Problèmes de satisfaction de contraintes (CSP)

- Cours 1: Introduction
  - Description du domaine d'applications
  - Aperçus des modèles et algorithmes de résolutions
- Cours 2 : Modelisation
  - Concept de bases : variables, domaines, contraintes
  - Exemples
- **Cours 3: Résolution**
  - Concept de base : recherche et propagation
  - exemples

CSP: Résolution

En raisonnement par contraintes, un modèle est construit en utilisant

- des **variables**
- des **domaines** des variables
- des **contraintes** entre les variables

Un tel modèle est appelé **Problème de Satisfaction de Contraintes (CSP)**

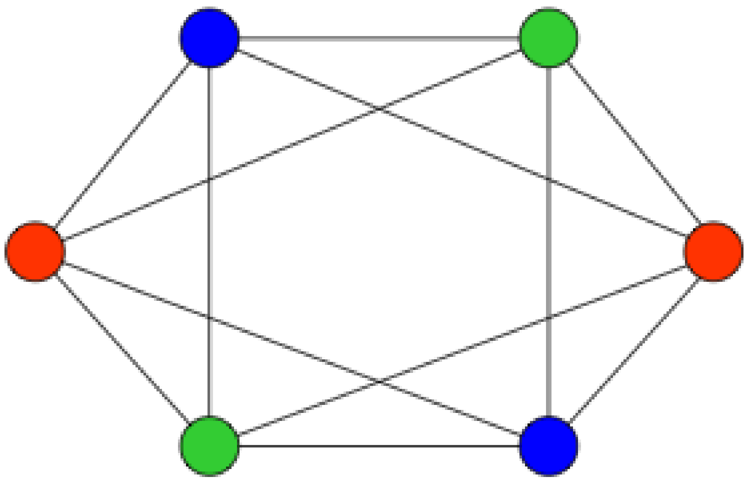
Une **solution** à un CSP est :

- Affecter à chaque variable une valeur de son domaine de manière à satisfaire toutes les contraintes

Problème de coloriage de graphe

Coloriage des nœuds d’un graphe

- Quelle est le nombre **minimal** de couleur tel que deux nœuds adjacents reçoivent des couleurs différentes?



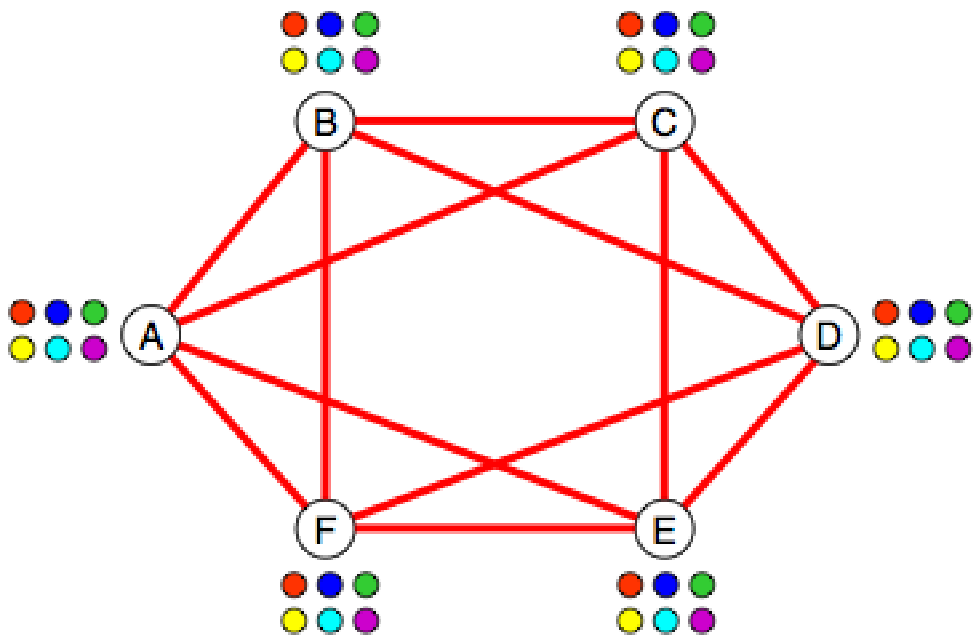
Problème de coloriage de graphe : modèle

Modèle complet :

Variables : A, B, C, D, E, F

Domaines : {rouge, bleu, vert, jaune, blueciel, violet}

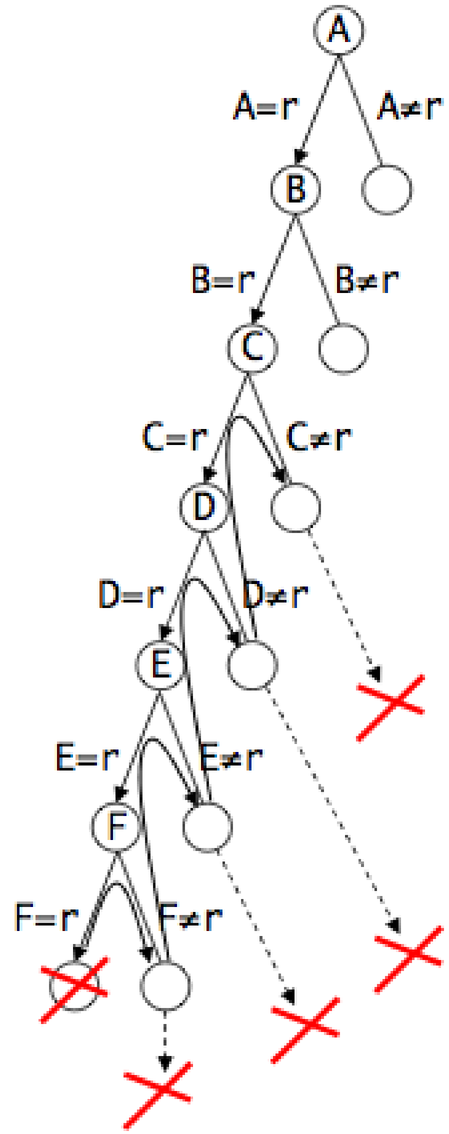
Contraintes : $A \neq B$, $A \neq C$, $A \neq E$, $A \neq F$, $B \neq C$, $B \neq D$, $B \neq F$,
$C \neq D$, $C \neq E$, $E \neq E$, $D \neq F$, $E \neq F$

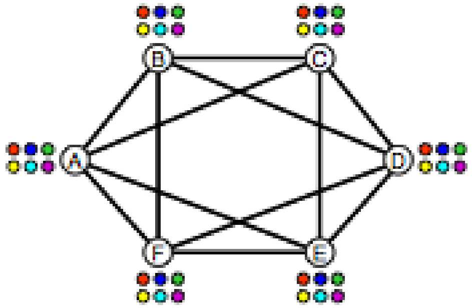


Problème de coloriage de graphe : un solveur simple

$\mathcal{V}$	$\mathcal{D}$
A	{r,b,g,y,c,p}
B	{r,b,g,y,c,p}
C	{r,b,g,y,c,p}
D	{r,b,g,y,c,p}
E	{r,b,g,y,c,p}
F	{r,b,g,y,c,p}

$\mathcal{C}$
A≠B
A≠C
A≠E
A≠F
B≠C
B≠D
B≠F
C≠D
C≠E
D≠E
D≠F
E≠F

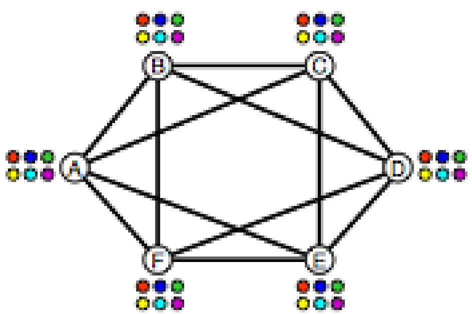


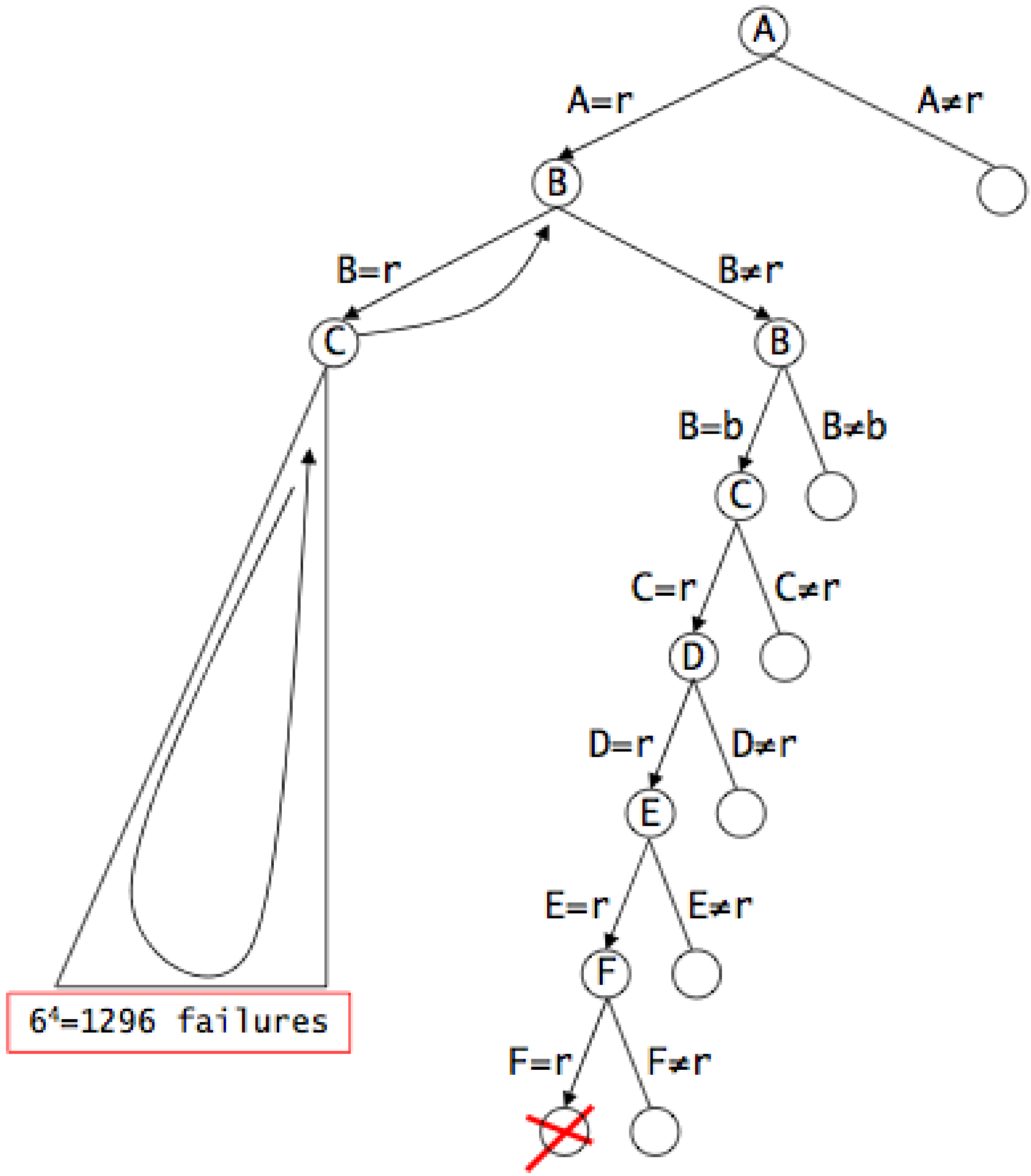


Problème de coloriage de graphe : un solveur simple

$\mathcal{V}$	$\mathcal{D}$
A	{r}
B	{b,g,y,c,p}
C	{r,b,g,y,c,p}
D	{r,b,g,y,c,p}
E	{r,b,g,y,c,p}
F	{r,b,g,y,c,p}

$\mathcal{C}$
A≠B
A≠C
A≠E
A≠F
B≠C
B≠D
B≠F
C≠D
C≠E
D≠E
D≠F
E≠F

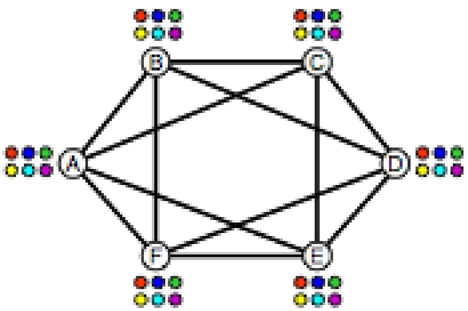


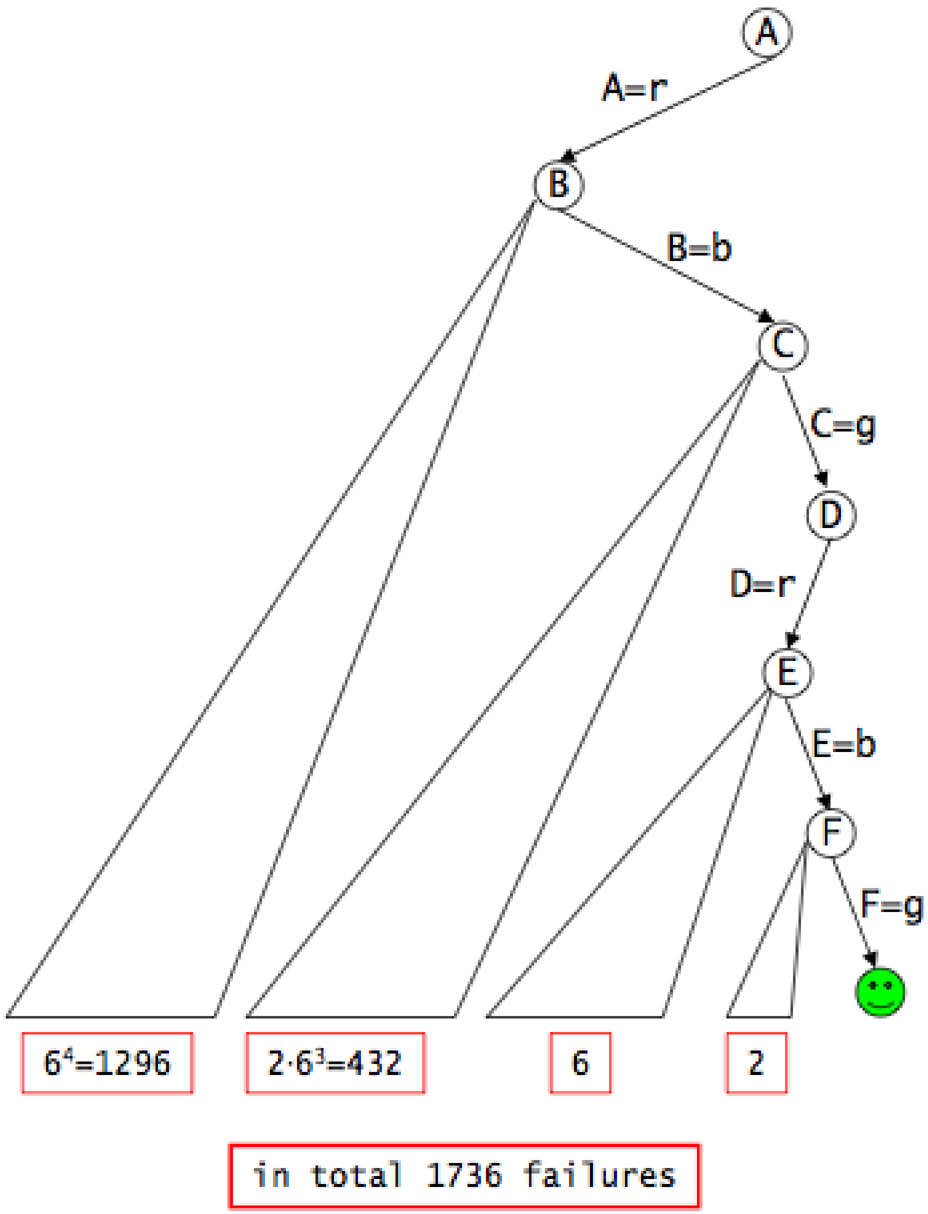


Problème de coloriage de graphe : un solveur simple

$\mathcal{V}$	$\mathcal{D}$
A	{r}
B	{b}
C	{g}
D	{r}
E	{b}
F	{g}

$\mathcal{C}$
A≠B
A≠C
A≠E
A≠F
B≠C
B≠D
B≠F
C≠D
C≠E
D≠E
D≠F
E≠F

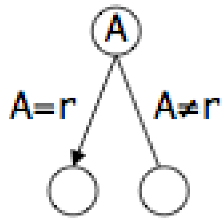


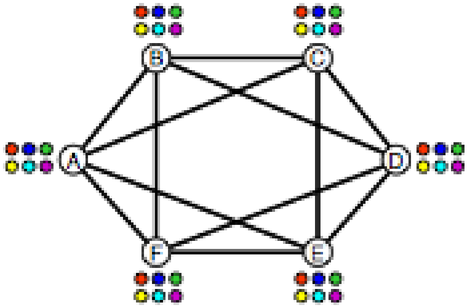


Problème de coloriage de graphe : solveur de contraintes

𝒱	𝒟
A	{r}b,g,y,c,p}
B	{r,b,g,y,c,p}
C	{r,b,g,y,c,p}
D	{r,b,g,y,c,p}
E	{r,b,g,y,c,p}
F	{r,b,g,y,c,p}

𝒞
A≠B
A≠C
A≠E
A≠F
B≠C
B≠D
B≠F
C≠D
C≠E
D≠E
D≠F
E≠F

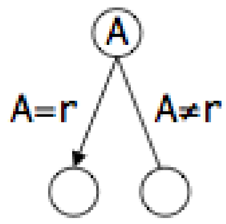


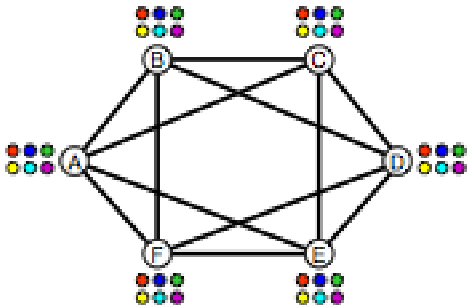


Problème de coloriage de graphe : solveur de contraintes

$\mathcal{V}$	$\mathcal{D}$
A	{r}
B	{b,g,y,c,p}
C	{b,g,y,c,p}
D	{r,b,g,y,c,p}
E	{b,g,y,c,p}
F	{b,g,y,c,p}

$\mathcal{C}$
A≠B
A≠C
A≠E
A≠F
B≠C
B≠D
B≠F
C≠D
C≠E
D≠E
D≠F
E≠F

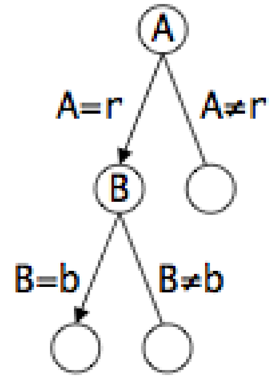


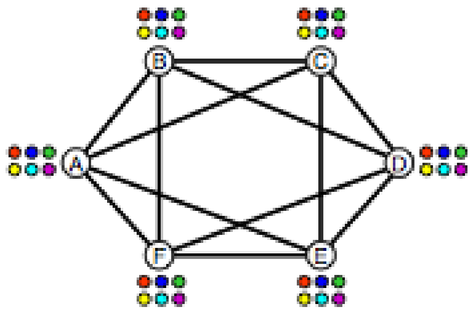


Problème de coloriage de graphe : solveur de contraintes

$\mathcal{V}$	$\mathcal{D}$
A	{r}
B	{b}
C	{g,y,c,p}
D	{r,g,y,c,p}
E	{b,g,y,c,p}
F	{g,y,c,p}

$\mathcal{C}$
A≠B
A≠C
A≠E
A≠F
B≠C
B≠D
B≠F
C≠D
C≠E
D≠E
D≠F
E≠F

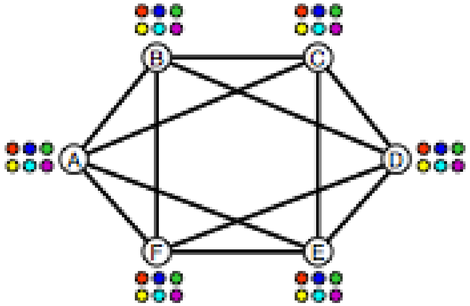


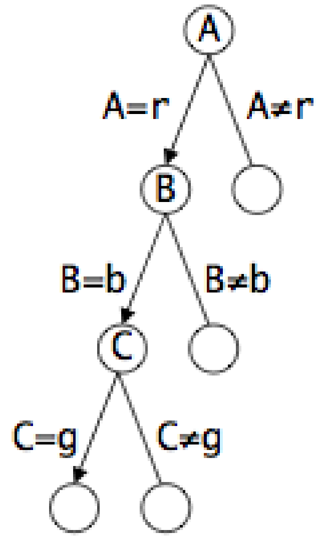


Problème de coloriage de graphe : solveur de contraintes

$\mathcal{V}$	$\mathcal{D}$
A	{r}
B	{b}
C	{g}
D	{r,y,c,p}
E	{b,y,c,p}
F	{g,y,c,p}

$\mathcal{C}$
A≠B
A≠C
A≠E
A≠F
B≠C
B≠D
B≠F
C≠D
C≠E
D≠E
D≠F
E≠F

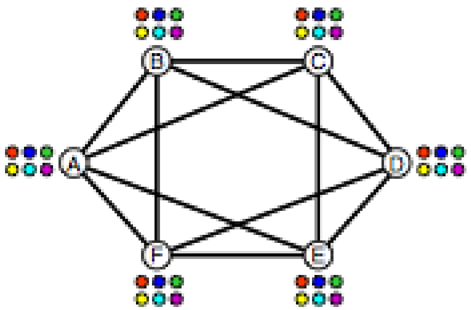


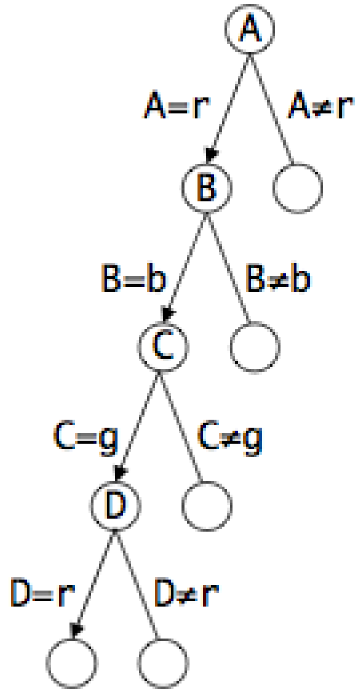


Problème de coloriage de graphe : solveur de contraintes

$\mathcal{V}$	$\mathcal{D}$
A	{r}
B	{b}
C	{g}
D	{r}
E	{b,y,c,p}
F	{g,y,c,p}

$\mathcal{C}$
A≠B
A≠C
A≠E
A≠F
B≠C
B≠D
B≠F
C≠D
C≠E
D≠E
D≠F
E≠F

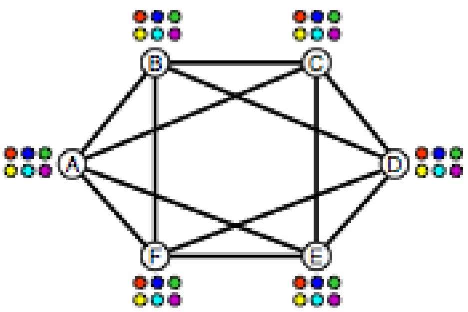


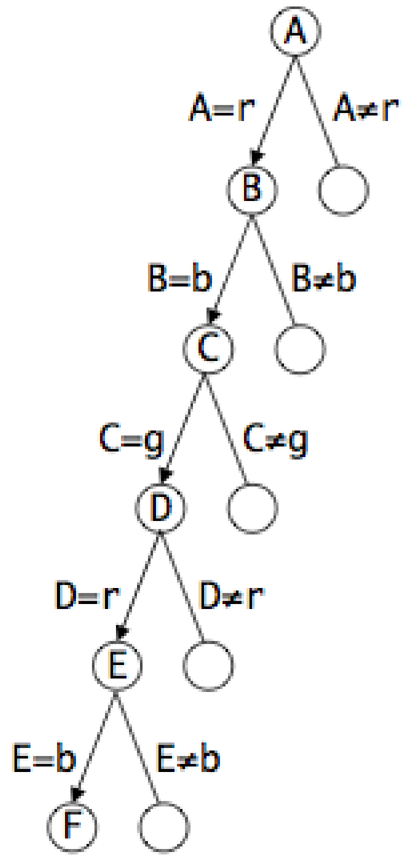


Problème de coloriage de graphe : solveur de contraintes

$\mathcal{V}$	$\mathcal{D}$
A	{r}
B	{b}
C	{g}
D	{r}
E	{b}
F	{g,y,c,p}

$\mathcal{C}$
A≠B
A≠C
A≠E
A≠F
B≠C
B≠D
B≠F
C≠D
C≠E
D≠E
D≠F
E≠F

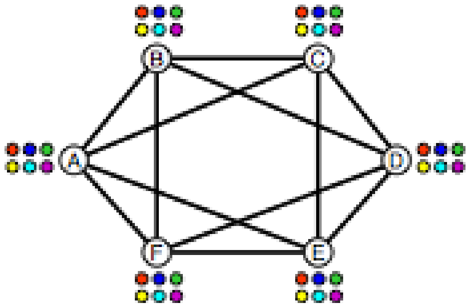


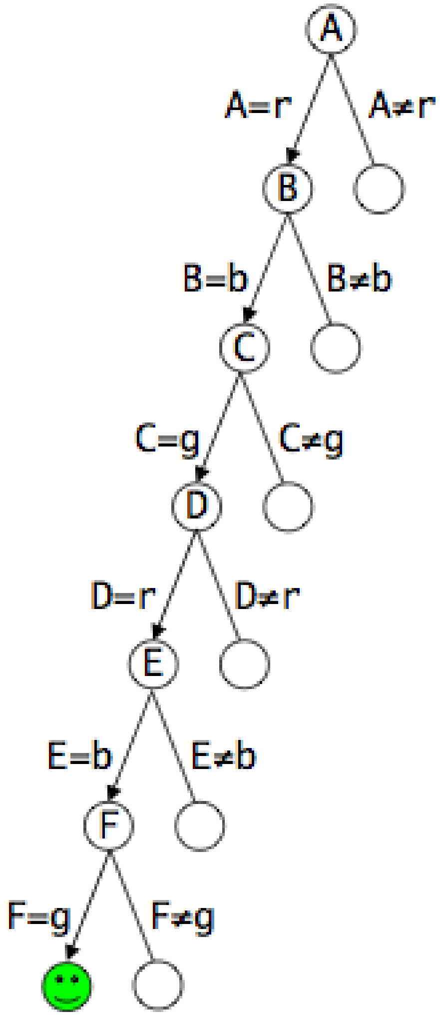


Problème de coloriage de graphe : solveur de contraintes

$\mathcal{V}$	$\mathcal{D}$
A	{r}
B	{b}
C	{g}
D	{r}
E	{b}
F	{g}

$\mathcal{C}$
A≠B
A≠C
A≠E
A≠F
B≠C
B≠D
B≠F
C≠D
C≠E
D≠E
D≠F
E≠F



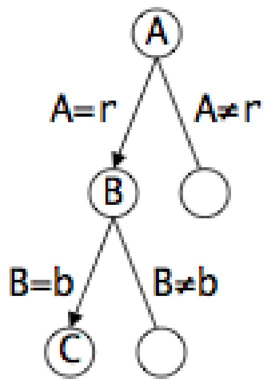


in total 0 failures!

Solveur de contraintes

- Un solveur de contraintes est une succession de deux étapes [recherche - propagation]

Recherche & Propagation



$A \neq B \rightarrow$ Supprimer **r** du **D(B)**

$B \neq C \rightarrow$ Supprimer **r** du **D(C)**

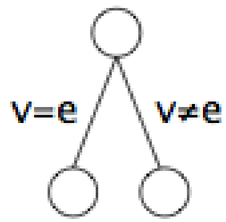
Taille exponentiel en général

Réduit la taille de l'espace de recherche

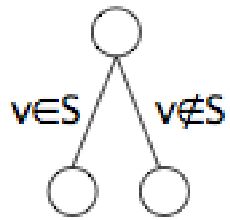
Recherche

Un arbre de recherche peut être défini par

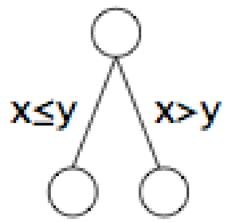
- Enumération
  - Choisir une variable v
  - Choisir une valeur e de D(v)
  - Brancher sur v=e Vs v≠ e



- Partitionnement
  - Choisir une variable v
  - Choisir S ⊆ D(v)
  - Brancher sur v ∈ S Vs v ∉ S



- Contraintes de branchement
  - Par exemple x ≤y Vs x > y



Une feuille de l'arbre :

- toutes les variables ont une valeur singleton (succès) -> solution trouvé
- Une variable admet un domaine vide (conflit) -> retour arrière (backtrack)

Propagation

La propagation de contraintes, permet d'assurer la consistance des domaines avec chaque contrainte

Définition : une contrainte C est arc consistante si toutes les valeurs associés au variables appartiennent aux solutions de la contrainte

$\mathcal{V}$	$\mathcal{D}$
x	{0,1,2,3,4}
y	{0,1,2,3}

$\mathcal{C}$
x < y

← n'est pas arc consistante

$\mathcal{V}$	$\mathcal{D}$
x	{0,1,2}
y	{1,2,3}

$\mathcal{C}$
x < y

← arc consistante

Propagation

Question : comment rendre une contrainte arc consistante?

Réponse : réduire le domaines des variables en éliminant les valeurs inconsistantes

Un tel processus s'appelle propagation de contraintes

Un algorithme de base de propagation d'une contrainte C est :

```
bool Propagate(C,𝒱_C,𝒟) {
    for all v∈𝒱_C {
        for all e∈𝒟(v) {
            find a solution to C with v=e;
            if no solution exists, remove e from 𝒟(v);
            if 𝒟(v) is empty return false;
        }
    }
    return true;
}
```

$\mathcal{V}$	$\mathcal{D}$
x	{0,1,2,~~3~~,~~4~~}
y	{~~0~~,1,2,3}

$\mathcal{C}$
x < y

Propagation

Un CSP admet plusieurs contraintes

Qu'en est t-il des interactions entre les contraintes

Définition :

Un CSP est arc consistant si toutes ses contraintes sont arc consistantes

	$\mathcal{C}$
(1)	$x + y \leq 4$
(2)	alldifferent(x,y,z)

$\mathcal{V}$	$\mathcal{D}$
x	{2,3}
y	{1,2,3,4}
z	{1,2}

$\xrightarrow{(1)}$

$\mathcal{V}$	$\mathcal{D}$
x	{2,3}
y	{1,2}
z	{1,2}

$\xrightarrow{(2)}$

$\mathcal{V}$	$\mathcal{D}$
x	{3}
y	{1,2}
z	{1,2}

$\xrightarrow{(1)}$

$\mathcal{V}$	$\mathcal{D}$
x	{3}
y	{1}
z	{1,2}

$\xrightarrow{(2)}$

$\mathcal{V}$	$\mathcal{D}$
x	{3}
y	{1}
z	{2}

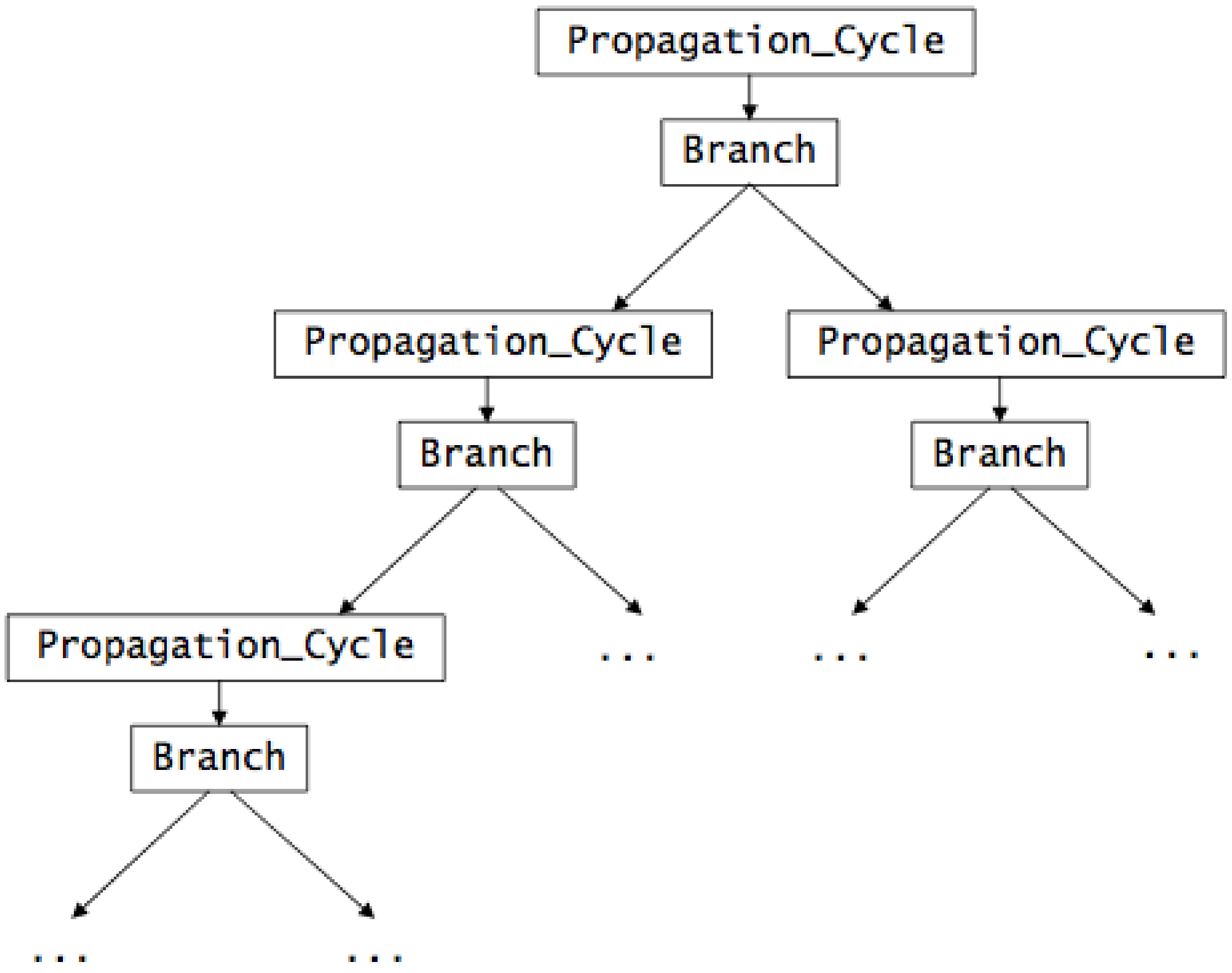
Cycle de propagation

Plus formellement, l'arc consistance d'un CSP peut être maintenu par un cycle de propagation

```
bool Propagation_Cycle(𝒱,𝒟,𝒞){
    Q := 𝒱;
    while Q nonempty {
        pick v in 𝒱;
        Q := Q - v;
        for all constraints C in 𝒞 containing v {
            if (Propagate(C,𝒱_C,𝒟) = false) return false;
            if a domain of variable x has changed, set Q := Q + x;
        }
    }
    return true;
}
```

Processus de résolution

Recherche + propagation de contraintes



Processus de résolution

Recherche + propagation de contraintes

```
bool Solve(𝒱,𝒟,𝒞){
    if |𝒟(v)| = 1 for all v in 𝒱 {
        return true;   // we found a solution!
    }
    else {
        if (Propagation_Cycle(𝒱,𝒟,𝒞) = false) {
            return false;
        }
        else {
            choose variable v in 𝒱 with |𝒟(v)| > 1;
            choose element e in 𝒟(v);
            return ( Solve(𝒱,𝒟,𝒞+‘v=e’) OR Solve(𝒱,𝒟,𝒞+‘v≠e’) );
        }
    }
}
```

Complexité de la propagation

Considérant une contrainte simple x < y défini par la table suivante :

x \ y	~~0~~	1	2	3
0	~~f~~	t	t	t
1	~~f~~	f	t	t
2	~~f~~	f	f	t
~~3~~	~~f~~	~~f~~	~~f~~	~~f~~
~~4~~	~~f~~	~~f~~	~~f~~	~~f~~

t = true
f = false

$x \in \{0,1,2,\cancel{3},\cancel{4}\}$
$y \in \{\cancel{0},1,2,3\}$
$x < y$

Tester toutes les valeur de D(x)

Tester toutes les valeurs de D(y)

Ce processus est en O(|D(x)| x |D(y)|)

Attention : complexité en général

Complexité de la propagation

Considérant encore une fois la contrainte simple x < y :

Exemple :

```
x∈{0,1,2,3,4}
y∈{0,1,2,3}
x < y
```

Noter que cette contrainte est arc consistante ssi
min(x) < min(y) et max(x) < max(y)

On peut donc établir l'arc consistance de cette contrainte en :

- éliminant de D(y) toutes les valeurs <= min(x)
- éliminant de D(x) toutes les valeurs >= max(y)

Complexité en O(1) [mettre à jour les bornes]

Pour certaines contraintes la complexité est largement en dessous du pire cas.

Propagation : contraintes n-aires

- Quoi faire avec des contraintes de plus de 2 variables?
- Consistance d'yper-arc: étendre l'arc consistance à un nombre arbitraire de variables
- Déterminer la hyper-arc consistance est plus difficile
- Solution possible : consistance aux bornes

Consistance aux bornes

- CSP arithmétique: les contraintes sont sur des entiers
- intervalles: [l ..u] représente l'ensemble des valeurs {l , l + 1, . . . , u}
- Idée: Utiliser la consistance sur les réels et examiner seulement les bornes (inférieurs et supérieurs) du domaine de chaque variable
- Définir min(D,x) comme l'élément minimum dans le domaine de x, pareil max(D, x)

Comment obtenir un CSP bornes-consistant?

- Etant donné un domaine D, on doit modifier les bornes, de sorte que le résultat est bornes- consistant
- Utilisation de règles de propagation
- Exemple:
  - $X = Y + Z$ équivalent à $Y = X - Z$ et $Z = X - Y$
  - Raisonner avec max et min
  - $X \geq \min(D,Y)+\min(D,Z)$, $X \leq \max(D,Y)+\max(D,Z)$
  - $Y \geq \min(D,X)-\max(D,Z)$, $Y \leq \max(D,X)-\min(D,Z)$
  - $Z \geq \min(D,X)-\max(D,Y)$, $Z \leq \max(D,X)-\min(D,Y)$
  - cela donne des règles de propagation

Comment obtenir un CSP bornes-consistant?

Exemple :

- $X = Y + Z$, $D(X) = [4..8]$, $D(Y) = [0..3]$, $D(Z) = [2..2]$

- Les règles de propagation donnent:
  - $(0+2=)2 \leq \mathbf{X} \leq 5(=3+2)$
  - $(4-2=)2 \leq \mathbf{Y} \leq 6(=8-2)$
  - $(4-3=)1 \leq \mathbf{Z} \leq 8(=8-0)$

- Les domaines peuvent être réduits:
  - $D(X) = [4..5]$, $D(Y) = [2..3]$, $D(Z) = [2..2]$

D'autres règles de propagation

- $4W + 3P + 2C \leq 9$
- $W \leq 9/4 - 3/4 \min(D,P) - 2/4 \min(D,C)$
- $P \leq 9/2 - 4/3 \min(D,W) - 2/3 \min(D,C)$
- $C \leq 9/2 - 4/2 \min(D,W) - 3/2 \min(D,P)$

- Etant donné un domaine initial
  $D(W) = [0..9]$, $D(P) = [0..9]$, $D(C) = [0..9]$ on détermine
  que $W \leq 9/4$, $P \leq 9/3$, $C \leq 9/2$

- nouveau domaine:
  $D(W) = [0..2]$, $D(P) = [0..3]$, $D(C) = [0..4]$

Inégalités $Y \neq Z$

- Les inégalités ($Y \neq Z$) donnent des règles de propagation très faibles

- Seulement, si une des deux variables prend une valeur fixe qui est égale au minimum ou maximum de l'autre, il y a propagation

- D(Y) = [2..4], D(Z) = [2..3] pas de propagation

- D(Y) = [2..4], D(Z) = [3] pas de propagation

- D(Y) = [2..4], D(Z) = [2] propagation D(Y) = [3..4],D(Z) = [2]

Multiplication X = Y * Z

- Si toutes les variables sont positives
  $X \geq \min(D,Y) * \min(D,Z)$, $X \leq \max(D,Y) * \max(D,Z)$ etc. pour Y, Z

- sinon
  $X \geq$ minimum{
  $\min(D,Y) * \min(D,Z)$, $\min(D,Y) * \max(D,Z)$,
  $\max(D,Y) * \min(D,Z)$, $\max(D,Y) * \max(D,Z)$
  }

- similaire pour la borne supérieure pour X en utilisant maximum

Multiplication X = Y * Z

- Règles de propagation pour Y et Z?
- si min(D,Z) < 0 et max(D,Z) > 0 il n' y a pas de restriction pour Y
- On "attend » jusqu'à-ce que le domaine de Z devienne positif ou négatif et ensuite en utilise des règles de la forme

  $Y \geq$ minimum{min(D,X)/min(D,Z), min(D,X)/max(D,Z), max(D,X)/min(D,Z), max(D,X)/max(D,Z)}

  Attention à la division par 0

Consistances généralisés

- On peut combiner plusieurs consistances (noeud, arc, bornes).
- Toutes ses méthodes utilisent les contraintes simples une par une.
- On peut considérer des contraintes complexes qui sont une conjonction de contraintes simples avec un mécanisme de propagation spécial.
- Exemple: all_different({V1,...,Vn})
- All_different ({X , Y , Z }) signifie $X \neq Y \wedge Y \neq Z \wedge X \neq Z$
- Arc-consistant avec D (X) = {1, 2}, D (Y) = {1, 2}, D(Z) = {1,2}
- Mais il n'y a pas de solution.
- Il existe un algorithme spécialisé pour la contrainte all_different [Régin: 1994]