Université Larbi ben M'Hidi **13 Mai 2025**
Faculté des Sciences Exactes & Sciences de la Nature et de la Vie **Dr. KOUAH Sofia**
Département MI **1ière Année Master AD**

Examen Final de Programmation par Contraintes

Questions de compréhension : Répondre uniquement à trois questions (3 Pts)

1. Qu'est-ce qu'un problème de satisfaction de contraintes (CSP) ?
2. Quelle est la différence entre une contrainte en extension et une contrainte en intension ?
3. Quelle est la différence entre une affectation partielle et une affectation totale dans un CSP ?
4. Comment distingue-t-on un CSP sur-contraint d'un CSP sous-contraint ? Expliquez avec des exemples ou situations simples.

Exercice 1 (8 Pts)

Soit le CSP suivant : CSP = (X, D, C), tels que :

- X = {X1, X2, X3}
- D(X1) = D(X2) =D(X3) = {3,6,9}
- C = {C1, C2} avec :
  - C1 : X1< X2
  - C2 : X2< X3

1) Ce CSP est-il **consistant** ? Justifier votre réponse.
2) On se donne **l'ordre statique** suivant **d'instanciation de variables :** (X1, X2, X3), ainsi que **l'ordre statique** suivant **de choix des valeurs** du domaine des variables : (3, 6, 9).
   Que représente cette information, quel est **son type** et **son utilité** ? est ce qu'elle est **toujours fiable** ?
3) En adoptant l'ordre de la question précédente d'instanciation de variables et de choix des valeurs, appliquer les algorithmes suivants sur ce CSP, en donnant l'arbre de recherche qui correspond à chacun d'eux :
   a) L'algorithme **Générer et Tester**.
   b) L'algorithme **Simple Retour Arrière.**
   c) L'algorithme **d'anticipation.**
4) D'après les résultats obtenus en 3, expliquer la différence entre ces algorithmes.

Exercice 2 (4,5 Pts)

Soit le programme ChocoSolver suivant :

```
- Model model = new Model("ChocoSolver");                          …. (1)
- IntVar a = model.intVar("a", new int[ ]{4, 6, 8});               …. (2)
- IntVar b = model.intVar("b", 0, 2);                              …. (3)
- model.arithm(a, "+", b, "<", 8).post();                          …. (4)
- int i = 1;                                                       …. (5)
- while (model.getSolver().solve( )) {                             …. (6)
- System.out.println("Solution " + i++ + " found : " + a + ", " + b); }  …. (7)
```

1. Donner le problème CSP correspondant.
2. Donner la représentation graphique correspondante.
3. Comment étendre ce modèle à trois variables a, b, c avec la contrainte a + b + c < 10 ?

Exercice 3 (4,5 Pts)

On considère le CSP suivant : P = (X, D, C), tels que :

- X = {X1, X2, X3, X4}
- D(X1) = D(X2) = {a, b}, D(X3) = {b, c}, D(X4) = {a, c}
- C = {c1 : X1 ≠ X2, c2 : X2 ≠ X3, c3 : X1 ≠ X4, c4 : X3 ≠ X4}

1) Est-ce que P est **nœud-consistant** ?
2) Est-ce que P est **arc-consistant** ?
3) Quel est la **différence entre consistance de nœud et consistance d'arc** ?
4) Donner la **représentation graphique** de P ?

Bon courage

Université Larbi ben M'Hidi **13 Mai 2025**
Faculté des Sciences Exactes & Sciences de la Nature et de la Vie **Dr. KOUAH Sofia**
Département MI **1ière Année Master AD**

Corrigée de l'Examen Final de Programmation par Contraintes

Questions de compréhension : Répondre uniquement à trois questions (3 Pts)

1. Un CSP est défini par un ensemble de variables, chacune avec un domaine de valeurs possibles, et un ensemble de contraintes précisant les combinations valides de valeurs. L'objectif est de trouver une affectation des variables qui satisfait toutes les contraintes.
2. **En extension** : liste explicite des combinaisons autorisées (ou interdites). Ex. : C(x, y) = {(1,2), (2,3)}
   En intension : définie par une formule logique ou arithmétique. Ex. : C(x, y) ⇔ x + y ≤ 5
3. - Affectation partielle : seules certaines variables sont attribuées.
   - Affectation totale : toutes les variables du CSP ont une valeur.

Seule une affectation totale valide est considérée comme solution du CSP.

4. **Sur-contraint** : trop de contraintes → aucune solution possible. Ex. : x, y, z ∈ {1,2} avec x ≠ y, y ≠ z, x ≠ z
   Sous-contraint : pas assez de contraintes → trop de solutions, dont certaines peu pertinentes. Ex. : x, y ∈ {1,2,3} sans contrainte.

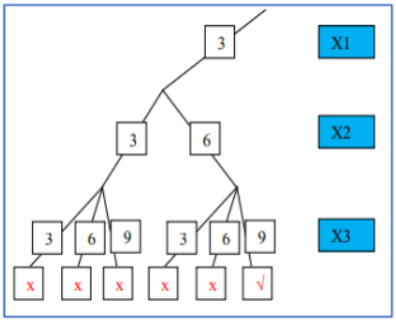
Exercice 1 : (8 Pts)

1. Le CSP est consistant **(0.75 Pts),**

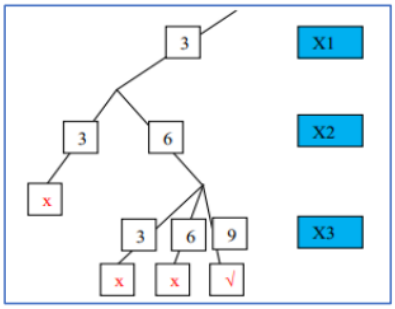
Justification : Il admet au moins une solution **(0.25 Pts)**, par exemple, l'instanciation (X1, X2, X3) = (3,6,9) est une solution puisqu'elle satisfait chacune des deux contraintes du CSP (affectation totale et consistante) **(0.25 Pts)**

2. Cette information **est une heuristique** (**0.25 Pts**), elle est **de type statique (0.25 Pts**). Elle permet de **guider la recherche afin de minimiser la combinatoire (0.5 Pts)**. Les heuristiques **ne sont pas toujours fiables (0.25 Pts).**
3.

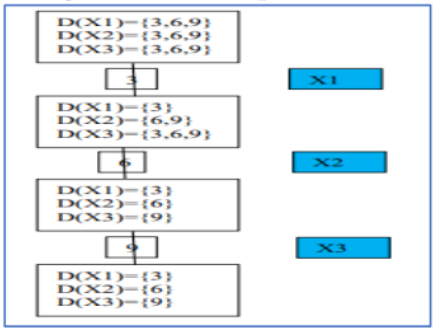
a) L'algorithme Générer et Tester (1.5 Pts)



b) L'algorithme Simple Retour Arrière (1.5 Pts)



c)L'algorithme d'anticipation (1.5 Pts)



4. Comparaison des algorithmes (1 Pt) : l'algorithme « générer et tester » est l'algorithme de base des autres algorithmes, il est **simple** et **intuitif**. En revanche, il pose le **problème d'explosion combinatoire**. L'algorithme « simple retour arrière » est une amélioration de ce dernier, en introduisant le retour arrière donc, le nombre d'états est minimisé, mais le problème de la combinatoire reste toujours posé. L'algorithme d'anticipation est une amélioration de l'algorithme simple retour arrière en ajoutant une étape de filtrage qui sert à minimiser l'explosion combinatoire.

Exercice 2 (4,5 Pts):

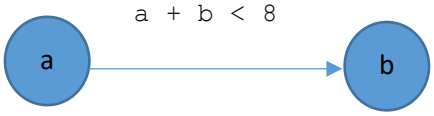
1. Le **CSP** contient: (**1.5 pts**)

- **Variables** :
  - a ∈ {4, 6, 8}
  - b ∈ {0, 1, 2}
- **Contrainte** : a + b < 8

Objectif : trouver toutes les **paires (a, b)** satisfaisant cette contrainte.

2. Représentation graphique correspondante : On peut représenter le CSP sous forme de **graphe biparti** :

- **Nœuds Variables** : a, b
- **Domaines** :
  - a → {4, 6, 8}
  - b → {0, 1, 2}



(**1.5 pts**)

On peut avoir une représentation détaillée (a**rêtes** relient chaque couple (a, b) satisfaisant la contrainte)

3. Modifications à apporter (**1.5 pts**)**:**

IntVar c = model.intVar("c", 0, 2);// on a choisi l'intervalle [0,2] pour la variable c
model.arithm(a, "+", b, "+", c, "<", 10).post();

Exercice 2 : (4,5 Pts)

1) Oui, consistant de nœud, puisque toutes les contraintes ont deux variables (Plus d'une seule variable) (**1 Pts**).

2) Oui, car pour tout couple de variables (Xi, Xj) de X et pour toute valeur vi appartenant à D(Xi), il existe une valeur vj appartenant à D(Xj) telle que l'affectation partielle {(Xi,vi),(Xj,vj)} satisfasse toutes les contraintes binaires de C (**1 Pts**).

3) Différence entre consistance de nœud et consistance d'arc : Consistance de nœud porte sur un seul nœud et concerne les contraintes unaires, par contre la consistance d'arc porte sur les arcs et concerne donc les contraintes binaires. La consistance d'arcs est plus forte que la consistance de nœuds (**1 Pts**).

4) **Représentation Graphique** (**1,5 Pts**)