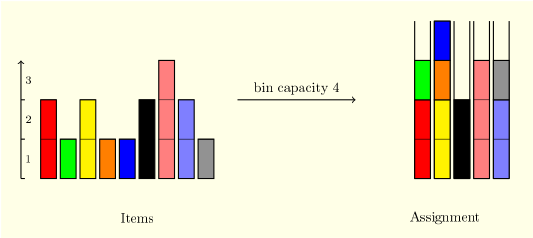
Bin-Packing

# Problème de bin packing :

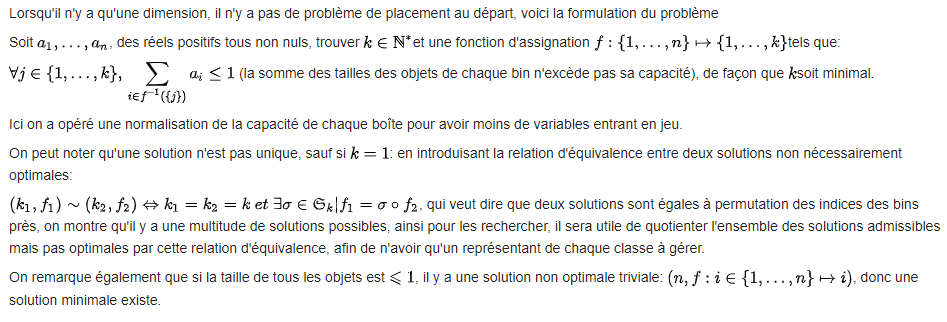
Le problème de bin packing consiste à déterminer le nombre minimum de conteneurs (bins) identiques nécessaires pour ranger un ensemble d’objets. Il a fait l’objet depuis quelques années d’une attention croissante pour deux raisons : outre son intérêt théorique, il a de nombreuses applications dans le domaine industriel, de la logistique, de l’informatique et de l’édition. On retrouve le problème notamment dans l’industrie du tissu, du métal, mais aussi dans le cadre de la découpe de bois, de verre ou même de placement des spots publicitaires dans les journaux



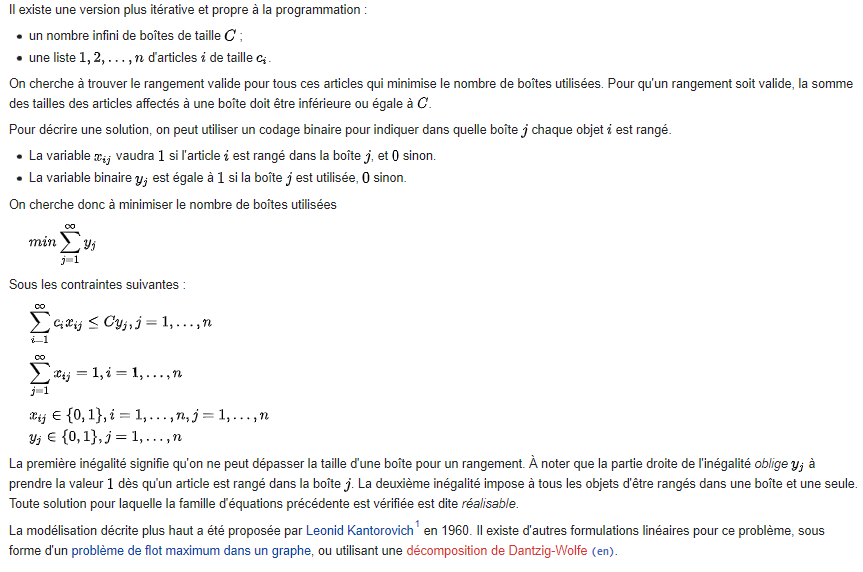
# Formulation du problème:

Dans sa forme générale le Bin-Packing est un problème d'optimisation, on a une population d'objet à ranger avec des contraintes dans le moins de boîtes identiques (appelées bin) possibles

## En une dimension :



## Formulation itérative :



## Différentes variantes :

## Domaines d’applications :

Il a de nombreuses applications dans le domaine industriel, de la logistique, de l’informatique et de l’édition. On retrouve le problème notamment dans l’industrie du tissu, du métal, mais aussi dans le cadre de la découpe de bois, de verre ou même de placement des spots publicitaires dans les journaux

## Exemple de résolution du problème :

nous disposons de :  
X medias (cd) de taille identique T  
N fichiers de tailles differents (tailles contenu ds un tableau C)  
  
Nous cherchons à minimiser le nombre de Cd utilisés, pour cela j'ai opté pour trier les medias et les cd par ordre decroissant puis d'inserer le premier fichier (donc le plus gros) ds le premier CD ou il y a assez de place pour le stocker.  
Les medias sont soit ouvert (on enregistre des choses dessus), soit fermé (ils sont plein) soit rien (pas encore utilisé).  
  
Nous faisons donc une Guerre eclair (permettant de trouver vite une solution ), nous construisons un arbre puis nous faisons une Guerre de front qui a partir de l arbre construit permettra de trouver LA solution. Je ne sais pas si tu est familier avec ces notions "Guerre Eclair" et "Guerre de Front", si c'est le cas cool sinon renvoi moi un message et je te repondrai plus precisément.  
  
Notre "séparation", pour la creation de l arbre (savoir ce qu'on met ds le fils Gauche et ds le fils droit), est "trouver un media suffisament grand pour le prochain fichier a mettre (en regardant en priorité ds les médias deja ouvert)" et notre separation .. bne justement on en a pas.. (rappel : sepration : ce qui sera contenu ds les noeuds de l arbre et qui permettra de savoir si nous allons ds le fils gauche ou droit lors des différentes guerres, il s agit en fait d'une formule mathématique -à trouver- mais qui devrait ressembler à :  
{  
nb de media ferme +  
nb de media ouvert +  
[  
(taille des fichiers restant apres insertion du fichier en cours)-  
(la taille dispo sur les medias ouverts)  
]  
/ (taille initiale du cd)  
}

## Méthodes de résolution et algorithmes proposés :

Le problème de bin packing a été largement étudié dans la communauté de recherche opérationnelle. Il existe des [heuristiques](https://fr.wikipedia.org/wiki/Heuristique_(math%C3%A9matiques)" \o "Heuristique (mathématiques)) très efficaces pour le résoudre, et une modélisation très efficace utilisant l'[optimisation linéaire](https://fr.wikipedia.org/wiki/Optimisation_lin%C3%A9aire" \o "Optimisation linéaire).

### Méthode heuristique :

Pour résoudre le problème de bin packing, on utilise souvent des algorithmes simples comme first-fit decreasing (FFD) ou best-fit decreasing (BFD). Les deux méthodes fonctionnent suivant un principe similaire : on trie la liste d'articles par ordre décroissant de taille, puis on range chaque article dans l'ordre. Dans first-fit, on range l'article courant dans la première boîte qui peut le contenir. Dans best-fit, on range l'article dans la boîte la mieux remplie qui puisse le contenir. Ces algorithmes ne sont pas optimaux, mais ils permettent d'obtenir de très bons résultats en pratique.

Les algorithmes Best Fit Decreasing et First Fit Decreasing n'utilisent jamais plus de 11/9 OPT + 1 boîtes (où OPT est le nombre optimal de boîtes dans une solution optimale)[2](https://fr.wikipedia.org/wiki/Probl%C3%A8me_de_bin_packing#cite_note-2). La procédure de tri est la partie la plus coûteuse de l'algorithme, mais sans elle, la qualité de la méthode est beaucoup moins bonne. On obtient dans ce cas des solutions utilisant au pire 17/10 OPT.

Une version plus efficace de FFD utilise au plus 71/60 OPT + 1 boîtes.

### Méthodes exactes :