\theta_r) + \frac{1}{\theta_r}(z_{ga} - \theta_r) \geq \sum_{i\in N, p_{iga}\neq 0} \log(p_{iga}) * x_i \quad \forall g \in G, a \in A$$

où $\theta_r = \theta_1^{\frac{h-r}{h-1}}$ avec $\theta_1 = 0.001$ et $h = 50$.

3.3 Solution

On a étudié une instance de la taille

- 8 individu, dont 4 mâles et 4 femelles
- 1 paire de chromosomes

	x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8
cas 1	1	3	3	1	3	0	3	2
cas 2	2	2	2	2	2	2	2	2

Table 3.1: Solutions des cas différents

- 5 locus par chromosome
- 2 allèles par gène

Détail de cette instance peut se trouver dans la fichier donnée ou sur le site du cours [1].

2 cas sont considérés:

- cas 1: $x_i \leq 3 \quad \forall i$
- cas 2: $x_i \leq 2 \quad \forall i$

Notez que cas 2 en fait a une solution triviale. Car nous ajoutons la contrainte de garder le nombre de la population, le nombre moyen d'enfant de chaque individu est au moins 2. Si on ajoute une contrainte que le nombre d'enfant de chaque individu est au maximum 2, ceci nécessite le fait que tout individu a exactement 2 enfants.

On a la solution pour ces deux cas sur le tableau 3.1 [2].

3.4 Performance

Voir tableau 3.2.

[1] `http://cedric.cnam.fr/~lamberta/MPRO/RODD/projet3/`

[2] Notez que la solution de cas 1 est un peu différent que la solution proposé par la professeure. Dans notre solution, il y a que b qui a une probabilité de disparition. e n'a plus de risque de disparaître car individu 4 a un enfant. Les valeurs objectives sont de même donc on ne connait pas encore quelle solution est correcte et la raison pour la différence.

	Temps	Nbr Noeuds	Proba Disparition	E(Nbr allèle disparus)	borne inférieure
Cas1	0.02s	0	0.015625 (b)	0.015625	0.0155881
Cas2	0.00s	0	0.0625 (b)	0.0625	0.0624334

Table 3.2: performance et solution de cas différents

Nbr Morceaux	Temps	Nbr Noeuds	Proba Disparition	E(Nbr allèle disparus)	borne inférieure
10	0.02s	0	0.015625	0.015625	0.0146234
20	0.02s	0	0.015625	0.015625	0.0154033
30	0.02s	0	0.015625	0.015625	0.0155242
40	0.02s	0	0.015625	0.015625	0.0155651
50	0.02s	0	0.015625	0.015625	0.0155881
60	0.02s	0	0.015625	0.015625	0.0156014
70	0.02s	0	0.015625	0.015625	0.0156091
80	0.02s	0	0.015625	0.015625	0.0156139
90	0.03s	0	0.015625	0.015625	0.0156170
100	0.03s	0	0.015625	0.015625	0.0156191
200	0.03s	0	0.015625	0.015625	0.0156247
300	0.05s	0	0.015625	0.015625	0.0156250

Table 3.3: L'effet du nombre de morceaux sur le performance

3.5 Sensibilité de la taille d'instance et du nombre de morceaux de la fonction de l'approximation

Afin d'étudier la sensibilité du nombre de morceaux de la fonction de l'approximation, nous fixons l'instance de 8 individu et 5 locus (donnée), en changeant le nombre de morceaux T et la valeur initiale θ_0. Les résultats se trouvent dans tableau 3.3 et tableau 3.4. Quand le nombre de morceaux augmente, le gap entre la borne inférieure et l'espérance du nombre allèle disparus diminue et on a une meilleur approximation. Par contre, la valeur initiale ne semble pas avoir trop d'impact sur les résultats.

Afin d'étudier la sensibilité de la taille, nous générons des instances aléatoires avec les règles suivantes:

- l'allèle est choisi aléatoirement entre $\{1, 2\}$
- Nombre d'allèle, nombre de paires de chromosomes considérées sont fixés
- On change la taille de population et la taille du locus

Valeur Initiale	Temps	Nbr Noeuds	Proba Disparition	E(Nbr allèle disparus)	borne inférieure
0.00001	0.02s	0	0.015625	0.015625	0.0155881
0.0001	0.02s	0	0.015625	0.015625	0.0156206
0.001	0.02s	0	0.015625	0.015625	0.0155881
0.01	0.02s	0	0.015625	0.015625	0.0156206
0.1	0.02s	0	0.03125	0.03125	0.0
0.5	0.00s	0	0.125	0.125	0.0

Table 3.4: L'effet de la valeur initiale sur le performance

Nbr Individu	Temps	Nbr Noeuds	E(Nbr allèle disparus)	borne inférieure
2	0.00s	0	0.75	0.7496
4	0.00s	0	0.140625	0.140469
6	0.05s	0	0.000732422	0.000283136
8	0.05s	0	0.000244141	0
10	0.01s	0	0.00012226	0
12	0.00s	0	3.8147e−6	0
14	0.00s	0	0	0

Table 3.5: L'effet du nombre d'individu sur le performance

Nbr Locus	Temps	Nbr Noeuds	E(Nbr allèle disparus)	borne inférieure
12	0.02s	0	0.00390625	0.00390177
10	0.02s	0	0.000610352	0.000283136
8	0.03s	0	0.000274658	0
6	0.00s	0	0.000976563	0.000566273
5	0.00s	0	0.000244141	0
4	0.02s	0	0.00012207	0
3	0.00s	0	0.015625	0.0155881

Table 3.6: L'effet du nombre de locus par chaque chromosome sur le performance

Les résultats se trouvent dans tableau 3.5 et tableau 3.6. Quand la taille de la population augmente, le risque de la perte génétique diminue. On peut aussi voir que quand le nombre de locus augmente, le gap entre la borne inférieure et l'espérence du nombre allèle disparus augmente.

4. EXPLOITATION DURABLE DE LA FORÊT

Dans ce projet, nous étudions le problème de la gestion de durable des forêts. L'objective est de faire la exploitation de la forêt visant à protéger le mieux possible certaines espèces.

4.1 Pré-modélisation

Le problème peut se forme par un programme linéaire en variables 0-1 suivant:

$$(P1)\begin{cases} \max & w_1 \sum\limits_{(i,j)\in M\times N} t_{ij}(1-x_{ij}) + w_2 gl \sum\limits_{(i,j)\in M\times N} 4x_{ij} - d_{ij} \\ \text{s.t.} & d_{ij} \geq \sum\limits_{(k,l)\in A_{ij}} x_{kl} - |A_{ij}|(1-x_{ij}) \qquad (i,j)\in M\times N \\ & d \in \mathbb{R}_+^{M\times N} \\ & x \in \{0,1\}^{M\times N} \end{cases}$$

où w_1 et w_2 sont les coefficients de pondération. l est la longueur du côté de chaque parcelle et A_{ij} désigne l'ensemble des couples (k,l) tels que la parcelle s_{kl} est adjacente à la parcelle s_{ij}. x_{ij} désigne si la parcelle est coupée.

Donc d_{ij} désigne le nombre de parcelles coupées autour d'une parcelles coupée. Quand s_{ij} est une parcelle coupée, $1-x_{ij}$ est 0, $d_{ij} = \sum_{(k,l)\in A_{ij}} x_{kl}$ est le nombre de parcelles coupées dans ses voisines. Quand s_{ij} est une parcelle non coupée, $1-x_{ij}$ est 1 et on a $\sum_{(k,l)\in A_{ij}} x_{kl} - |A_{ij}|(1-x_{ij})$ toujours négative, donc $d_{ij} = 0$. Ceci signifie que $4x_{ij} - d_{ij}$ est des lisères entre une parcelle coupée et ses voisines non coupées.

4.2 Modélisation quadratique

On peut aussi formuler le problème par un programme quadratique suivant:

$$(P2)\begin{cases}\max & w_1 \sum\limits_{(i,j)\in M\times N} t_{ij}(1-x_{ij}) + w_2 gl \sum\limits_{(i,j)\in M\times N}\ \sum\limits_{(k,l)\in A_{ij}} x_{ij}(1-x_{kl}) \\ \text{s.t.} & x \in \{0,1\}^{M\times N}\end{cases}$$

4.3 Linéarisions pour une matrice TU

$$(P2L)\begin{cases}\max & w_1 \sum\limits_{(i,j)\in M\times N} t_{ij}(1-x_{ij}) + w_2 gl \sum\limits_{(i,j)\in M\times N}\ \sum\limits_{(k,l)\in A_{ij}} x_{ij} - y_{ijkl} \\ \text{s.t.} & y_{ijkl} \geq x_{ij} + x_{kl} - 1 \quad \forall (i,j) \in M\times N, (k,l) \in A_{ij} \\ & y_{ijkl} \leq x_{ij} \quad (i,j) \in M\times N \\ & y_{ijkl} \leq x_{kl} \quad (k,l) \in A_{ij} \\ & y_{ijkl} \geq 0 \\ & x \in \{0,1\}^{M\times N} \\ & y \in \{0,1\}^{M\times N\times M\times N}\end{cases}$$

Voir que nous maximisons l'objectif, que le coefficient de y est négatif et que $x_{ij}, y_{ijkl} \in \{0,1\}$, les contraints $y_{ijkl} \leq x_{ij}$ et $y_{ijkl} \leq x_{kl}$ sont redondants. Donc la matrice de contrainte est:

$$M = \begin{array}{c} \begin{matrix} x_{11} & x_{12} & \dots & x_{mn} & \dots & y_{ijkl} & \dots \end{matrix} \\ \begin{bmatrix} 1 & \dots & 1 & 0 & -1 & 0 & \dots \\ 0 & 1 & 1 & 0 & 0 & -1 & \dots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \ddots \end{bmatrix} \end{array}$$

M est la juxtaposition de $[M^X, M^Y]$ où M^Y est une matrice négative identité et M^X une matrice d'incidence d'un graphe bipartie (non orienté) $[X, X]$ tel que une arrête existe si et seulement si s_{ij} et s_{kl} adjacente.

Proposition 4.3.1 *La matrice d'incidence d'un graphe bipartie est une matrice totalement unimodulaire (TU).*

Proof Vue au cours de l'optimisation dans les Graphes. ■

Proposition 4.3.2 *La juxtaposition d'une matrice totalement uni-modulaire et une matrice (négative) identité est une matrice totalement uni-modulaire (TU).*

Proof Vue au cours de l'optimisation dans les Graphes. ■

Lemma 1 *La matrice de contrainte M du programme (P2L) est une matrice TU.*

Proof La preuve est directe après les deux propositions ci-dessus. ■

Car la matrice de contrainte est TU et le second membre est entier, nous savons que la solution de la relaxation continue de $(P2L)$ est toujours entière. Donc résoudre $(P2L)$ est égale à résoudre sa relaxation continue, qui est:

$$(P2L') \begin{cases} \max & w_1 \sum\limits_{(i,j)\in M\times N} t_{ij}(1-x_{ij}) + w_2 gl \sum\limits_{(i,j)\in M\times N} \sum\limits_{(k,l)\in A_{ij}} x_{ij} - y_{ijkl} \\ \text{s.t.} & y_{ijkl} \geq x_{ij} + x_{kl} - 1 \quad \forall (i,j) \in M\times N, (k,l) \in A_{ij} \\ & x_{ij} \leq 1 \\ & y_{ijkl} \in \mathbb{R}_+ \\ & x_{ij} \in \mathbb{R}_+ \end{cases}$$

4.4 Solution

On a étudié une instance de la taille 10×10 (Figure 4.1) et une autre de 5×5 (Figure 4.2), et on les a résolu par les deux méthodes de $(P1)$, $(P2L')$. Ils retournent même résultat qui se trouve dans Figure 4.3 et 4.4.

4.5 Performance

Voir tableau 4.1.

Comme le $(P2L')$ est un programme linéaire des variables continues, il n'y a pas des noeuds développés, son temps de calcul est toujours efficace car c'est un programme simple à résoudre. Le $(P1)$ est un peu moins efficace que le $(P2L')$ car c'est un programme

		Temps	Nbr Noeuds
10×10	$(P1)$	$0,01$s	0
	$(P2L')$	$0,00$s	$---$
5×5	$(P1)$	$0,01$s	0
	$(P2L')$	$0,00$s	$---$

Table 4.1: performance et solution de deux méthodes sur deux instances

84	68	97	98	64	89	82	71	74	76
87	83	98	75	60	90	78	67	92	94
84	68	70	81	67	61	73	92	86	90
79	62	86	79	73	84	76	98	84	90
62	72	66	72	92	80	71	91	87	70
85	77	63	93	90	94	76	81	99	98
76	63	66	84	94	93	72	92	79	65
76	63	92	69	60	88	79	93	66	73
92	82	77	72	77	81	89	95	80	80
88	89	83	86	69	78	91	64	94	92

Figure 4.1.: instance de taille 10×10 avec $w_1 = 1, w_2 = 5, l = 3, g = 1.26157$

10	10	10	1	10
10	10	1	1	10
10	10	1	10	10
1	10	10	10	10
1	10	10	10	10

Figure 4.2.: instance de taille 5×5 avec $w_1 = 2, w_2 = 1, l = 3, g = 1.26157$

linéaire des variables entières. Mais sur ce taille d'instance donnée (et les deux instances données), il est assez performant par branch&cut.

4.6 Contraint ajouté

Ça suffit d'ajouter un contraint $\sum_{(i,j)\in M\times N} x_{ij} \geq 60$ dans $P1$ et $P2L$. Notez que maintenant la matrice des contraintes de $P2L$ n'est plus TU, donc on ne peut plus résoudre

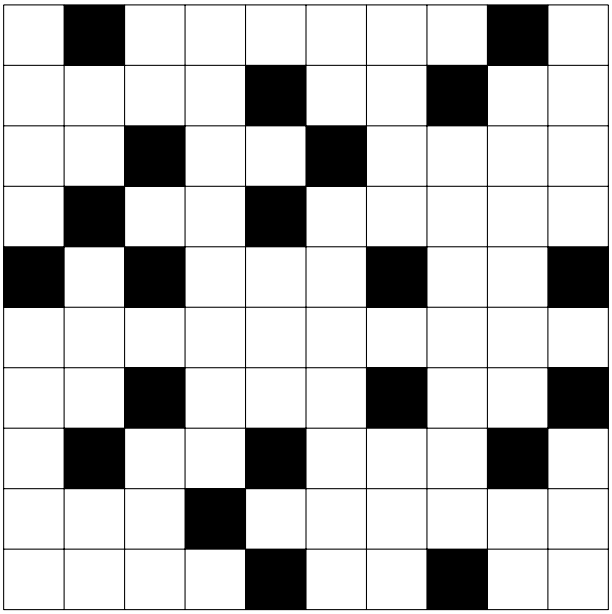

Figure 4.3.: solution de instance 10×10-les parcelles noires sont les non coupées avec Valeur objectif 8219.58, Nombre de parcelles non coupées 21 et Effectif de l'espèce e_1 6630

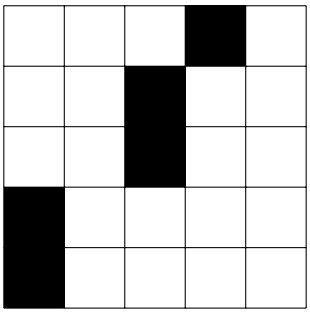

Figure 4.4.: solution de instance 5×5-les parcelles noires sont les non coupées avec Valeur objectif 442.56, Nombre de parcelles non coupées 5 et Effectif de l'espèce e_1 382

$P2L$ par résoudre sa relaxation continue. Le résultat est suivant dans le figure 4.5 et le tableau 4.3:

		Temps	Nbr Noeuds
10×10	$(P1)$	$0,12$s	0
	$(P2L)$	$0,02$s	0

Table 4.2: performance et solution de deux méthodes avec nouveau contraint

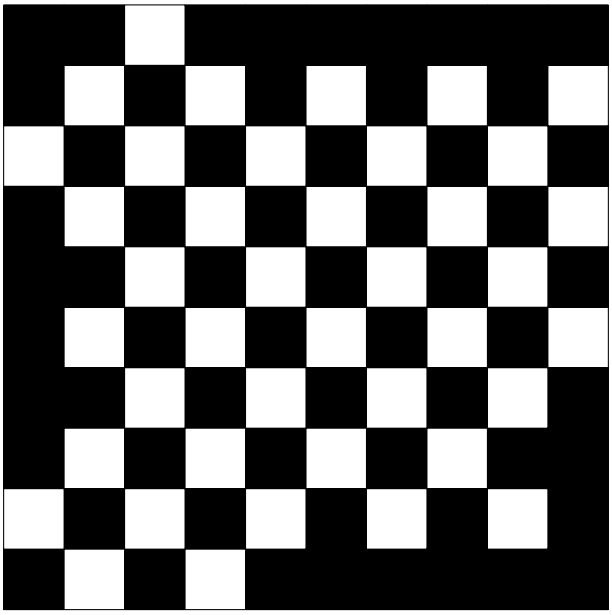

Figure 4.5.: solution de instance 10×10 avec contraint; Valeur objectif 6753.93, Nombre de parcelles non coupées 60 et Effectif de l'espèce e_1 3272

4.7 Sensibilité de la taille d'instance

Afin d'étudier la sensibilité de la taille, nous générons des instances aléatoires avec les règles suivantes:

- t_{ij}, la population attendue de l'espèce e_1 dans chaque parcelle s_{ij} si coupée est choisi aléatoirement entre $[50, 51, \ldots, 100]$
- w_1, w_2, g, L sont fixés

On voit dans le tableau 4.3 que le modèle $P1$ est beaucoup plus robuste par rapport à la relaxation linéaire d'un modèle quadratique, soit $P2L$ ou $P2L'$, quand la taille d'instance augmente. De 20×20 à 120×120, le temps de calcul pour $P2L$ et $P2L'$ augmente rapidement tel que ils ne peuvent plus retourner une solution dans un temps donné quand la taille est 120×120. Le temps de calcul pour $P1$ aussi augmente mais plus lentement (presque linéaire à la taille m). La raison est que les nombres des variables et des contraintes de $P2L$ et $P2L'$ augmente plus vite que celles de $P1$.

		Temps	Nbr Noeuds
20×20	$(P1)$	$0,08$s	0
	$(P2L')$	$0,04$s	$---$
	$(P1)$ avec contraint	$0,08$s	0
	$(P2L)$ avec contraint	$0,09$s	0
50×50	$(P1)$	$0,31$s	0
	$(P2L')$	$1,31$s	$---$
	$(P1)$ avec contraint	$0,23$s	0
	$(P2L)$ avec contraint	$2,84$s	0
80×80	$(P1)$	$0,34$s	0
	$(P2L')$	$8,67$s	$---$
	$(P1)$ avec contraint	$0,62$s	0
	$(P2L)$ avec contraint	$20,68$s	0
100×100	$(P1)$	$0,36$s	0
	$(P2L')$	$17,99$s	$---$
	$(P1)$ avec contraint	$1,24$s	0
	$(P2L)$ avec contraint	$48,58$s	0
120×120	$(P1)$	$0,52$s	0
	$(P2L')$	$---$	$---$
	$(P1)$ avec contraint	$1,12$s	0
	$(P2L)$ avec contraint	$---$	$---$

Table 4.3: performance et solution de deux méthodes avec nouveau contraint