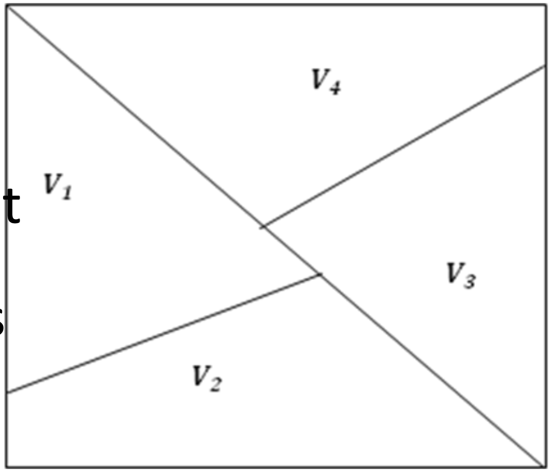
Exercice 1 - AC3

Soit la carte suivante décrivant les frontières entres quatre villes (V_1, …, V_4). On voudrait colorier la carte en utilisant seulement les couleurs rouge, bleu et vert, de sorte que V_1 soit en rouge ou en vert; V_2 et V_3 soient en bleu ou en vert; et V_4 soit en vert. Toutefois, deux villes adjacentes ne peuvent avoir la même couleur.



Donnez le résultat de l'algorithme AC-3 sur ce problème. Que concluez-vous de ce résultat?

Exercice 1 - AC3 (solution)

Variables et domaine des valeurs

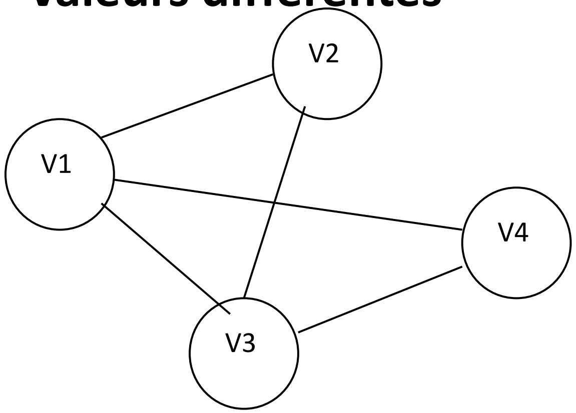
V1 = {r, v}

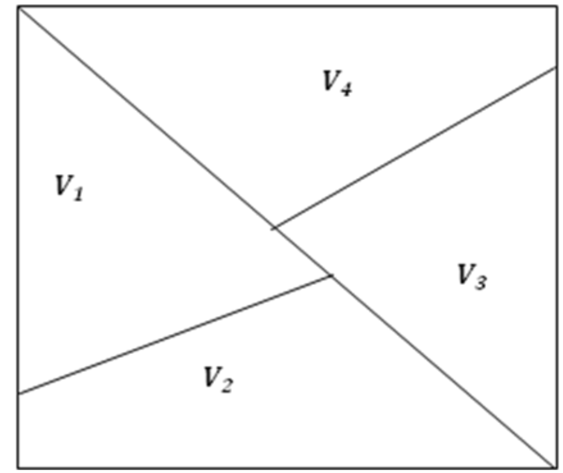
V2 = {b, v}

V3 = {b, v}

V4 = {v}

Graphe de contraintes (lien => valeurs différentes





Queue (ensemble des arcs de contrainte entre variables) =

{ <V1, V1≠V2>, <V1, V1≠V4, <V1, V1≠V3>,
<V2, V1≠V2 >, <V2, V2 ≠V3>,
<V3, V1≠V3>, <V3, V3≠V4>, <V3, V3≠V2>,
<V4, V1≠V4>, <V4, V3≠V4>
}

Exercice 1 - AC3 (solution)

Variables et domaine des valeurs

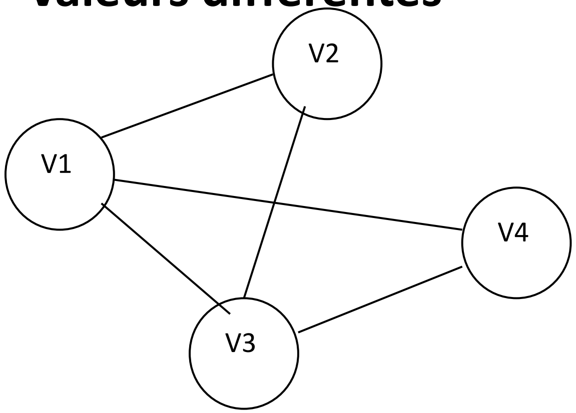
V1 = r ~~{r, v}~~

V2 = {b, v}

V3 = {b, v}

V4 = {v}

Graphe de contraintes (lien => valeurs différentes



<V1, V1≠V2>
<V1, V1≠V4>
<V1, V1≠V3>

Pas de réduction de domaine et Pas d’arcs reliés à V2, V3 et V4 à ajouter puisqu’ils sont déjà dans la liste

Queue (ensemble des arcs de contrainte entre variables) =

{
<V2, V1≠V2 >, <V2, V2 ≠V3>,
<V3, V1≠V3>, <V3, V3≠V4>, <V3, V3≠V2>,
<V4, V1≠V4>, <V4, V3≠V4>
}

Exercice 1 - AC3 (solution)

Variables et domaine des valeurs

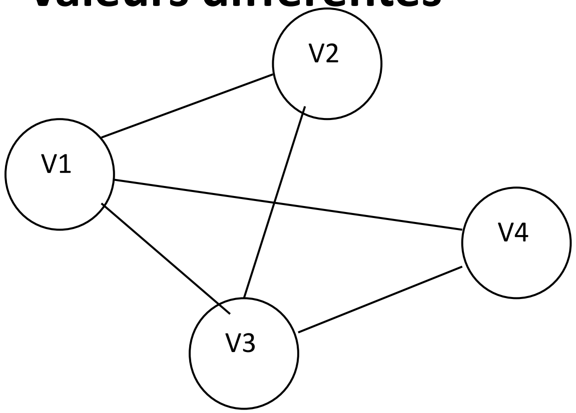
V1 = r ~~{r, v}~~

V2 = b ~~{b, v}~~

V3 = {b, v}

V4 = {v}

Graphe de contraintes (lien => valeurs différentes



<V2, V1≠V2 >

Pas de réduction de domaine et Pas d’arcs reliés à V2, V3 et V4 à ajouter puisqu’ils sont déjà dans la liste

Queue (ensemble des arcs de contrainte entre variables) =

{
<V2, V2 ≠V3>,
<V3, V1≠V3>, <V3, V3≠V4>, <V3, V3≠V2>,
<V4, V1≠V4>, <V4, V3≠V4>
}

Exercice 1 - AC3 (solution)

Variables et domaine des valeurs

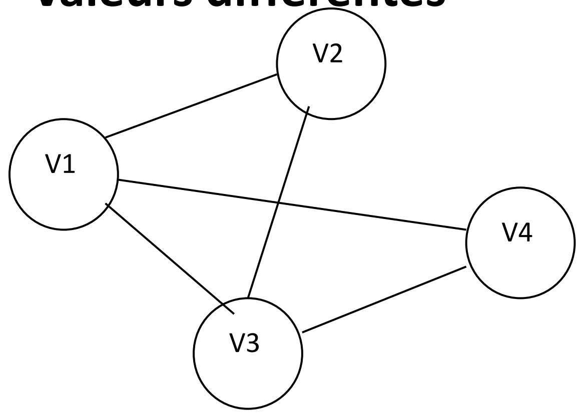
V1 = r ~~{r, v}~~

V2 = b ~~{b, v}~~

V3 = {v}

V4 = {v}

Graphe de contraintes (lien => valeurs différentes



<V2, V2 ≠V3>

Réduction de domaine et ajout de contraintes reliées à V3

Queue (ensemble des arcs de contrainte entre variables) =

{

<V3, V1≠V3>, <V3, V3≠V4>, <V3, V3≠V2>,

<V4, V1≠V4>, <V4, V3≠V4>

<V1, V1≠V3>,

}

Exercice 1 - AC3 (solution)

Variables et domaine des valeurs

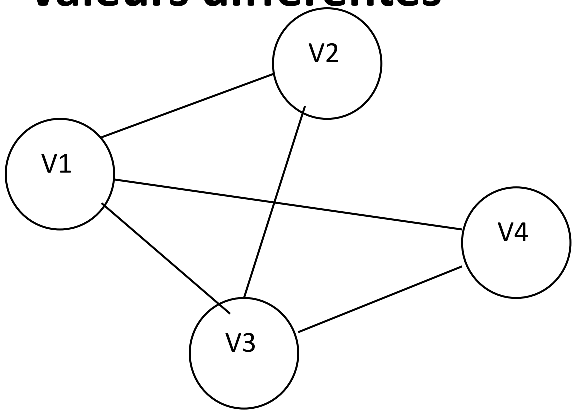
V1 = r ~~{r, v}~~

V2 = b ~~{b, v}~~

V3 = v

V4 = {v}

Graphe de contraintes (lien => valeurs différentes



<V3, V1≠V3>,

Pas de réduction de domaine

Queue (ensemble des arcs de contrainte entre variables) =

{

<V3, V3≠V4>, <V3, V3≠V2>,

<V4, V1≠V4>, <V4, V3≠V4>

<V1, V1≠V3>,

}

Exercice 1 - AC3 (solution)

Variables et domaine des valeurs

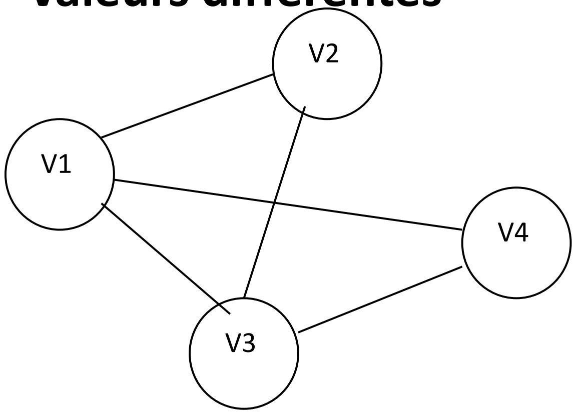
V1 = r ~~{r, v}~~

V2 = b ~~{b, v}~~

V3 = v

V4 = {}

Graphe de contraintes (lien => valeurs différentes



<V3, V3≠V4>

Réduction de domaine

Domaine de V4 vide

Backtrack sur le dernier choix

Queue (ensemble des arcs de contrainte entre variables) =

{
<V3, V3≠V2>,
<V4, V1≠V4>, <V4, V3≠V4>
<V1, V1≠V3>,
}

Exercice 1 - AC3 (solution)

Variables et domaine des valeurs

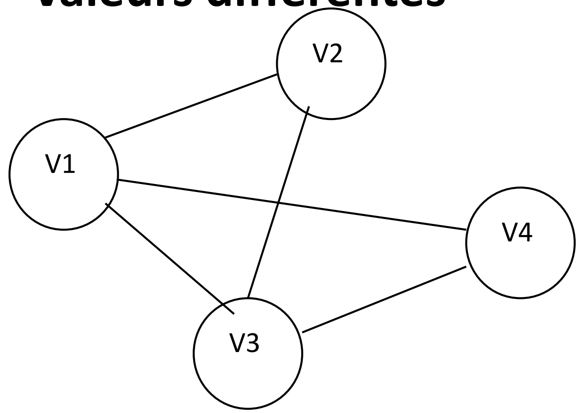
V1 = **r ~~{r, v}~~**

V2 = **v ~~{b, v}~~**

V3 = **{b, v}**

V4 = **{v}**

Graphe de contraintes (lien => valeurs différentes



<V2, V1≠V2 >

Pas de réduction de domaine et Pas d’arcs reliés à V2, V3 et V4 à ajouter puisqu’ils sont déjà dans la liste

Queue (ensemble des arcs de contrainte entre variables) =

{
<V2, V2 ≠V3>,
<V3, V1≠V3>, <V3, V3≠V4>, <V3, V3≠V2>,
<V4, V1≠V4>, <V4, V3≠V4>
}

Exercice 1 - AC3 (solution)

Variables et domaine des valeurs

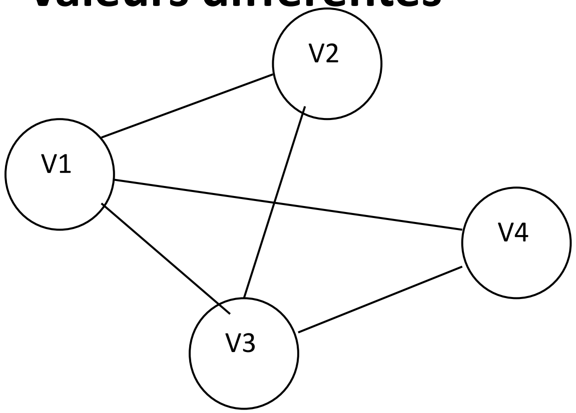
V1 = **r ~~{r, v}~~**

V2 = **v ~~{b, v}~~**

V3 = **{b}**

V4 = **{v}**

Graphe de contraintes (lien => valeurs différentes



<V2, V2 ≠V3>

Réduction de domaine et
Pas d'ajout de contraintes
reliées à V3

Queue (ensemble des arcs de contrainte entre variables) =

{
<V3, V1≠V3>, <V3, V3≠V4>, <V3, V3≠V2>,
<V4, V1≠V4>, <V4, V3≠V4>
}

Exercice 1 - AC3 (solution)

Variables et domaine des valeurs

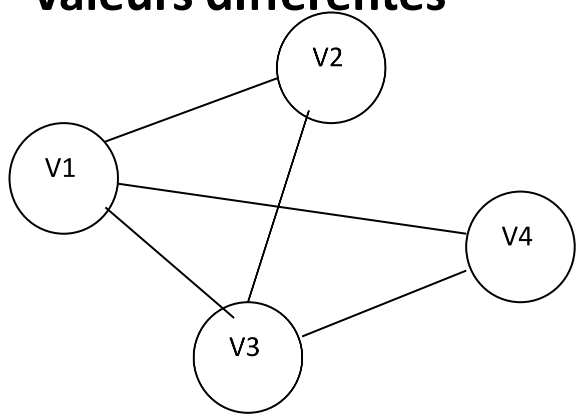
V1 = r ~~{r, v}~~

V2 = v ~~{b, v}~~

V3 = {b}

V4 = {v}

Graphe de contraintes (lien => valeurs différentes



<V3, V1≠V3>
<V3, V3≠V4>
<V3, V3≠V2>,

Pas de réduction de domaine

Queue (ensemble des arcs de contrainte entre variables) =

{
<V4, V1≠V4>, <V4, V3≠V4>
}

Exercice 1 - AC3 (solution)

Variables et domaine des valeurs

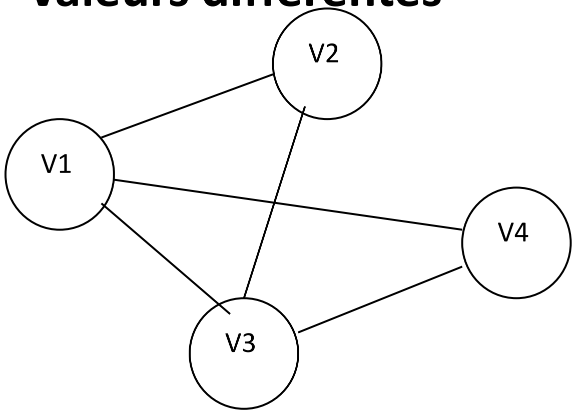
V1 = **r ~~{r, v}~~**

V2 = **v ~~{b, v}~~**

V3 = **b**

V4 = **{v}**

Graphe de contraintes (lien => valeurs différentes



<V4, V1≠V4>
<V4, V3≠V4>

Pas de réduction de domaine

Queue (ensemble des arcs de contrainte entre variables) =
{ }

Exercice 1 - AC3 (solution)

Variables et domaine des valeurs

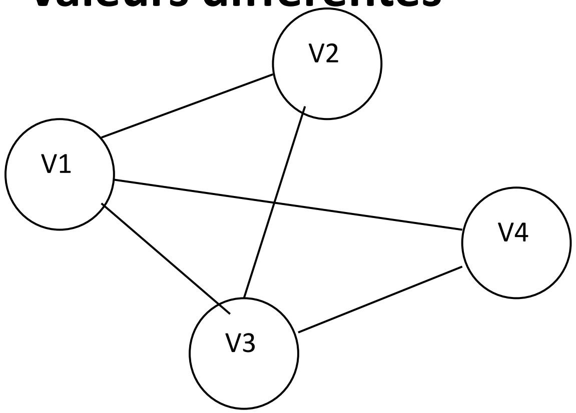
V1 = r ~~{r, v}~~

V2 = v ~~{b, v}~~

V3 = b

V4 = v

Graphe de contraintes (lien => valeurs différentes



Queue (ensemble des arcs de contrainte entre variables) =

{ }

<u>Conclusion:</u>

{V1=r, v2=v, V3=b, V4=v} est donc une assignation complète (toutes les variables sont assignées) et légale (aucune contrainte n'est violée puisqu'il ne reste plus de contrainte à réviser).

C'est donc une Solution

Exercice 1 – Trace simplifiée de AC3

Variables et domaine des valeurs

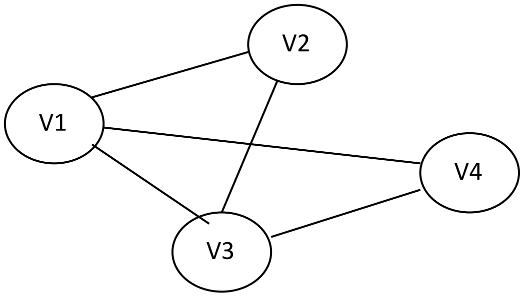
V1 = {r, v}

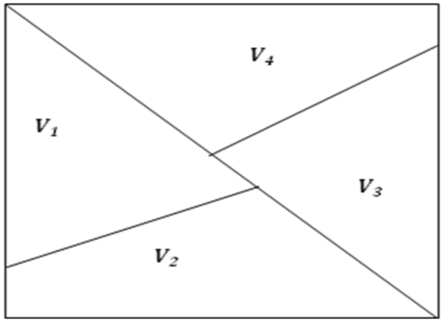
V2 = {b, v}

V3 = {b, v}

V4 = {v}

Graphe de contraintes (lien => valeurs différentes





V1	V2	V3	V4	Contrainte
{r, v}	{b, v}	{b, v}	{v}	
{r}	{b, v}	{b, v}	{v}	V1 --- v4
{r}	{b, v}	{b}	{v}	V3 --- v4
{r}	{v}	{b}	{v}	V2 --- V3

Conclusion :

Le domaine de chaque variable est réduit exactement à une seule valeur. Comme il n'y a que des contraintes binaires, on a une solution.

Exercice 2

Lors de votre party de fin de session, on vous mandate d'être le D.J. Votre mission est de sélectionner les cinq pièces musicales à jouer lors de la soirée. Votre répertoire musical est composé d'un ensemble de dix pièces musicales $\{M_1, \ldots, M_{10}\}$. De ce nombre, six sont en anglais ($M_1, \ldots, M_6$) et quatre en français ($M_7, \ldots, M_{10}$). Les pièces sont classées en styles musicaux : *rock*$=\{M_1, M_2, M_7\}$, *jazz*$=\{M_3, M_8\}$, *techno*$=\{M_4, M_5, M_9\}$ et *alternatif*$=\{M_6, M_{10}\}$. Le comité organisateur vous impose certaines contraintes que vous devez respecter :

- vous ne pouvez pas jouer deux pièces consécutives dans la même langue;
- vous ne pouvez pas jouer deux pièces consécutives du même style de musique;
- vous devez faire jouer au moins une pièce de chaque style;
- vous devez placer une demande spéciale du président de votre association qui veut la pièce M_{10}.
- vous devez terminer la soirée avec une pièce de *jazz*.

a) Indiquez comment modéliser ce problème dans un cadre CSP. Donnez les variables et les contraintes nécessaires.

b) Quel algorithme utiliseriez-vous pour résoudre votre problème? Simulez les étapes de l'algorithme et donnez la solution obtenue.

Exercice 2a (solution)

Variables :

Il faut une variables pour chaque entrée de la sélection : S1, S2, S3, S4 et S5.

Le domaine est l'ensemble des 10 pièces musicales W = {M1 à M10}.

Fonctions :

Langue(M) : retourne la langue d'une pièce

Style(M) : retourne le style d'une pièce

Contraintes :

Contraintes unaires : $S5 \in \{M3, M8\}$ // Jazz (5)

Contraintes binaires :

$Langue(S1) \neq Langue(S2) \neq Langue(S3) \neq Langue(S4) \neq Langue(S5)$ // #1

$Style(S1) \neq Style(S2) \neq Style(S3) \neq Style(S4) \neq Style(S5)$ // #2

Contraintes n-aires

$\cup\, Style(Si) = \{rock, jazz, techno, alternatif\}$ // #3

$\{M10\} \subset \cup\, Si$ // #4

Exercice 2b (solution)

Algorithme : « Backtracking search avec forward checking »

1) utiliser les contraintes unaires pour restreindre les domaines des variables.

2) utiliser les contraintes binaires pour le « forward checking ».

3) lors d'une assignation complète, tester les contraintes n-aires.

-

Stratégies (~ heuristique) pour choisir la prochaine variable + valeurs à assigner:

1) MRV (Most Restrictive VALUE) : commencer par la variable la plus contrainte.

2) allouer M10 aussitôt qu'elle apparaît dans le domaine d'une variable.

3) dans le choix des valeurs, prioriser les styles qui n'ont pas été encore alloués.

Exercice 2b (solution)

Étape \ Var	**S1**	**S2**	**S3**	**S4**	**S5**
Init Domaines	{M1,…, M10}	{M1,…, M10}	{M1,…, M10}	{M1,…, M10}	{M3, M8}
Choix S5 / Heuristique 1	{M1,…, M10}	{M1,…, M10}	{M1,…, M10}	{M1,…, M10}	M3
Forward Checking	M1, M2, M4,M5, M6	M7, M8,M9,M10	M1, M2, M4,M5, M6	M7, M8,M9,M10	M3
Choix S5 / Heuristique 2	M1, M2, M4,M5, M6	M10	M1, M2, M4,M5, M6	M7, M8,M9,M10	M3
Forward Checking	M1, M2, M4,M5, M6	M10	M1, M2, M4,M5, M6	M7, M8,M9	M3
Choix S4 / Heuristique 3	M1, M2, M4,M5, M6	M10	M1, M2, M4,M5, M6	M7	M3
Choix S1 / Heuristique 3	M4	M10	M1, M2, M4,M5, M6	M7	M3
Forward Checking	M4	M10	M1, M2,M5, M6	M7	M3
Choix S3 / (arbitraire)	M4	M10	M1	M7	M3