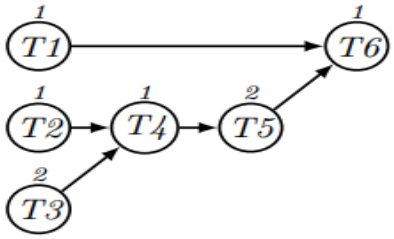
Université Larbi ben M'Hidi
Faculté des sciences exactes & sciences de la nature et de la vie
Département MI

21 Mai 2023
Dr. KOUAH Sofia
1ière Année Master AD

Examen Final de Programmation par Contraintes

Exercice 1 : (6 Pts)

Nous avons un ensemble de tâches : T1, T2, T3, T4, T5 et T6, que nous devons planifier dans un délai de 6 heures, tout en respectant l'ordre indiqué par le schéma suivant :



Telle que la tâche en début de flèche doit s'effectuer avant la tâche en fin de flèche. Les durées des tâches sont affichées au-dessus. Chaque tâche ne peut commencer qu'en début d'une heure. La tâche T1 ne peut pas être planifier à la même heure que la tâche T4. Modélisez ce problème comme un CSP.

Exercice 2 : (6 Pts)

Considérez la portion suivante d'un programme Java :

```
Model model = new Model("ProblemCSP");
    IntVar A = model.intVar("A", 1, 3);
    IntVar B = model.intVar("B", 1, 3);
    IntVar C = model.intVar("C", 1, 3);
    IntVar D = model.intVar("D", 1, 3);
    model.arithm(A, "!=", B).post( );
    model.arithm(A, "!=", C).post( );
    model.arithm(B, "!=", C).post( );
    model.arithm(B, "!=", D).post( );
    model.arithm(C, "!=", D).post( );
    model.getSolver( ).solve();
    System.out.println("Solution :");
    System.out.println("A : " + A.getValue( ));
    System.out.println("B : " + B.getValue( ));
    System.out.println("C : " + C.getValue( ));
    System.out.println("D : " + D.getValue( ));
    model.getSolver().printStatistics( );
```

1. Que représente ce programme ? Formalisez le CSP correspondant et donnez la représentation graphique sous-jacente.
2. Commentez brièvement les principales instructions.
3. Qu'est-ce qu'une heuristique ? quelles sont ses types ?

Exercice 3 : (8 Pts)

Soit le problème du placement de 4 reines sur un échiquier 4x4, avec la modélisation CSP suivante :

- **Variables:** X = {X1, X2, X3, X4}
- **Domaines:** D(X1) = D(X2) = D(X3) = D(X4) = {1,2,3,4}
- **Contraintes** : C = {Xi ≠ Xj / i élément_de {1,2,3,4}, j élément_de {1,2,3,4} et i ≠ j} U {Xi+i ≠ Xj+j / i élément_de {1,2,3,4}, j élément_de {1,2,3,4} et i ≠ j} U {Xi-i ≠ Xj-j / i élément_de {1,2,3,4}, j élément_de {1,2,3,4} et i ≠ j}

1. En appliquant l'algorithme simple retour arrière sur ce problème, donner l'arbre d'exécution correspondant (chaque nœud de l'arbre doit être représenté par un dessin d'échiquier modélisant l'affectation partielle en cours).
2. Même question pour l'algorithme d'Anticipation/nœud.
3. Quelle est la différence entre consistance de nœud et consistance d'arc ?
4. Quelle est la différence entre les algorithmes AC1 et AC3 ?

Bon courage

Université Larbi ben M'Hidi
Faculté des sciences exactes & sciences de la nature et de la vie
Département MI

21 Mai 2023
Dr. KOUAH Sofia
1ière Année Master AD

Corrigé type

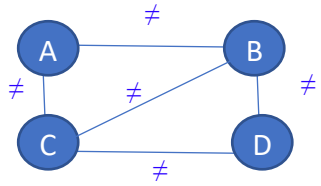
Exercice 1 :

- **Variables** : X= {S1, S2, S3, S4, S5, S6}, tel que Si représente le temps de début de la tâche i. **(0,5 Pt)**
- **Domaines** : D(Si) = {0, …, 6-Pi}, tel que Pi est la durée de la tâche i. Par exemple, D(S1) = {0, 1,2,3,4,5}. **(1,5 Pts)**
- **Contraintes** : C= {C1, C2, C3, C4, C5, C6} tels que : **(4Pts)**
  C1= {S1 + 1 ≤S6}, C2= {S2 + 1 ≤ S4}, C3= {S3 + 2 ≤ S4}, C4= {S4 + 1 ≤ S5}, C5= {S5 + 2 ≤ S6}, C6= {S1 ≠ S4}

Exercice 2 :

1.

- Ce programme représente la résolution du problème de coloration de graphes, tel que deux nouds adjacents ne doivent pas être coloriés par la même couleur. Le programme utilise le Solveur ChocoSolver. **(1 Pt)**
- Formalisation : CSP= (X,D,C), tel que : X={A,B,C,D}, D(A)=D(B)=D(C)=D(D)= {1,2,3}, C= {C1,C2,C3,C4,C5} tels que :
- C1= {A≠ B}, C2={A ≠C}, C3={B≠ C}, C4={B≠D}, C5-{C≠D}. **(1 Pt)**
- Représentation graphique : **(1 Pt)**



2. Commentaires sur les principales instructions. **(1,25 Pts)**

- Model model = **new** Model("Coloration de graphe"); // Création d'un modèle CSP
- // Définition des variables (une variable pour chaque sommet)

```
IntVar A = model.intVar("A", 1, 3);  IntVar B = model.intVar("B", 1, 3);
IntVar C = model.intVar("C", 1, 3);  IntVar D = model.intVar("D", 1, 3);
```

- // Définition des contraintes (les sommets adjacents ne peuvent pas avoir la même couleur)

```
model.arithm(A, "!=", B).post();model.arithm(A, "!=", C).post(); model.arithm(B, "!=", C).post();model.arithm(B, "!=", D).post();
model.arithm(C, "!=", D).post();
```

- // Recherche d'une solution

```
model.getSolver().solve();
```

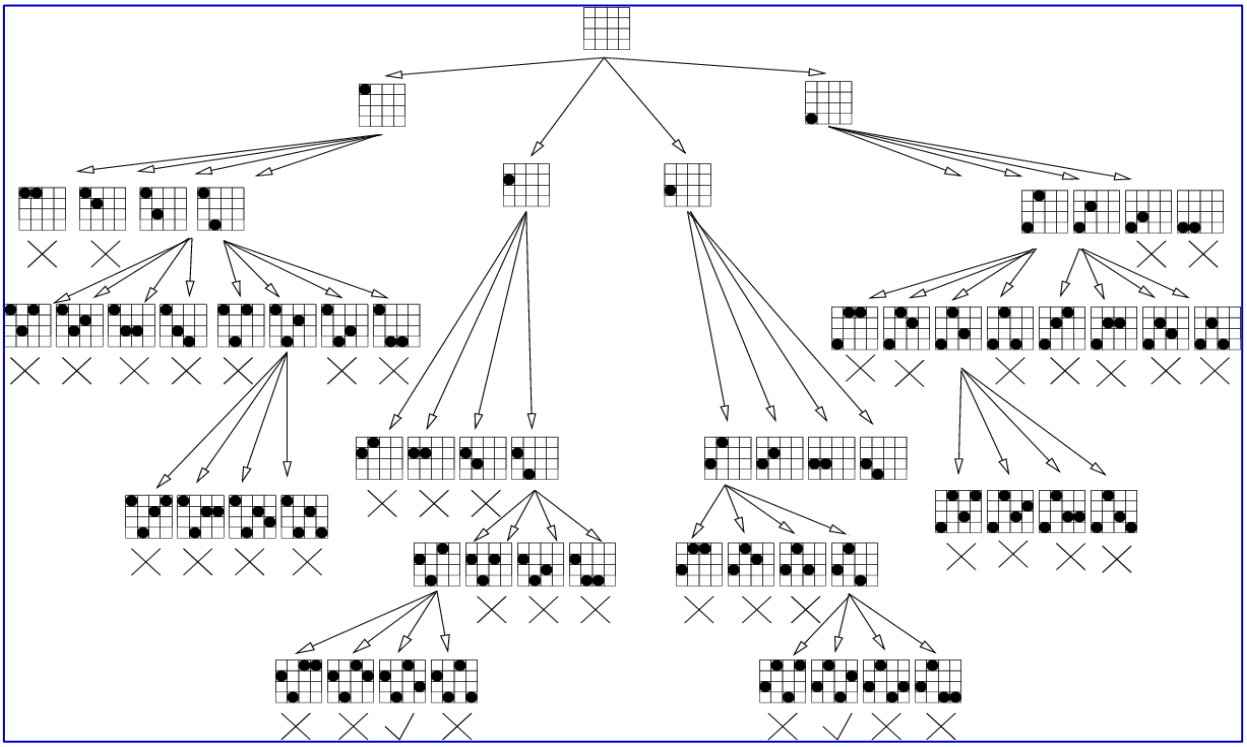
- // Affichage de la solution

```
System.out.println("Solution :");
    System.out.println("A : " + A.getValue()); System.out.println("B : " + B.getValue()); System.out.println("C : " +
    C.getValue());System.out.println("D : " + D.getValue());
// Affichage des statistiques
  model.getSolver().printStatistics();
```

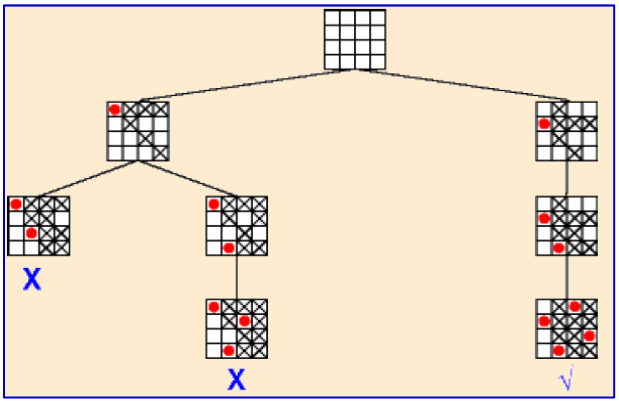
- Heuristique : une heuristique est une règle non systématique (dans le sens où elle n'est pas fiable à 100%) qui donne des indications sur la direction à prendre dans l'arbre de recherche. **(1 Pt)**
- Ses types : Deux types d'heuristiques sont distingués : Statique (L'ordre d'instanciation des variables est fixée avant la recherche) et dynamique (quand la prochaine variable à instancier est choisie dynamiquement à chaque étape de la recherche). **(0,75Pt)**

Exercice 3 : (8 Pts)

1. Application de l'algorithme simple retour arrière **(3 Pts)**



2. Application de l'algorithme d'Anticipation/nœud (une partie du graphe) **(2 Pts)**



3. La différence entre consistance de nœud et consistance d'arc : (1,5 Pts)

La différence entre la consistance de nœud et la consistance d'arc réside dans les types de contraintes qu'elles vérifient. La consistance de nœud concerne la consistance des variables individuelles avec les contraintes unaires, tandis que la consistance d'arc se concentre sur la consistance des couples de variables avec les contraintes binaires.

4. La différence entre les algorithmes AC1 et AC3 : (1,5 Pts)

L'algorithme qui enlève les valeurs des domaines des variables d'un CSP jusqu'à ce qu'il soit consistant d'arc s'appelle AC (Arc Consistency). Il existe différentes versions de cet algorithme tels que AC1 et AC3.

La différence entre eux réside dans le fait que le l'algorithme AC1 refait toute une passe en reconsidérant chacune des paires sur lesquelles il y a une contrainte. Tandis que, le AC3 ne réapplique la réduction qu'aux arcs dont le domaine de la variable extrémité a été modifié depuis la dernière révision.