

UNIVERSITÉ LIBRE DE BRUXELLES
FACULTÉ DES SCIENCES
DÉPARTEMENT D'INFORMATIQUE

INFO-F-302 - Logique Informatique
Projet : Le jeu Pattern et Utilisation de
MiniSAT

OOMS Aurélien, SONNET Jean-Baptiste

Table des matières

1	Énumération 3,2	2
1.1	Problème et notation	2
1.1.1	Grille	2
1.1.2	Variables	3
1.1.3	Sémantique	3
1.1.4	Codage	3
1.2	Contraintes	3
1.3	Parcours et représentation par énumération	4
2	Les Bandes	7
2.1	Problème et notation	7
2.1.1	Variables	7
2.1.2	Sémantique	7
2.1.3	Codage	8
2.2	Contraintes	8
2.2.1	Une bande	9
2.3	Représentation par bandes	11
2.3.1	Observation : position minimale, maximale et sous-espace	11
2.3.2	Énumération récursive	12

Chapitre 1

Énumération 3,2

1.1 Problème et notation

1.1.1 Grille

Le problème est présenté sous forme d'une grille 3×3 contenant au *max* 2 contraintes par ligne ou colonne.

Soit une matrice 3×3 ,

$$\begin{bmatrix} x_{0,0} & x_{0,1} & x_{0,2} \\ x_{1,0} & x_{1,1} & x_{1,2} \\ x_{2,0} & x_{2,1} & x_{2,2} \end{bmatrix} \rightarrow \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix}$$

où chacune des cases $x_{i,j}$ prendra potentiellement une des 2 valeurs : $\{-1, 1\}$, respectivement le blanc, le noir. Les cases sont initialisées avec la valeur 0.

On aura par exemple comme problème à résoudre :

$$\begin{array}{ccc} & 2 & 1 & 2 \\ \begin{array}{c} 1 \\ 1 \, 1 \\ 2 \end{array} & \begin{bmatrix} 1 & -1 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} & \begin{array}{|c|c|c|} \hline \blacksquare & \square & ? \\ \hline ? & ? & ? \\ \hline ? & ? & ? \\ \hline \end{array} \end{array}$$

Selon les contraintes précisées, la solution devra donner pour toutes les cases inconnues une valeur de 1 ou -1 :

$$\begin{array}{ccc} & 2 & 1 & 2 \\ \begin{array}{c} 1 \\ 1 \, 1 \\ 2 \end{array} & \begin{bmatrix} 1 & -1 & -1 \\ 1 & -1 & 1 \\ -1 & 1 & 1 \end{bmatrix} & \begin{array}{|c|c|c|} \hline \blacksquare & \square & \square \\ \hline \blacksquare & \square & \blacksquare \\ \hline \square & \blacksquare & \blacksquare \\ \hline \end{array} \end{array}$$

1.1.2 Variables

Chaque case est représentée par une variable x , de sorte que la case à la i^{eme} ligne et à la j^{eme} colonne se note $x_{i,j}$.

On a que $(i, j) \in \{1, 2, 3\}^2$ et que chaque variable peut prendre comme valeur $v \in \{-1, 0, 1\}$ soit :

$$X = \{x_{i,j} | (i, j) \in \{0, 1, 2\}^2, v \in \{-1, 0, 1\}\}$$

1.1.3 Sémantique

On a pour sémantique :

- $x_{i,j}$ est **vrai** si et seulement si la case (i, j) a pour valeur $v = 1$
- $x_{i,j}$ est **faux** si et seulement si la case (i, j) a pour valeur $v = -1$

1.1.4 Codage

Pour miniSAT, il faut coder les propositions par des entiers.

La taille d'une grille est de 3×3 , on codera en base 3.

Pour (i, j, v) on aura le codage : $(i \times n^2) + (j \times n) + (v + 1)$

1.2 Contraintes

1. (trivial) **Pour chaque** ligne, s'il existe une bande, **toutes** les cases ne sont pas fausses.

$$\bigwedge_{i \in \{0,1,2\}} \neg \left(\bigwedge_{j \in \{0,1,2\}} \neg x_{i,j} \right)$$

soit,

$$\bigwedge_{i \in \{0,1,2\}} \left(\bigvee_{j \in \{0,1,2\}} x_{i,j} \right)$$

idem pour chaque colonne.

$$\bigwedge_{j \in \{0,1,2\}} \left(\bigvee_{i \in \{0,1,2\}} x_{i,j} \right)$$

2. (trivial) **Pour toutes** lignes, **pour toutes** colonnes, **pour toutes** bandes, **il existe** une case qui satisfait à la fois une bande de la ligne et une de la colonne.

$$\bigwedge_{i \in \{0,1,2\}} \bigwedge_{j \in \{0,1,2\}} \bigwedge_{b_i^{(k)}, k \in \{1,2\}} \neg (\neg x_{i,j} \wedge x_{i,j})$$

soit,

$$\bigwedge_{i \in \{0,1,2\}} \bigwedge_{j \in \{0,1,2\}} \bigwedge_{b_i^{(k)}, k \in \{1,2\}} (x_{i,j} \vee \neg x_{i,j})$$

3. **Pour toutes** lignes i , **pour toutes** bandes $b_i^{(k)}, k \in \{1, 2\}$ associées à la ligne, **il existe** *autant* de cases vraies $x_{i,j}$ que ne l'exprime la taille de la bande.

Cela donne au cas par cas :

- (a) Une seule bande :

- i. Pour toutes lignes i où $b = 3$,

$$\bigwedge_{j \in \{0,1,2\}} \left(\bigvee_{l=1}^{n=1} x_{i,j} \right)$$

⊤	⊤	⊤
---	---	---

- ii. Pour toutes lignes i où $b = 2$,

$$(\neg x_{i,0} \vee \neg x_{i,2}) \wedge (x_{i,0} \vee x_{i,2}) \wedge x_{i,1}$$

⊤	⊤	⊥
---	---	---

 ou

⊥	⊤	⊤
---	---	---

- iii. Pour toutes lignes i où $b = 1$,

$$\bigwedge_{\substack{j,j' \in \{0,1,2\} \\ j < j', j'' \neq j, j'}} (\neg x_{i,j} \vee \neg x_{i,j'} \vee x_{i,j''})$$

⊤	⊥	⊥
---	---	---

 ou

⊥	⊤	⊥
---	---	---

 ou

⊥	⊥	⊤
---	---	---

- iv. Pour toutes lignes i où $b = 0$,

$$\bigwedge_{j \in \{0,1,2\}} \left(\bigvee_{l=1}^{n=1} \neg x_{i,j} \right)$$

⊥	⊥	⊥
---	---	---

- (b) Deux bandes, pour toutes lignes i :

$$\left(\bigvee_{l=1}^{n=1} x_{i,0} \right) \wedge \left(\bigvee_{l=1}^{n=1} \neg x_{i,1} \right) \wedge \left(\bigvee_{l=1}^{n=1} x_{i,2} \right)$$

⊤	⊥	⊤
---	---	---

Idem pour toutes colonnes.

1.3 Parcours et représentation par énumération

On va parcourir la grille et énumérer les combinaisons possibles premièrement suivant les contraintes de lignes ensuite selon les contraintes de colonnes.

Par ligne (respectivement colonne), on crée si nécessaire une clause représentant les cas possibles quant aux cases dont on ne connaît pas encore la nature.

Pour chaque case contenue dans une ligne i donnée :

1. Si elle contient une information (1 ou -1), on en créer une clause à part entière qui sera jointe (\wedge) à toutes les autres.
2. Si elle ne contient pas d'informations (0) et conditionnellement aux contraintes $c_i^{(k)}$, $k \in \{1, 2\}$, on énumère les possibilités sous forme disjonctive.

On effectue de même par colonne et relativement aux contraintes de la colonne.

Algorithm 1 Énumération selon la valeur de(s) bande(s) de chaque lignes

```
for ligne  $i \rightarrow n$  do
  for colonne  $j \rightarrow 2$  do                                     ▷ encodage de l'information connue
    if  $x_{i,j} == \perp$  then
      Créer une nouvelle clause avec  $\neg x_{i,j}$ 
    else if  $x_{i,j} == \top$  then
      Créer une nouvelle clause avec  $x_{i,j}$ 
    end if
  end for
  for colonne  $j \rightarrow 2$  do                                     ▷ application de la contrainte 1
    if  $\nexists clause_i$  then
      Créer  $clause_i$ 
    end if
    Ajouter  $x_{i,j}$  à  $clause_i$ 
  end for
  if nombre de bandes == 1 then
    if  $bande_i == 0$  then                                     ▷ si la taille de la bande est 0
      for colonne  $j \rightarrow 2$  do
        Créer une nouvelle clause avec  $\neg x_{i,j}$ 
      end for
    else if  $bande_i == 1$  then                                 ▷ si la taille de la bande est 1
       $J = \{0, 1, 2\}$ 
      for colonne  $j'' \rightarrow 2$  do
         $j = \min(J \setminus j'')$ 
         $j' = \max(J \setminus j'')$ 
        Créer une nouvelle clause avec  $(\neg x_{i,j} \vee \neg x_{i,j'} \vee x_{i,j''})$ 
      end for
    else if  $bande_i == 2$  then                                 ▷ si la taille de la bande est 2
      Joindre la FNC  $(\neg x_{i,0} \vee \neg x_{i,2}) \wedge (x_{i,0} \vee x_{i,2}) \wedge x_{i,1}$ 
    else if  $bande_i == 3$  then                                 ▷ si la taille de la bande est 3
      for colonne  $j \rightarrow 2$  do
        Créer une nouvelle clause avec  $x_{i,j}$ 
      end for
    end if
  else if nombre de bandes == 2 then
    Créer une nouvelle clause avec  $(x_{i,0} \vee \neg x_{i,1} \vee x_{i,2})$ 
  end if
  Joindre  $(\wedge)$  toutes les clauses
end for
```

Chapitre 2

Les Bandes

2.1 Problème et notation

Une suite de cases noires représente une *bande*.

Une bande appartient à une ligne ou colonne et commence à une position donnée. Comme il peut y avoir plusieurs bandes sur une même ligne ou colonne, chaque bande doit être identifiable.

2.1.1 Variables

On représente une bande par

$$LBande_{pos,ID,start}$$

où,

$L|C$ précise s'il s'agit d'une ligne ou d'une colonne.

pos désigne le numéro de la ligne ou la colonne où est située la bande.

ID désigne, s'il existe plusieurs bandes sur la ligne ou colonne, la bande que l'on considère.

$start$ désigne le numéro de colonne ou ligne à partir de laquelle commence la bande.

On définit deux fonctions donnant la longueur d'une bande :

$L(pos, ID)$ donnant la longueur de la ID^{eme} bande à la ligne pos .

$C(pos, ID)$ donnant la longueur de la ID^{eme} bande à la colonne pos .

2.1.2 Sémantique

On définit pour sémantique, dans le cas d'une ligne (raisonnement équivalent pour chaque colonne) :

$$LBande_{pos,ID,start} = \top \iff (x_{pos,j} = \top) \wedge (x_{pos,j+1} = \perp) \\ \forall j \in [start, start + \delta], \delta = L(pos, ID) \quad (2.1)$$

TABLE 2.1 – Grille 3×3

Par exemple, on aura pour la grille 2.1 :

$\text{LBande}_{3,1,2} = \top$, sachant $L(3, 2) = 2$

et

$x_{3,j} = \top$, pour tout j dans l'intervalle $[3, 2 + L(3, 2)]$

et

$x_{3,2+L(3,2)+1} = \perp$

où $x_{pos,j+1}$ donne \perp que ce soit parce qu'il existe une case vide ou parce qu'on excède la taille de la grille.

2.1.3 Codage

2.2 Contraintes

Selon la définition donnée d'une bande, la donnée décisive est la valeur du paramètre *start*.

1. Une bande est contenue dans la grille ou dans l'espace délimité par la bande suivante.

$$\max(start_k) = \begin{cases} start_{k+1} - L(pos, ID_k) - 1, & \text{si } ID > 1 \\ n - L(pos, ID_k), & \text{sinon} \end{cases}$$

où,

n est la taille de la grille

ID_k est la k^{eme} bande

$k = \{0, \dots, b\}$

b est le nombre de bandes sur la ligne/colonne, $b = \sum ID$

$start_k$ est la position de départ de la bande avec ID_k .

2. Deux bandes de la même ligne/colonne ne peuvent se superposer.

$$\min(start_k) = \begin{cases} start_{k-1} + L(pos, ID_{k-1}) + 1, & \text{si } ID > 1 \\ 0, & \text{sinon} \end{cases}$$

où,

ID_k est la k^{eme} bande

$k = \{0, \dots, b\}$

b est le nombre de bandes sur la ligne/colonne, $b = \sum ID$

$start_k$ est la position de départ de la bande avec ID_k .

3. **Pour toutes** bandes différentes appartenant à la même ligne/colonne, **il existe** au moins v cases vides :

où, v se donne comme :

Soit b , le nombre de bandes sur une ligne, $b = \sum ID$

Soit v , le nombre de cases vides,

alors $\min(v) = b - 1$

Ce qui peut se traduire comme, pour toutes lignes i ,

$$\bigwedge_{k \in \{0, \dots, b\}} \left(\bigvee_{\delta = \{0, \dots, \max(start_{k+1}) - \min(start_k)\}} \neg x_{i, j+\delta} \right)$$

Ou encore,

$$\bigwedge_{k \in \{0, \dots, b\}} \left(\bigvee_{\substack{j = \{\min(start_k), \dots, j'-1\} \\ j' = \{j+1, \dots, \max(start_{k+1})\}}} \neg x_{i, j'-j} \right)$$

Intuitivement, il s'agit de cerner les cases vides plausibles entre deux bandes :

⊤	⊥	⊥	⊥	⊤	⊤
↑ j				↑ j'	

2.2.1 Une bande

Le cas où il n'y a qu'une bande peut être vu comme la contraintes des possibles positions d'une bande relativement à un espace donné disponible.

Cette interprétation permet alors de réutiliser le cas "une bande" comme sous problème résoluble du cas de multiples bandes. Soit l'exemple suivant :

(a) Positions d'une bande 2 dans un espace 5

⊤	⊥	⊥	⊥	⊥
⊥	⊤	⊥	⊥	⊥
⊥	⊥	⊤	⊥	⊥
⊥	⊥	⊥	⊤	⊥

(b) Positions *start* possibles

Dans la table 2.2a, on représente les différentes façons dont se positionne une bande de taille 2 dans un espace de taille 5, on y voit 4 possibilités.

Le nombre de possibilités p est donné relativement à la taille de l'espace n dans lequel se positionne la bande et la taille de cette dernière :

$$p = n - L(pos, ID) + 1$$

où,

p est le nombre de position possible.

n est la taille de l'espace considéré.

On ne considère en réalité que la position de *start* à laquelle commence la bande. Ce que décrit la table 2.2b, on s'aperçoit que *start* ne prendra jamais comme position la dernière colonne, ce dans tous les cas possibles de l'exemple.

a	$\neg b$	$\neg c$	$\neg d$	$\neg e$
$\neg a$	b	$\neg c$	$\neg d$	$\neg e$
$\neg a$	$\neg b$	c	$\neg d$	$\neg e$
$\neg a$	$\neg b$	$\neg c$	d	$\neg e$
$a \oplus b \oplus c \oplus d$				$\wedge \neg e$

(c) Positions *start* possibles

a	$\neg b$	$\neg c$	$\neg d$	$\neg e$	$\neg f$
$\neg a$	b	$\neg c$	$\neg d$	$\neg e$	$\neg f$
$\neg a$	$\neg b$	c	$\neg d$	$\neg e$	$\neg f$
$\neg a$	$\neg b$	$\neg c$	d	$\neg e$	$\neg f$
$a \oplus b \oplus c \oplus d$				$\wedge \neg e \wedge \neg f$	

(d) Bande 3 dans espace 5

On peut tenter une première approche, soit les cases a, b, c, d , pouvant potentiellement être la case *start* et la case e qui ne le sera jamais.

On a alors, sur la table 2.2c : "Soit a ou soit b ou soit c ou soit d et pas e ", c'est-à-dire $(a \oplus b \oplus c \oplus d) \wedge \neg e$ où \oplus représente le *ou exclusif*.

En excluant la dernière colonne, on obtient une matrice carrée dont la diagonale présente toutes les positions acceptables.

La taille de la matrice correspond au nombre des positions possibles $n - L(pos, ID) + 1$

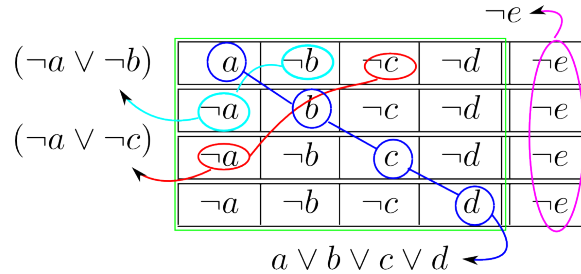


FIGURE 2.1 – Génération de clauses

Pour cette matrice, il est possible de formuler la contrainte en forme normale conjonctive suivante :

$$(a \vee b \vee c \vee d) \wedge (\neg a \vee \neg b) \wedge (\neg a \vee \neg c) \wedge (\neg a \vee \neg d) \wedge (\neg b \vee \neg c) \wedge (\neg b \vee \neg d) \wedge (\neg c \vee \neg d)$$

À quoi, on ajoutera la clause unaire $\neg e$.

Il est également à remarquer, comme sur la table 2.2d, que le nombre de clause représentant le nombre de colonne qui ne pourront jamais être une position *start*, est égal à :

$$L(pos, ID) - 1 \text{ pour une ligne}$$

$\mathbb{C}(pos, ID) - 1$ pour une colonne

Suite à ce qui précède, on obtient en concaténant les observations, la formulation suivante :
Pour toutes lignes i , selon un espace n défini,

$$\left(\bigvee_{j \in \{0, \dots, p\}} LB_{i, ID, j} \right) \bigwedge_{\substack{j, j' \in \{0, \dots, n\} \\ j \leq j'}} (\neg LB_{i, ID, j} \vee \neg LB_{i, ID, j'}) \bigwedge_{j \in p, \dots, n} \left(\bigvee_{k=1}^{n=1} \neg LB_{i, ID, j} \right)$$

où,

$p = n - \mathbb{L}(pos, ID) + 1$, soit le nombre de positions possibles

n est l'espace disponible

$LB_{i, ID, j} \equiv LB_{bande_{pos, ID, start}}$

2.3 Représentation par bandes

2.3.1 Observation : position minimale, maximale et sous-espace

Soit une colonne d'une longueur de 8 cases avec pour contrainte 3 bandes de taille 1, 2 et 1.

La première bande pourra prendre la position minimum $x_{1,0}$ et maximum $x_{1,\alpha}$.

La valeur de α peut être obtenue à partir du nombre de contraintes, de la taille des différentes bandes et de la règle selon laquelle deux bandes sont séparés d'au moins une case.

Dans l'exemple, on obtient :

$c = 3$, où c est le nombre de contraintes

$l = 8$ est la longueur de la colonne

$L_{total} = 4$, est la longueur cumulée des bandes dans la colonne

$\alpha = l - (L_{total} + (c - 1))$

























0			
1			
2			
3			
4			
5			
6			
7			
	<i>min</i>		<i>max</i>

TABLE 2.2 – position *max* et *min* sur une colonne 8×1 avec la contrainte 1 2 1

On voit dans le tableau 2.2 qu'il est possible de connaître *a priori* le nombre restreint de positions possibles pour la première bande.

On voit aussi que chaque possibilité délimite un *sous-espace* (zone grisée) au sein duquel devront être placées les bandes restantes.

Il est alors possible d'appliquer la même analyse quant à la détermination de la position de la bande suivante au sein du sous-espace.

Une fois les positions de la seconde bande déterminées, elles délimiteront l'espace restant pour la dernière bande.

Suite à cela, il est possible d'élaborer une méthode récursive, ou faussement récursive, au nombre d'itérations réduit, permettant d'énumérer les possibilités de placement des bandes.

2.3.2 Énumération récursive

						$\updownarrow L(1, 1) = 1$	$x_{1,1} = \top$
						+1	$x_{1,2} = \perp$
						# itération	$x_{1,3} = \perp$
						$\updownarrow L(1, 2) = 2$	$x_{1,4} = \top$
							$x_{1,5} = \top$
						+1	$x_{1,6} = \perp$
						$\updownarrow L(1, 3) = 1$	$x_{1,7} = \top$
							$x_{1,8} = \perp$
•		r_1		r_1		r_3	récursion

TABLE 2.3 – Énumération récursive sur une colonne 8×1 avec la contrainte 1 2 1

Algorithm 2 Énumération des bandes

```

for ligne  $i \rightarrow n$  do
  for colonne  $j \rightarrow n$  do nop();
  end for
end for

```
