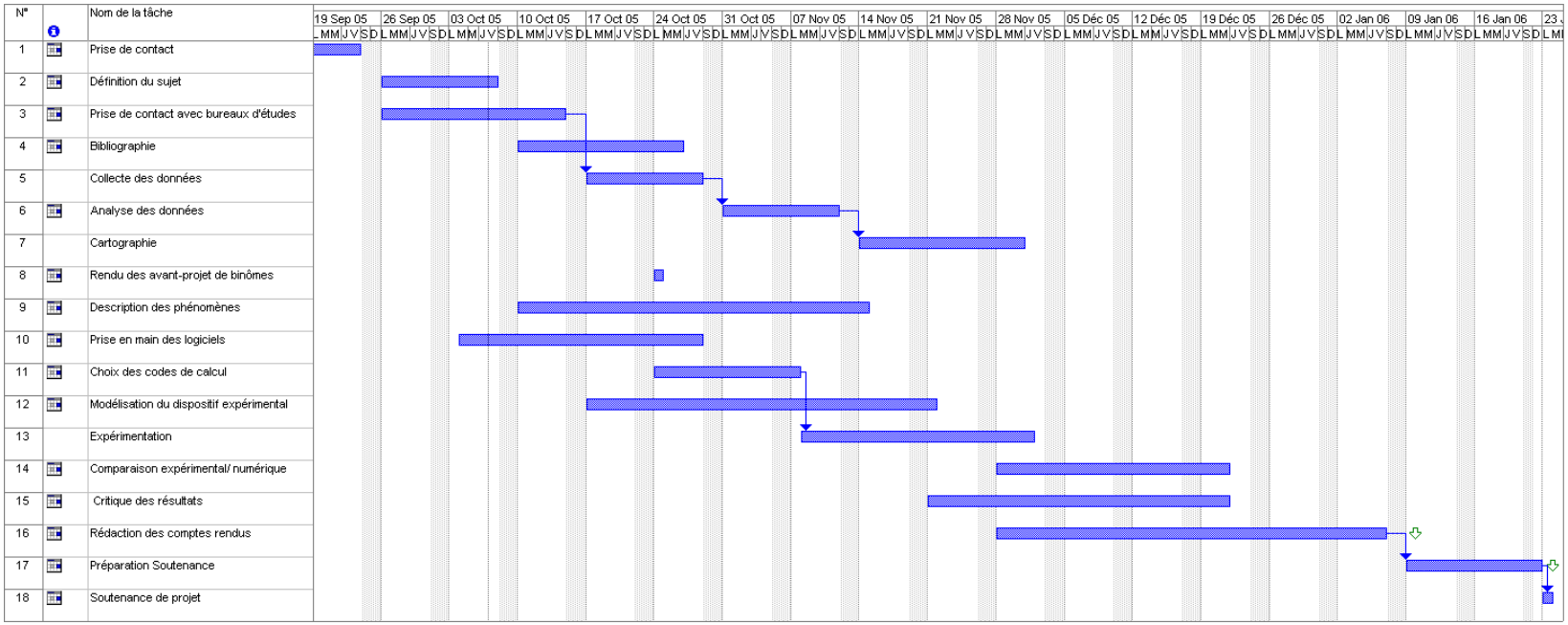
Allocation de ressources et programmation par contraintes

Master 1 MIAGE « Ingénierie Métier »

Raisonnement et Science de la Décision

Laurent.Perrussel@ut-capitole.fr

Planification de tâches



Allocation de ressources

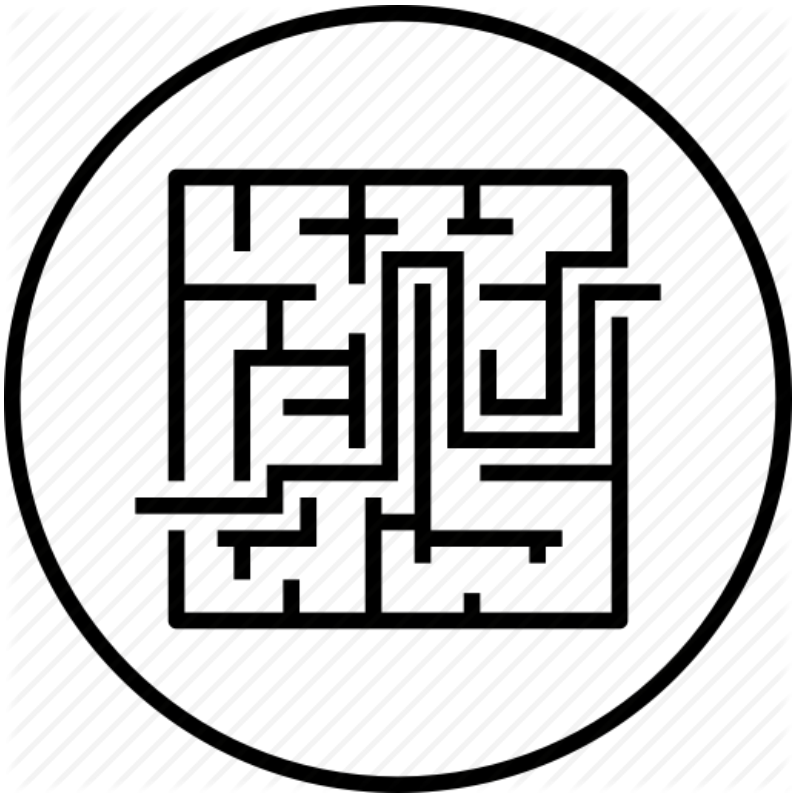
- Contraintes sur les disponibilités des ressources
  - Personnel
  - Machine
  - Salle
- Contraintes sur les dates de réalisation des tâches
  - Dates de début
  - Dates de fin

Allocation de ressources

- Objectif
  - Affecter des ressources à des tâches.
  - Organiser les activités
    - Dimension temporelle
  - Affectation contrainte
    - Ressources non universelles
    - Disponibilité limitée des ressources
- Données en entrée
  - Ressources
  - Tâches
  - Contraintes
- Données en sortie
  - Planification temporelle

Planification

- Trouver les tâches et leur ordonnancement pour satisfaire un objectif
- Données
  - Entrée
    - Tâches possibles
    - Contraintes
    - But
  - Sortie
    - Séquence de tâches



Organisation

- Programmation par contraintes
  - Définition
  - Rechercher une solution
- Représenter un problème d'allocation
  - Description des ressources et tâches
  - Calcul de la solution
- Exemple d'allocation
  - Atelier
  - Planning d'employés

Programmation par contraintes

- Problème
  - Variables
  - Domaines possibles de valeurs pour les variables
  - Contraintes sur les variables et les domaines
    - Combinaison de valeurs possibles pour les variables
    - Forme potentiellement complexe

- Exemple : formule « plat dessert »
  - Variables
    - Plat
    - Dessert
  - Domaines
    - Plat ∈ {steakHaché, paella}
    - Dessert ∈ {yaourt, pâtisserie}
  - Contraintes
    - (steakHaché, yaourt), (paella, yaourt), (steakHaché, patisserie)

Programmation par contraintes

- Affectation
  - Une valeur pour chaque variable
    - Affectation totale
  - Une valeur pour quelques variables
    - Affectation partielle
- Solution
  - Une affectation satisfaisant toutes les contraintes
- Optimisation possible
  - Fonction d'optimisation portant sur les variables
    - Grandeur numérique
  - Minimiser ou maximiser

Programmation par contraintes

- Exemple
  - Variables et domaines
    - $A \in \{0,1\}, B \in \{1, 2\}$
  - Contraintes
    - $A \neq B$
  - Affectation
    - (0,1), (1, 1), (0, 2) et (1, 2)
  - Affectations « solution »
    - (0,1), ~~(1, 1)~~, (0, 2) et (1, 2)
  - Optimisation
    - F : A + B à minimiser
    - Solution optimale
      - A = 0 et B =1

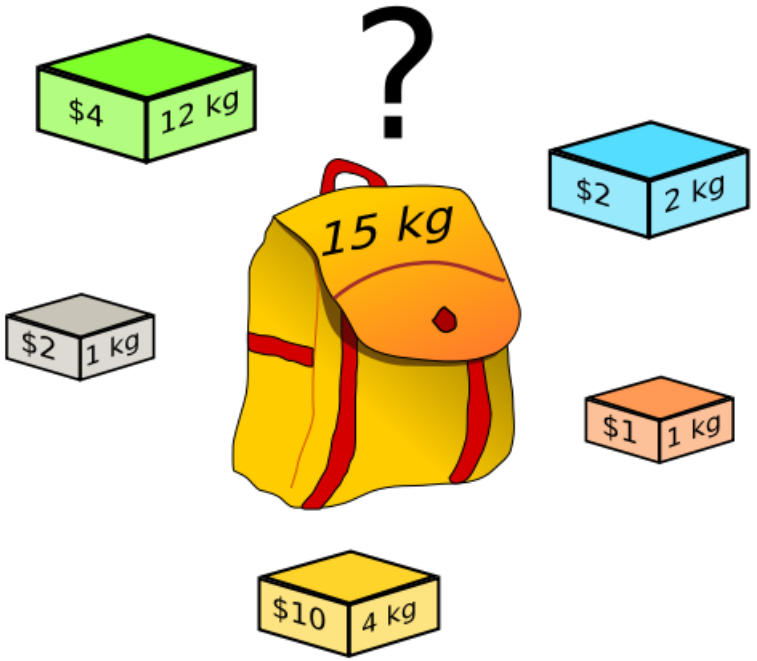
Programmation par contrainte : plusieurs technologies

- Programmation linéaire
  - Contraintes arithmétiques
    - Ex : $A + B < 10$
  - Opérations arithmétiques
    - Somme
      - $C = A + B$
    - Produit scalaire
      - $C = [X1..Xn] . [Y1..Yn]$
- Contraintes générales
  - Valeurs différentes
  - Présence d'une valeur dans une liste de variables
- Programmation logique
  - Combinaison logiques de contraintes
  - Ex :
  - $(A \in \{0,1\}$ ET $B \in \{1, 2\})$ OU $(A \in \{2,4\}, B \in \{5,6\})$

Programmation par contraintes : problème du sac à dos

- **Variables**
  - **Poids maximal :** Max,
  - **Poids :** $P1 \ldots PN$,
  - **Valeur :** $V1 \ldots VN$,
  - **Présent :** $X1 \ldots XN$
- **Domaines**
  - Pi et Vi **fixés**
  - $Xi \in \{0,1\}$
- **Contraintes:**
  $[P1 \ldots PN] \,.\, [X1 \ldots XN] \leq Max$
- **Optimisation**
  $F = [V1 \ldots VN] \,.\, [X1 \ldots XN]$

  maximiser(F)



Programmations par contraintes : rechercher les solutions

- Recherche « naïve »
  - Tout vérifier
- Inefficace et très couteux
  - 2^n affectations pour n variables (binaires)

Solution = {}

Pour chaque affectation totale A

  Si A satisfait toutes les contraintes

    Solution ← solution ∪ A

  Finsi

FinPour

Programmation par contraintes rechercher les solutions

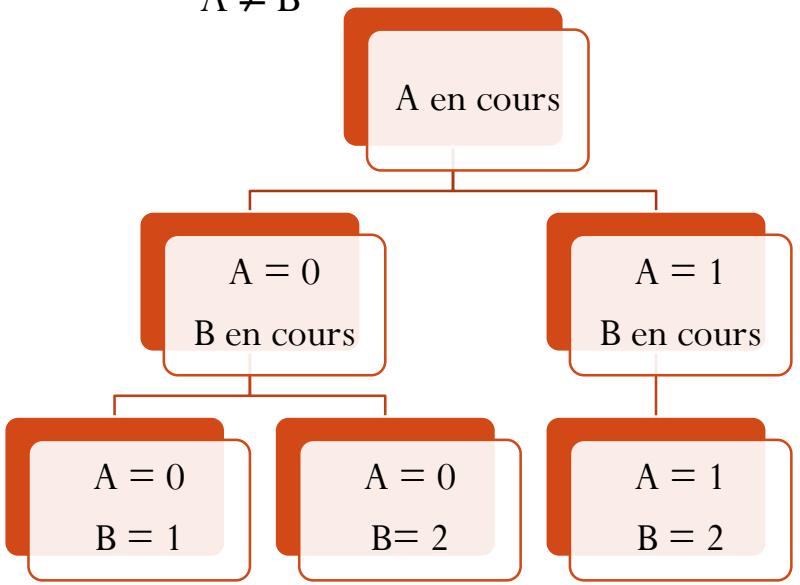
- Profondeur d'abord
  - Affectation partielle
  - Compléter l'affectation en vérifiant les contraintes à chaque étape
- Variables
  - Déjà affectées
  - En cours
  - Futures

Variables et domaines

$A \in \{0,1\}, B \in \{1, 2\}$

Contraintes

$A \neq B$



Programmation par contraintes rechercher les solutions

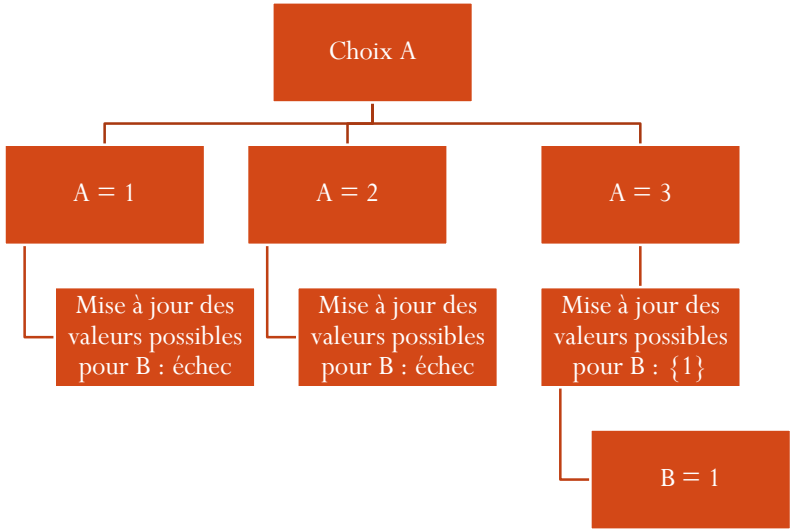
- Profondeur d'abord
  - Arbre de solutions partielles
  - Allers et retours dans l'arbre de solutions
    - « Forward » : affectation est complétée et explorée
    - « backward » : affectation est partiellement annulée

- Exploration « forward »

Choisir une variable X dans les variables futures
Futures ← futures - X
Pour toutes les valeurs possibles de X
Mettre à jour les valeurs possibles des variables futures
Si valeurs cohérentes existent
conserver affectation partielle
explorer (futures)
Finsi
FinPour

Programmation par contraintes rechercher les solutions

- Exemple :
  - 2 variables A et B
  - Domaine : 1..3 pour A et B
  - Contrainte : A = 3*B



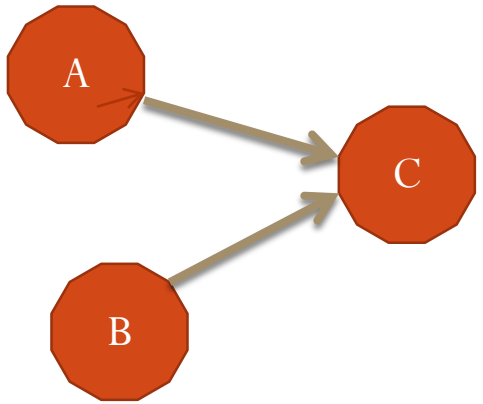
Représenter un problème d'allocation

- Objectif
  - Trouver une allocation optimale pour des ressources et des tâches
  - Contraintes
    - Chaque tâche a une durée
    - Relation de précédence entre tâches
  - Hypothèse
    - Tâche non interruptible
- Exemple
  - Planification de projet
  - Affectation de travaux à une machine
  - Planification employés
    - Quelles personnes dans quels services
  - Organisation d'un championnat sportif
  - Embarquement / débarquement dans un aéroport
  - …

Allocation : décrire les tâches

- Fonctions associées à une tâche X
  - Durée(X)
  - dateDebutPlusTot(X)
  - dateFinPlusTard(X)
  - TachesPrecedentes(X)

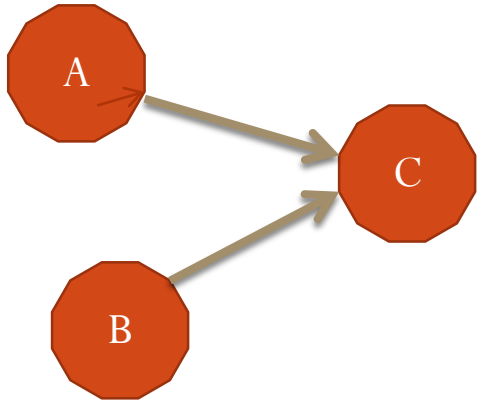
Tache	Durée	Tot	Tard
A	2	0	10
B	1	1	2
C	3	4	7



Allocation : décrire les contraintes

- Début des taches
  - Debut(X) ∈ [dateDebutPlusTot(X), dateDebutPlusTard(X)]
  - dateDebutPlusTard(X) = dateFinPlusTard(X) – duree(X)
- Précédence : X < Y
  - Debut(X) + duree(X) ≤ Debut(Y)

Tache	Durée	Tot	Tard
A	2	0	10
B	1	1	2
C	3	4	7



debut(A) ∈[0..8]
Debut(C) ∈[4..4]
Debut(A) + duree(A) ≤ Debut(C)

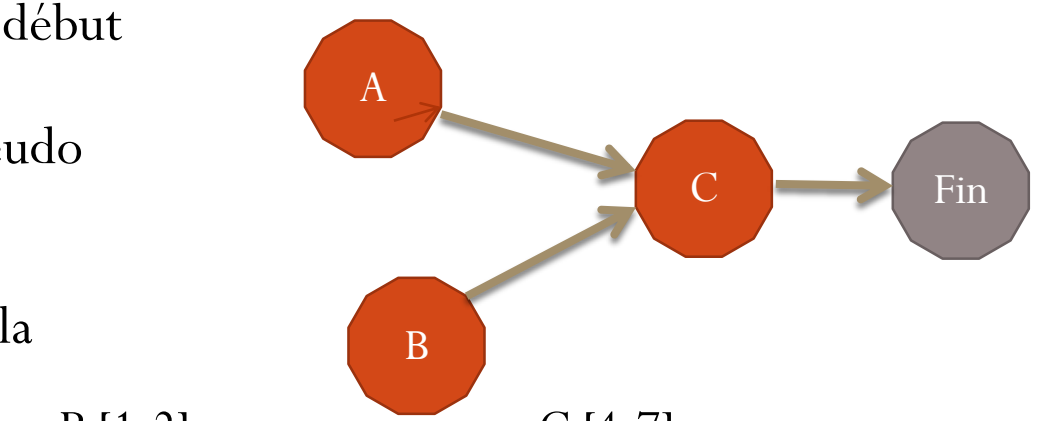
Allocation : optimisation

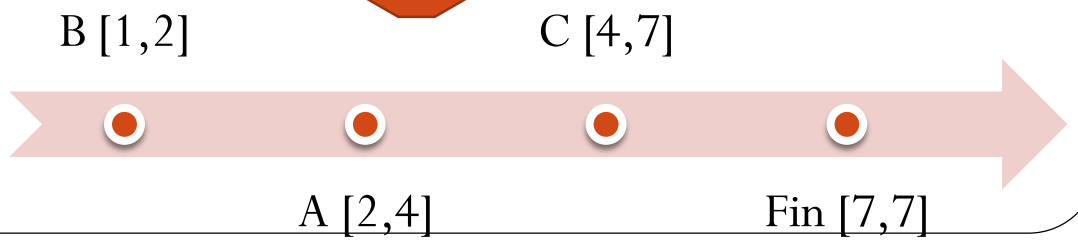
- Hypothèse : ressource unitaire
  - Contraintes d'unicité
- Précédences libres :
  - X <Y OU Y < X
- Solution : planification avec le plus court délai
  - Minimiser les dates de début des dernières tâches
  - Introduction d'une pseudo tache « agrégeant » les dernières taches
  - Minimiser le début de la pseudo-tache

- Pseudo-tache « Fin » après C

  Debut(C) + duree(C) ≤ Debut(Fin)

  Debut(C) + 3 ≤ Debut(Fin)

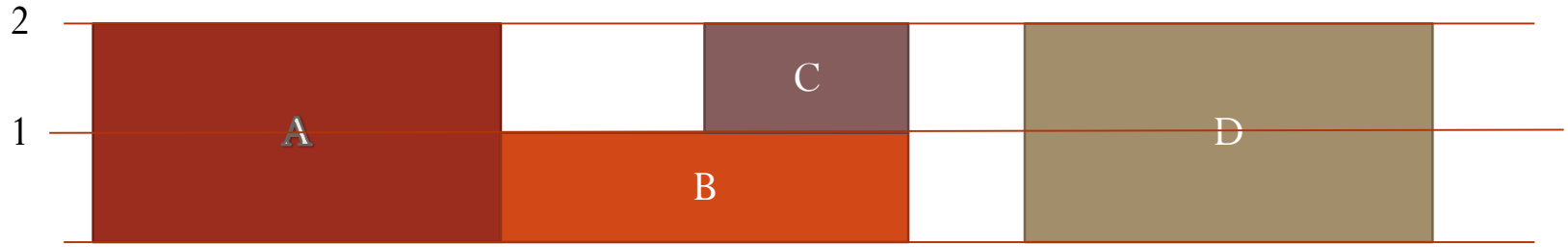




Allocation : ressources multiples

- Machines en différents exemplaires

- Permet de faire des tâches en parallèle
  - Ajout de règles avec des capacités



Allocation : exemples

- Atelier Tâches – Machine (Job-shop)
  - Chaque tâche est décomposée en opérations
  - Opérations sont ordonnées
  - Opérations ont une durée
  - Opérations à effectuer sur machines dédiées

- Objectif
  - Minimiser le temps passé (autrement dit le temps perdu)

Allocation : atelier

- 3 machines
  - M1, M2 et M3
- 3 tâches
  - T1 : opérations sur M3 (durée : 4) puis M2 (2) et M1 (1)
  - T2 : opérations sur M1 (3) et M3 (3)
  - T3 : opérations sur M2 (2), puis M1 (4) et M3 (1)

- Variables
  - 1 variable par opération sur une machine : OpI,J
    - Opération de la tâche I sur la machine J

Allocation : atelier

- Variables
  - 1 variable par opération de la tache I sur une machine J : OpI,J
  - 1 variable Fin
- Contraintes de précédence
  - Debut(Op1,3) + duree(Op1,3) ≤ debut(Op1, 2)
  - Debut(Op1,2) + duree(Op1,2)) ≤ debut(Op1, 1)
  - …
- Optimisation
  - Debut(Op1,1) + 1 ≤ debut(Fin) ET Debut(Op2,3) + 3 ≤ debut(Fin) ET Debut(Op3,3) + 1 ≤ debut(Fin)
  - Minimiser Debut(Fin)

Allocation : planning d'employés

- Planifier l'activité d'employés pour une semaine
  - Type de service : matin, après-midi, nuit
  - Nombre d'employés requis par type de service
    - Valeur minimale et maximale
  - Nombre de services par employés
    - Valeur minimale et maximale
  - Cout par employé et service

- Objectif
  - Minimiser le cout

Allocation : planning d'employés

- Employés : Alice, Bob, Charlie…
- 7 jours à planifier
- Service: Matin (AM), Après-midi (PM) et nuit (N)
- Présence par service
  - AM: 2-3, PM : 3 et N : 1-2
- Activités possibles par employé
  - AM: 2-3, PM : 1-2 et N : 1-2
- Activités par employé : 4-6 services
- Cout
  - AM : 10, PM : 12 et N : 15

Allocation : planning d'employés

Résultat visé

	Lundi	Mardi	Mercredi	…
Alice	AM	PM	PM	
Bob	AM	-	AM	
Charlie	PM	N	N	
…				

Allocation : planning d'employés

- Représentation
  - Variables pour chaque jour Y où l'employé X travaille TrXY
    - TrAliceLundi : Alice travaille Lundi
  - Domaine pour TrXY : AM, PM, N et Libre
- Variables pour min et max employés par service
  - NbEmpAM ∈[2..3] (nombre d'employés pour le matin)
- Variables pour min et max de services par employés
  - NbServN ∈[1..2] (nombre de nuits possibles)

Allocation : planning d'employés

- Contraintes
  - Somme des NbEmpAM, NbEmpPM, NbEmpN (pour tous les jours de la période)
  - Calculer le cout global sur la période
- Minimiser ensuite le cout.