Utilisation de Bob++

Pascal Vander-Swalmen

Université de Versailles Saint-Quentin-en-Yvelines
ANR HORUS

28 juin 2011

Plan

Plan

Introduction

But de la présentation

- Support afin d'aider à la prise en main de l'outil « bob++ »
- Adapté au projet HORUS
- Ne remplace pas l'aide en ligne (`http://bobpp.prism.uvsq.fr/`) mais offre un support visuel
- Sera mis à disposition des acteurs du projet

Situation de bob++ dans le spectre des solveurs

- Framework de développement de programmes parallèles
- Orienté Optimisation Combinatoire exacte
- Écriture de fonctions clés
  => parallélisation automatique et transparente pour l'utilisateur
  (quelque soit la technologie employée – tant qu'elle est implémentée –)

Plan

Rechercher une solution à un problème d'OC

Principe systématique

- Parcourir l'espace de solution
- Identifier une solution
- Évaluer la solution
- Selon l'évaluation, conservation en mémoire ou non

Résolution exacte

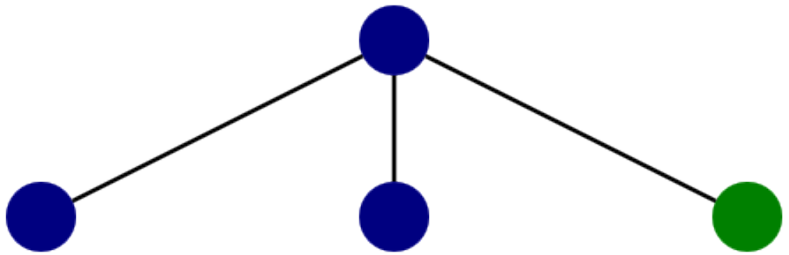
- Parcours d'un arbre de recherche
- Descente dans l'arbre en simplifiant le problème ...
- ...jusqu'à ce qu'une solution puisse être évaluée

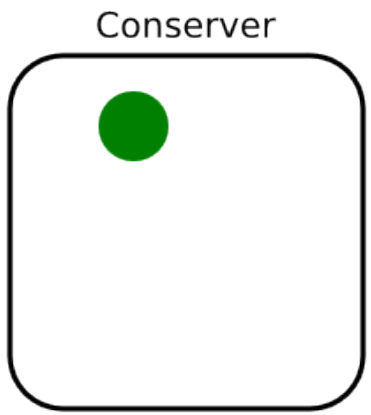
Rechercher une solution à un problème d'OC

Schéma typique de ce genre d'algorithme

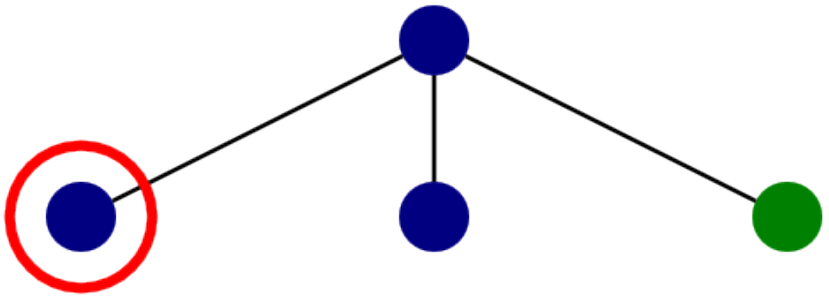
- Tant que l'objectif n'est pas atteint, faire
  - Sélectionner un nœud de l'arbre
  - Générer ses fils
  - Les évaluer
  - Pour chaque fils, selon le résultat par rapport à l'objectif
    - Supprimer le nœud
    - Insérer le nœud dans l'arbre (continuer la branche)
    - Marquer comme tout ou partie de l'objectif
- Fin Tant que

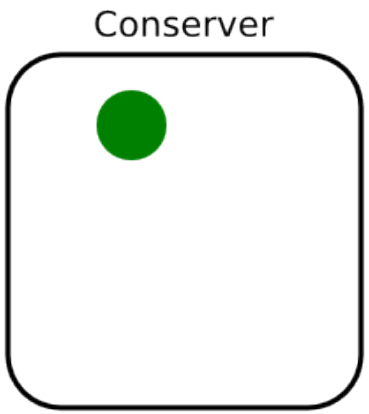
Rechercher une solution à un problème d'OC - schéma normal



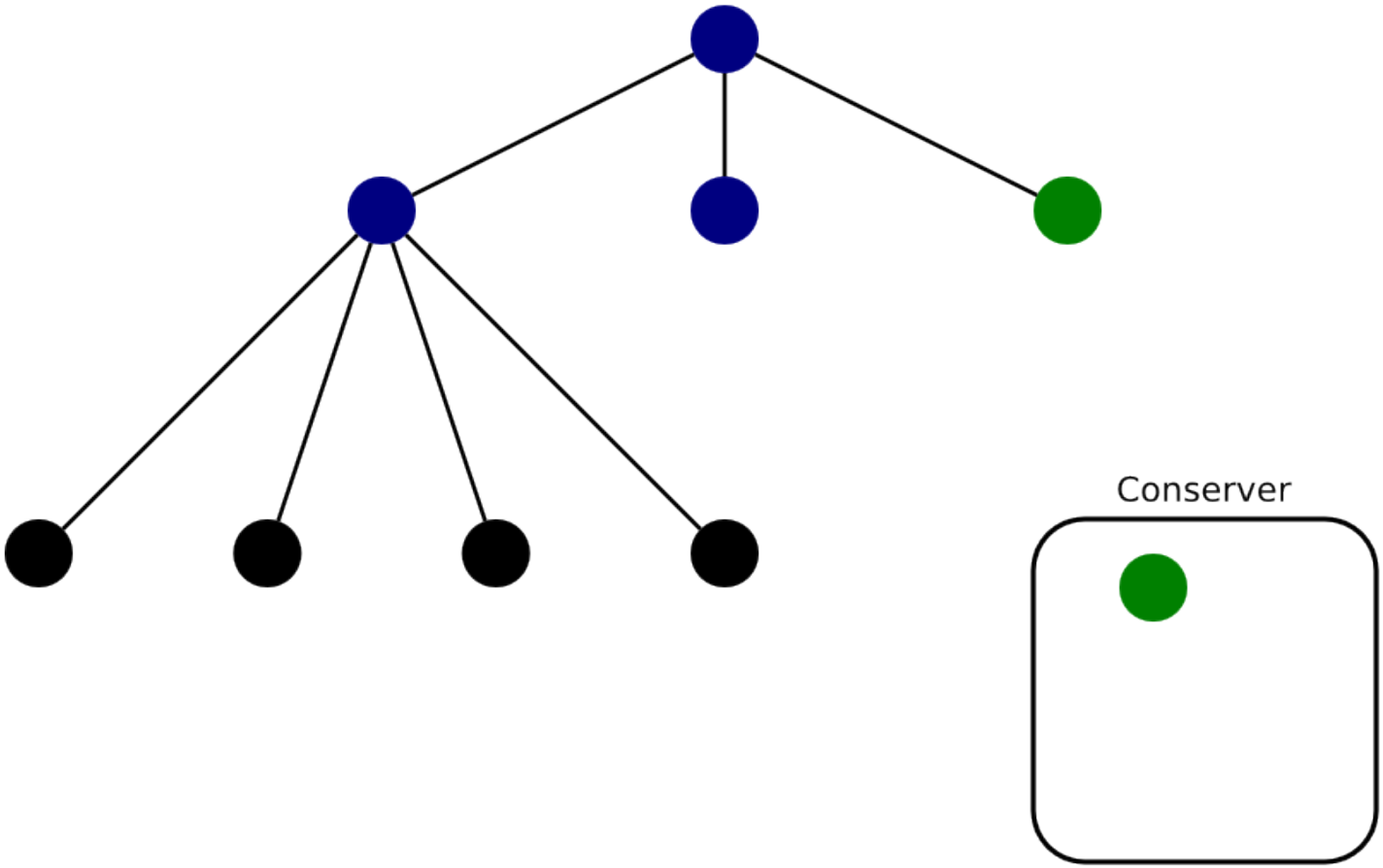


Rechercher une solution à un problème d'OC - schéma normal

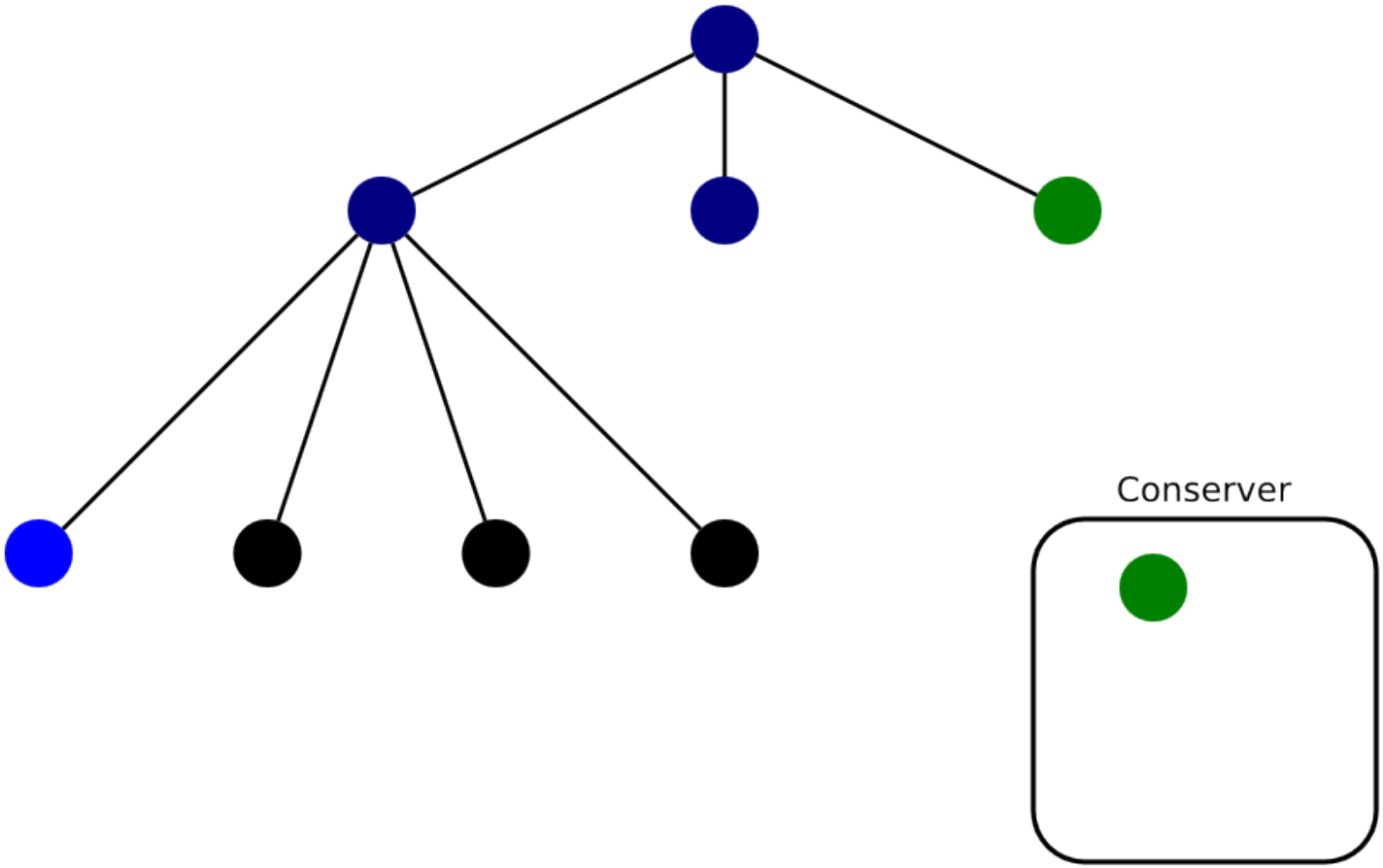




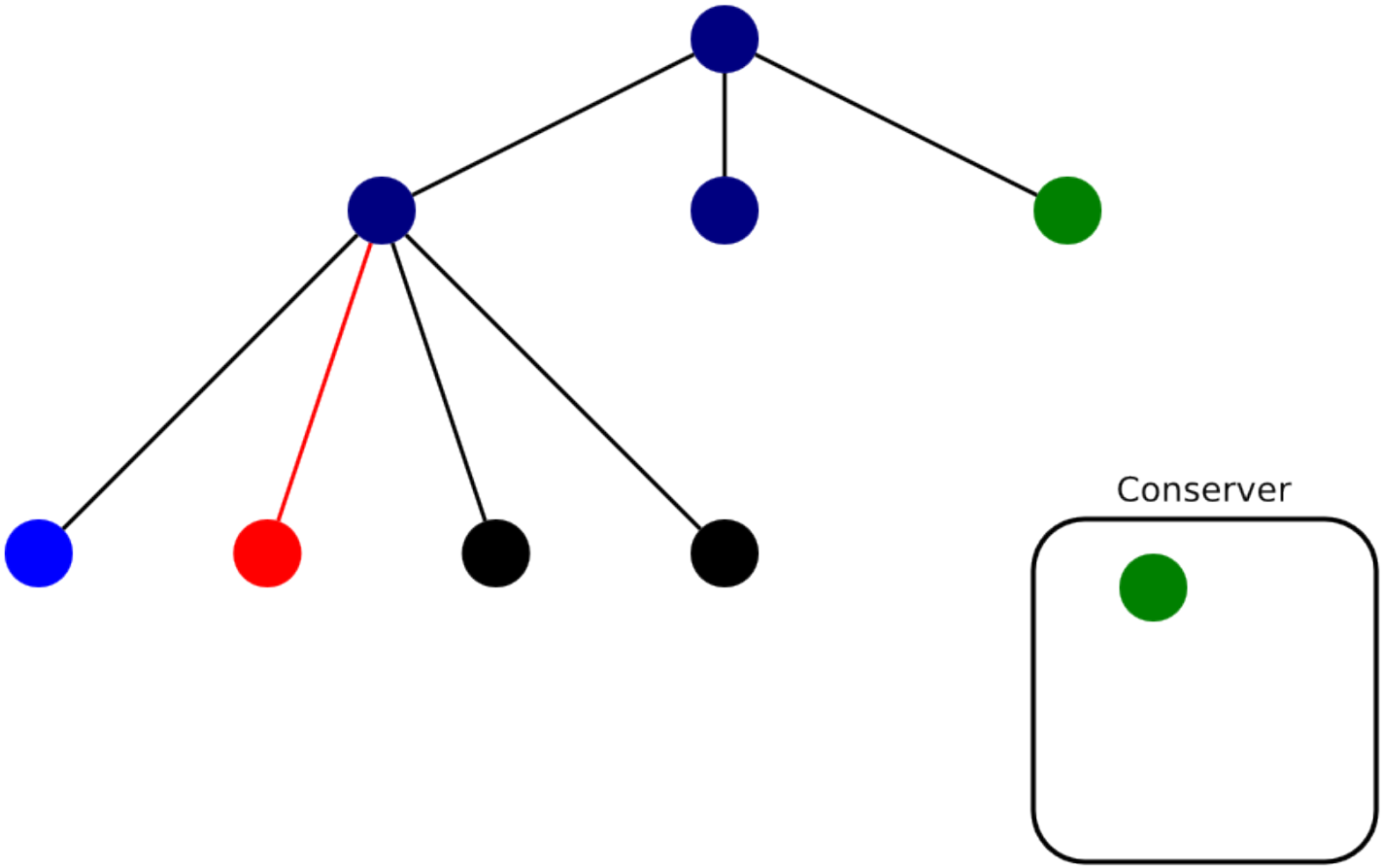
Rechercher une solution à un problème d'OC - schéma normal



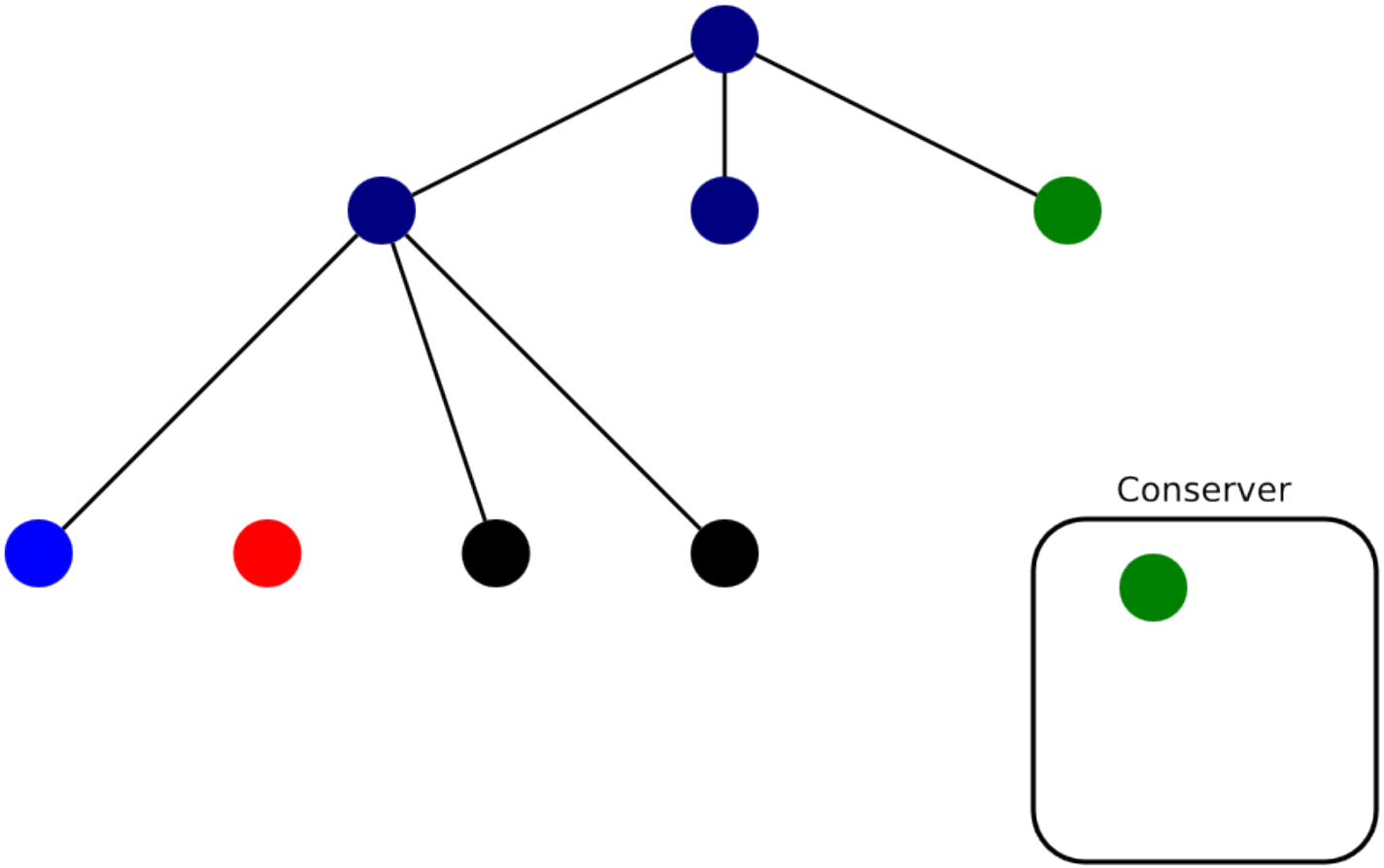
Rechercher une solution à un problème d'OC - schéma normal



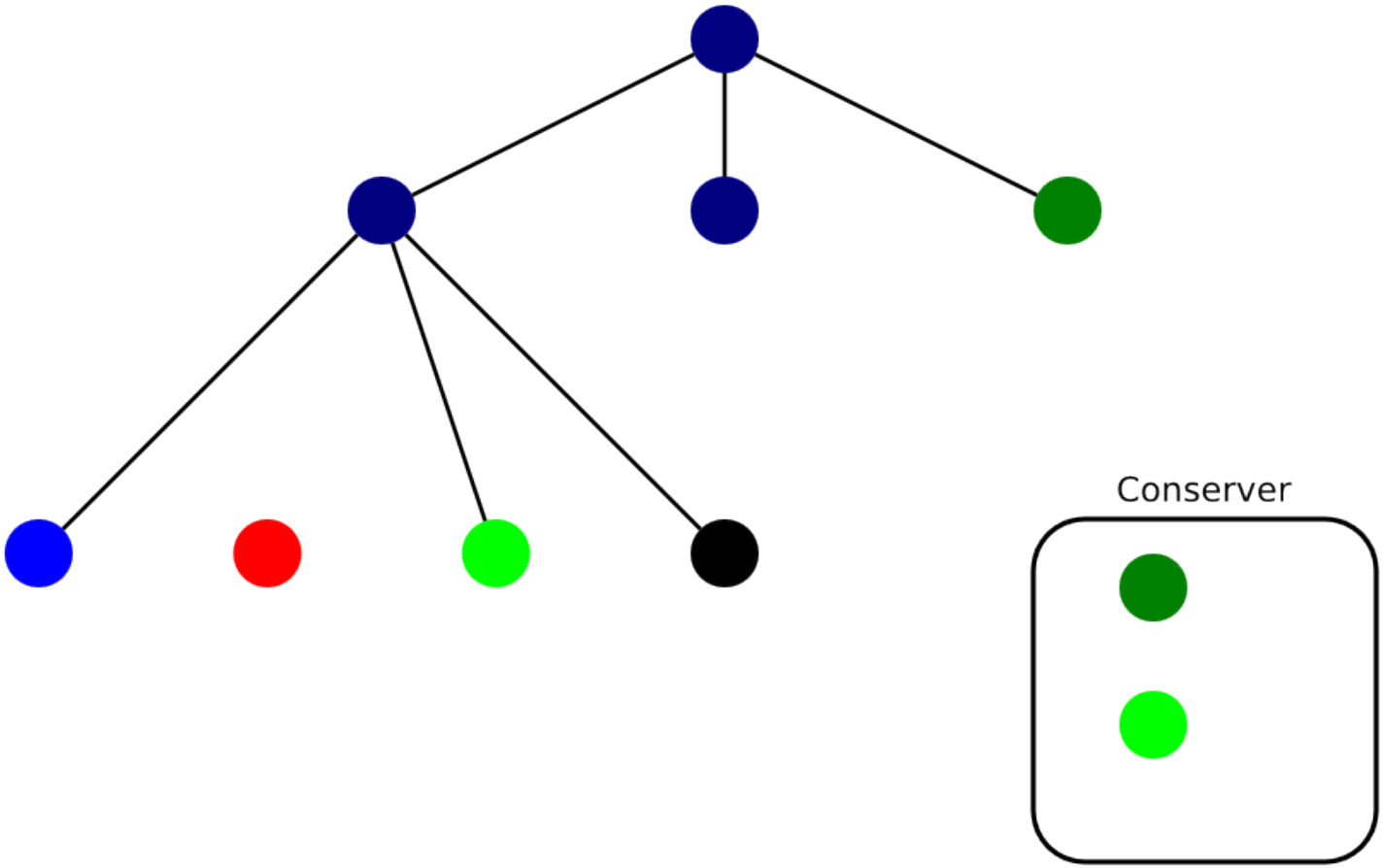
Rechercher une solution à un problème d'OC - schéma normal



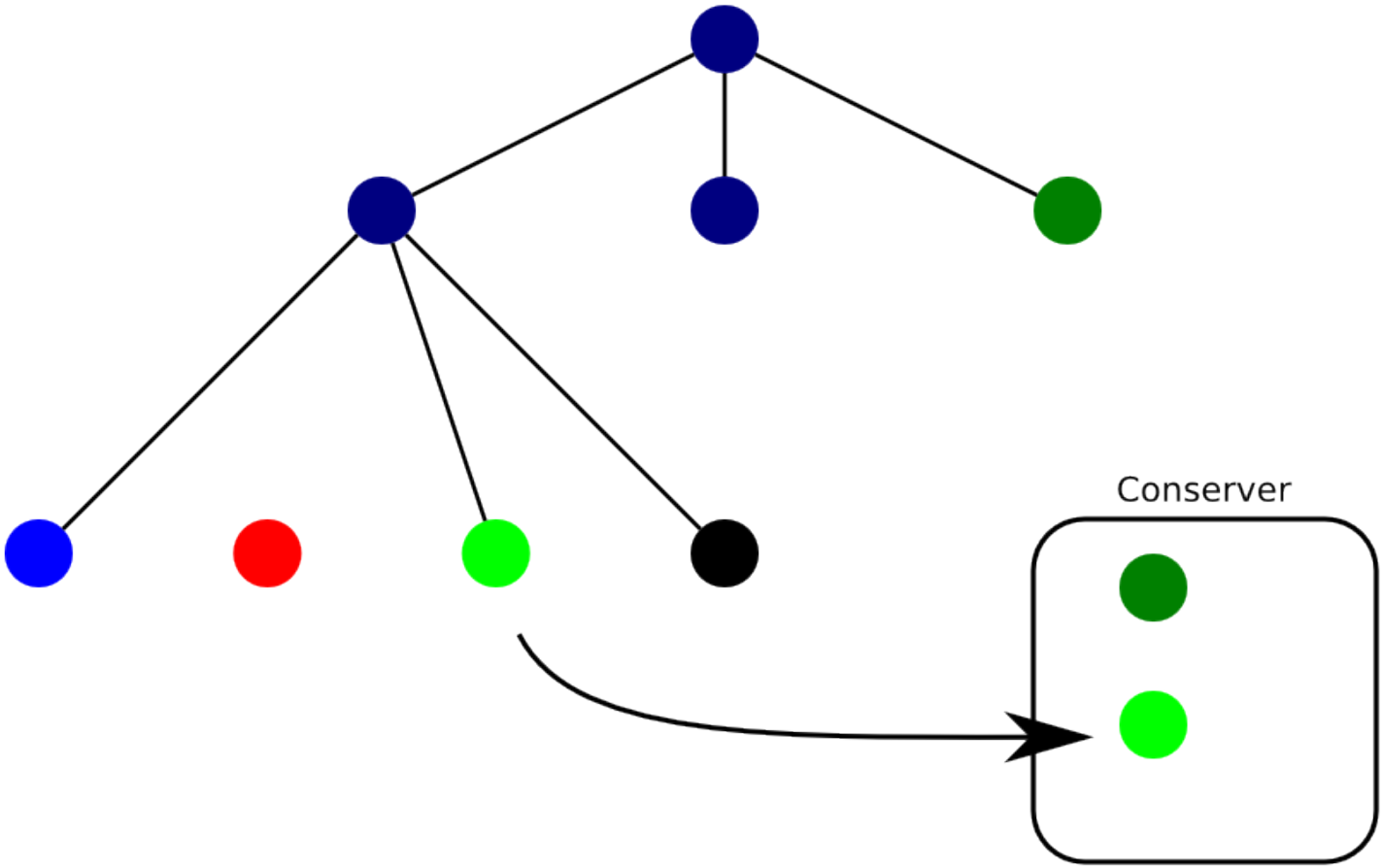
Rechercher une solution à un problème d'OC - schéma normal



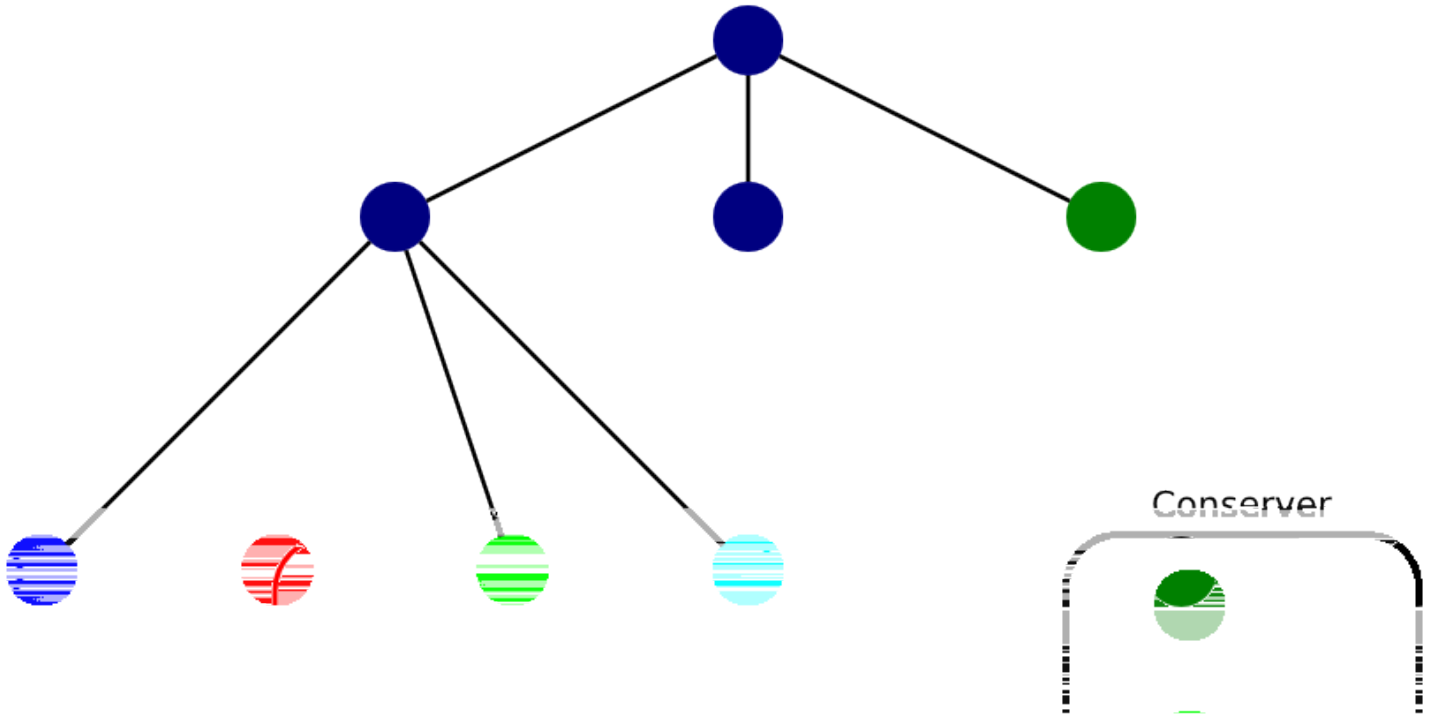
Rechercher une solution à un problème d'OC - schéma normal



Rechercher une solution à un problème d'OC - schéma normal



Rechercher une solution à un problème d'OC - schéma normal



Rechercher une solution à un problème d'OC

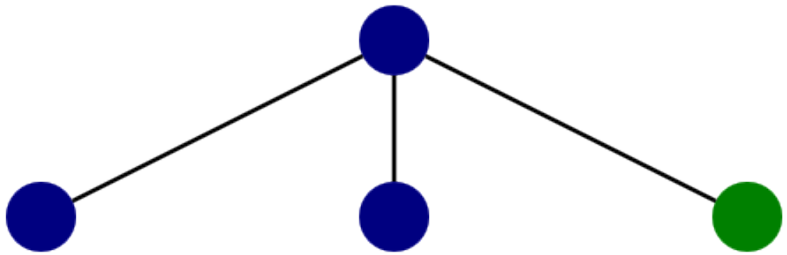
Parallélisation possible

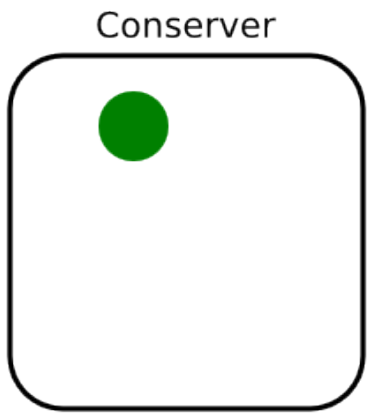
- ** Avoir un pool de nœuds à disposition **
- Tant que l'objectif n'est pas atteint, faire
  - Sélectionner un nœud ** dans le pool **
  - Générer ses fils
  - Les évaluer
  - Pour chaque fils, selon le résultat par rapport à l'objectif
    - Supprimer le nœud
    - Insérer le nœud ** dans le pool **
    - Marquer comme tout ou partie de l'objectif
- Fin Tant que

Remarques

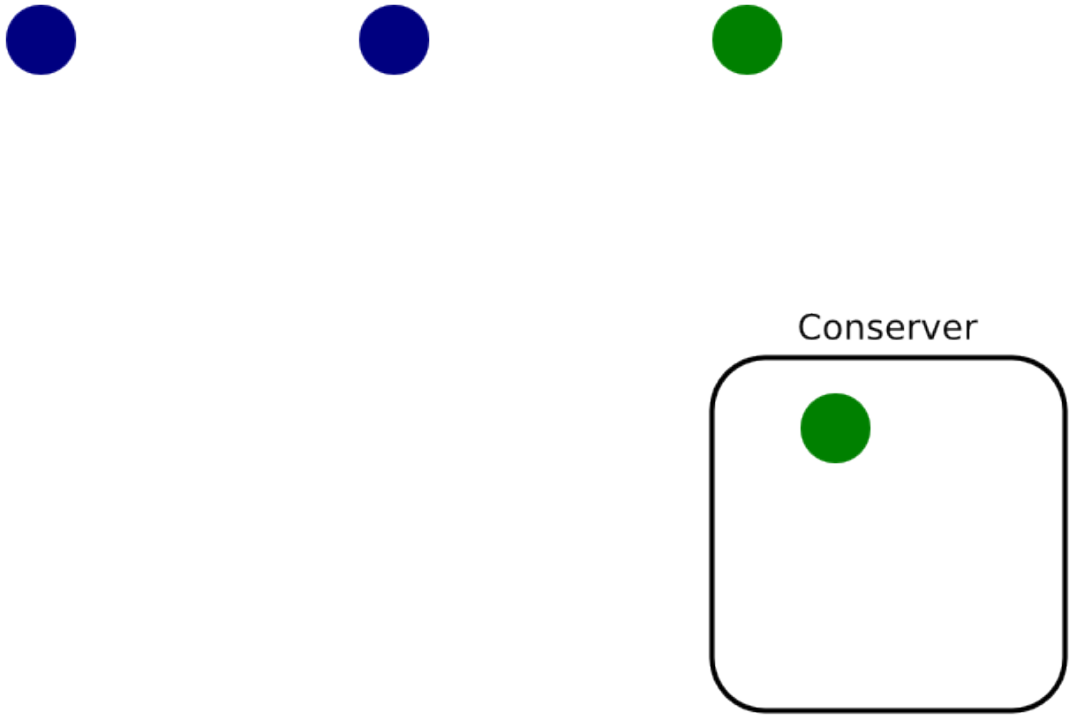
- L'objectif et le pool de nœuds doivent être connus de tous
- Les évaluations peuvent se faire simultanément

Rechercher une solution à un problème d'OC - schéma parallèle possible

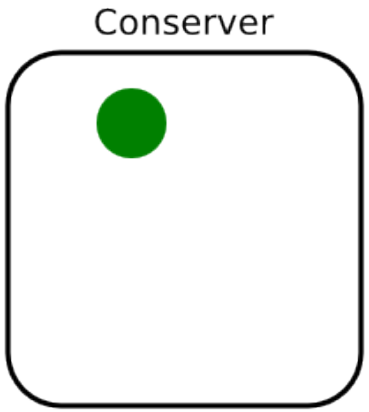




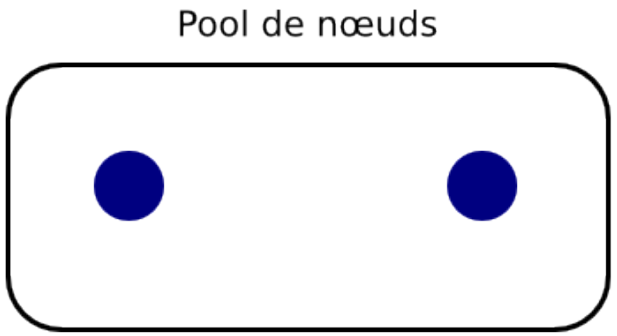
Rechercher une solution à un problème d'OC - schéma parallèle possible

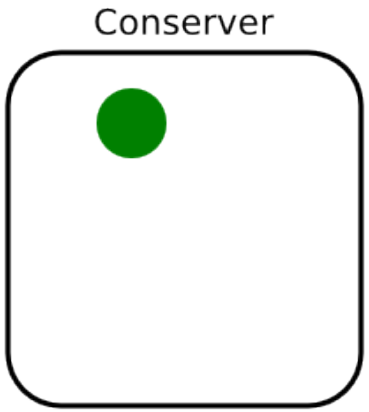


Rechercher une solution à un problème d'OC - schéma parallèle possible

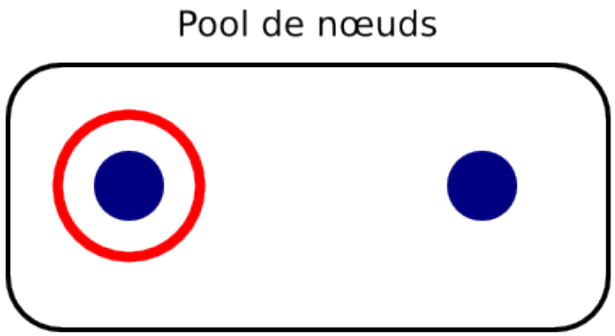


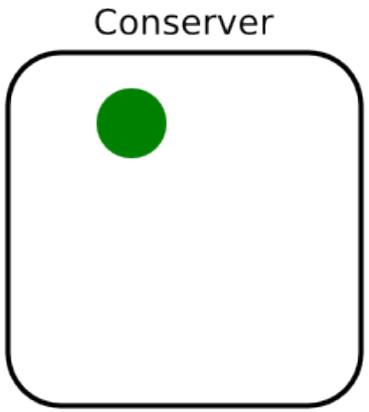
Rechercher une solution à un problème d'OC - schéma parallèle possible



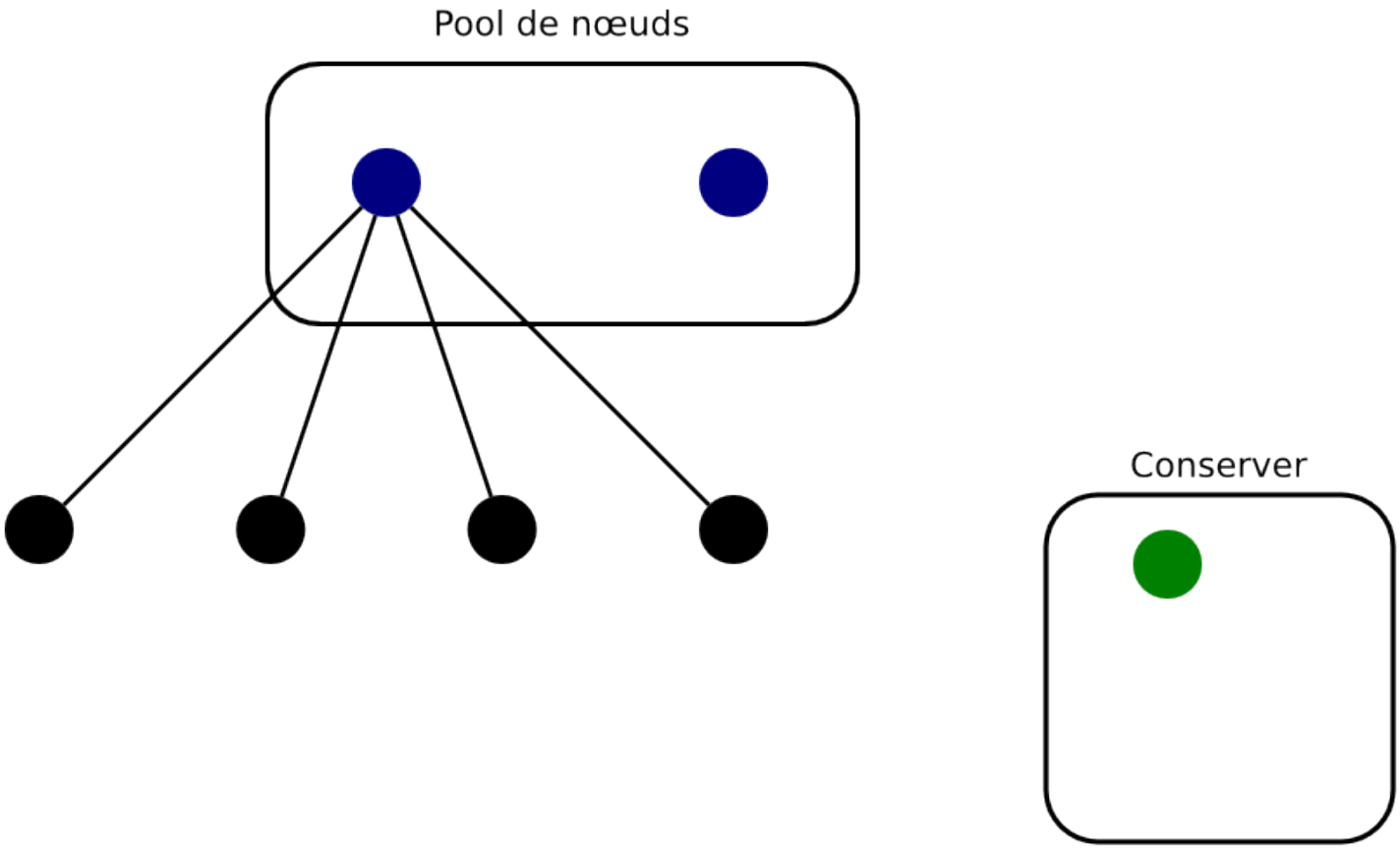


Rechercher une solution à un problème d'OC - schéma parallèle possible

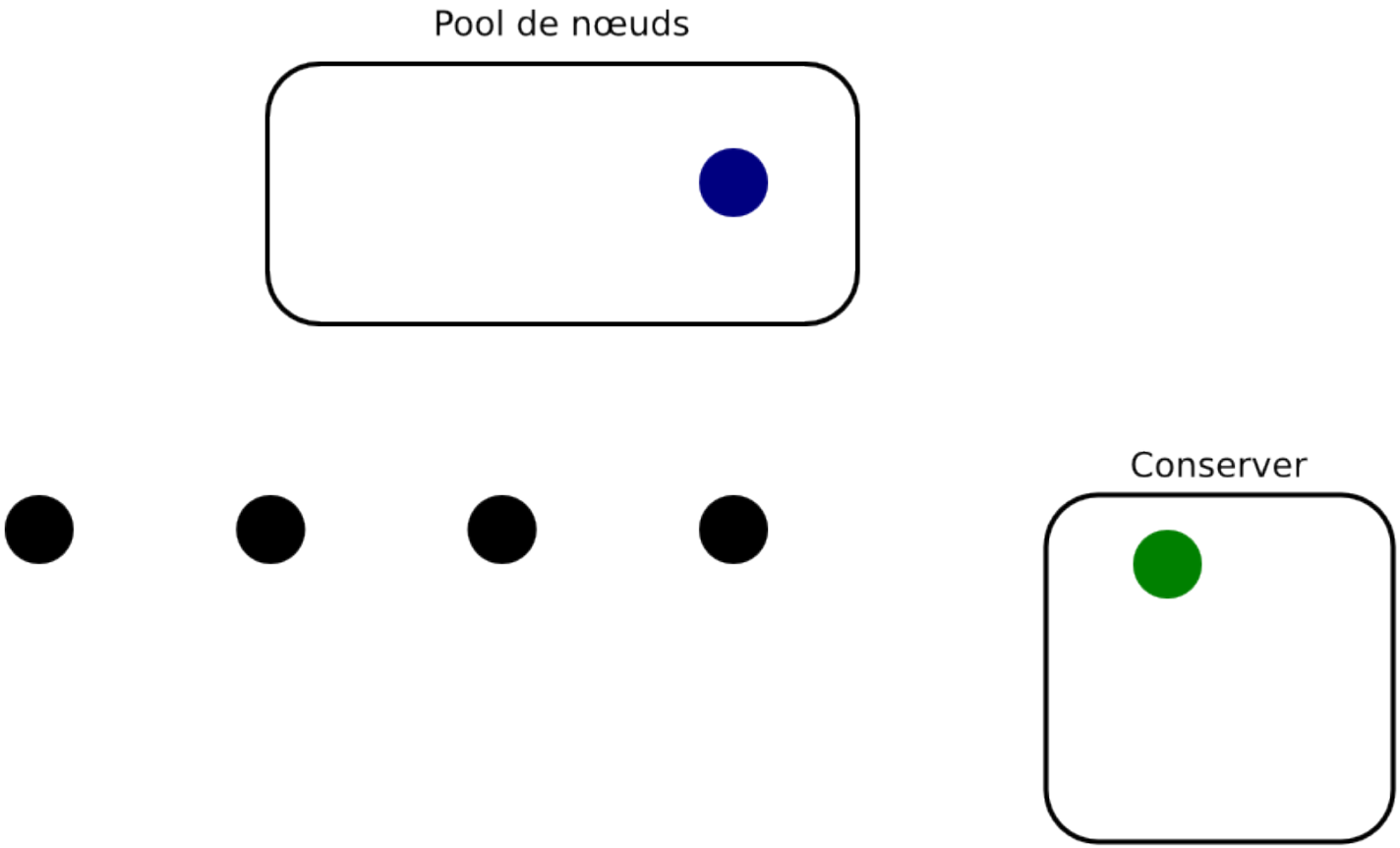




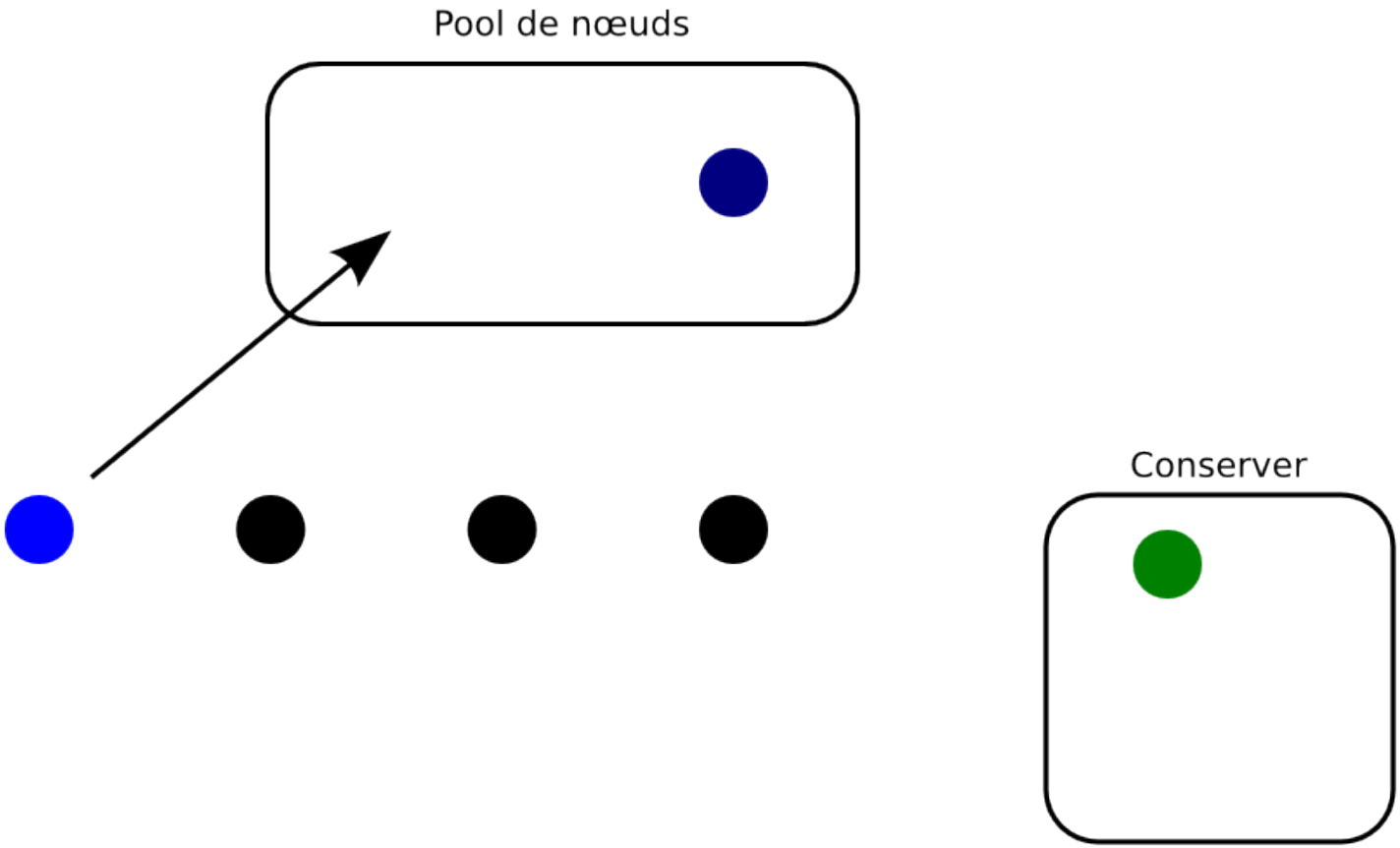
Rechercher une solution à un problème d'OC - schéma parallèle possible



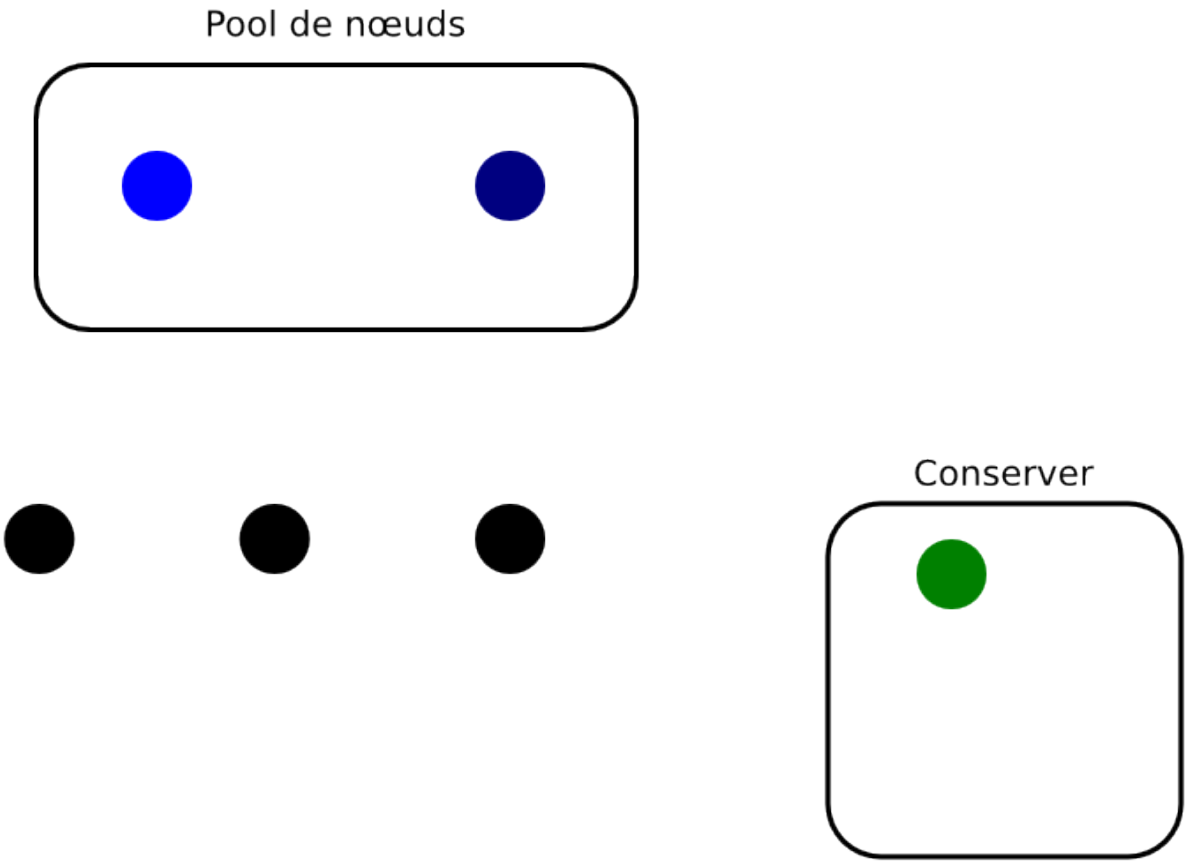
Rechercher une solution à un problème d'OC - schéma parallèle possible



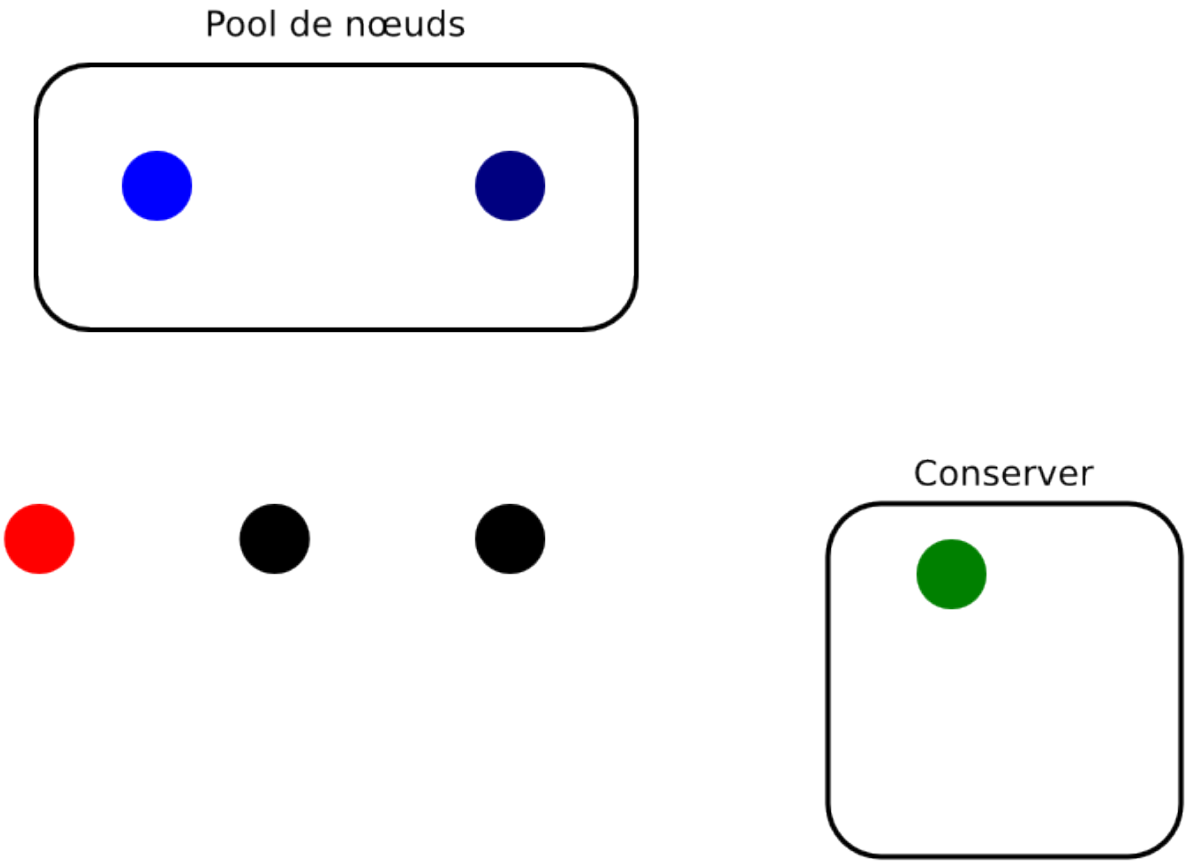
Rechercher une solution à un problème d'OC - schéma parallèle possible



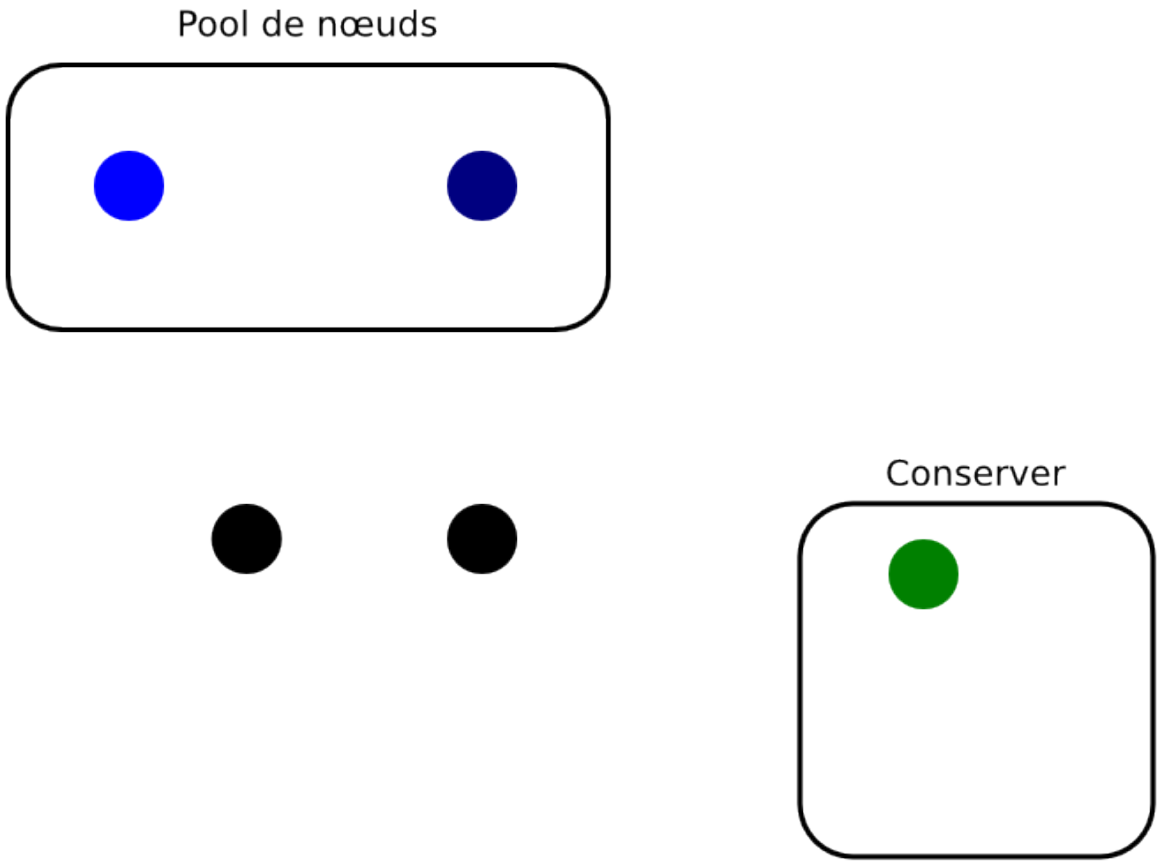
Rechercher une solution à un problème d'OC - schéma parallèle possible



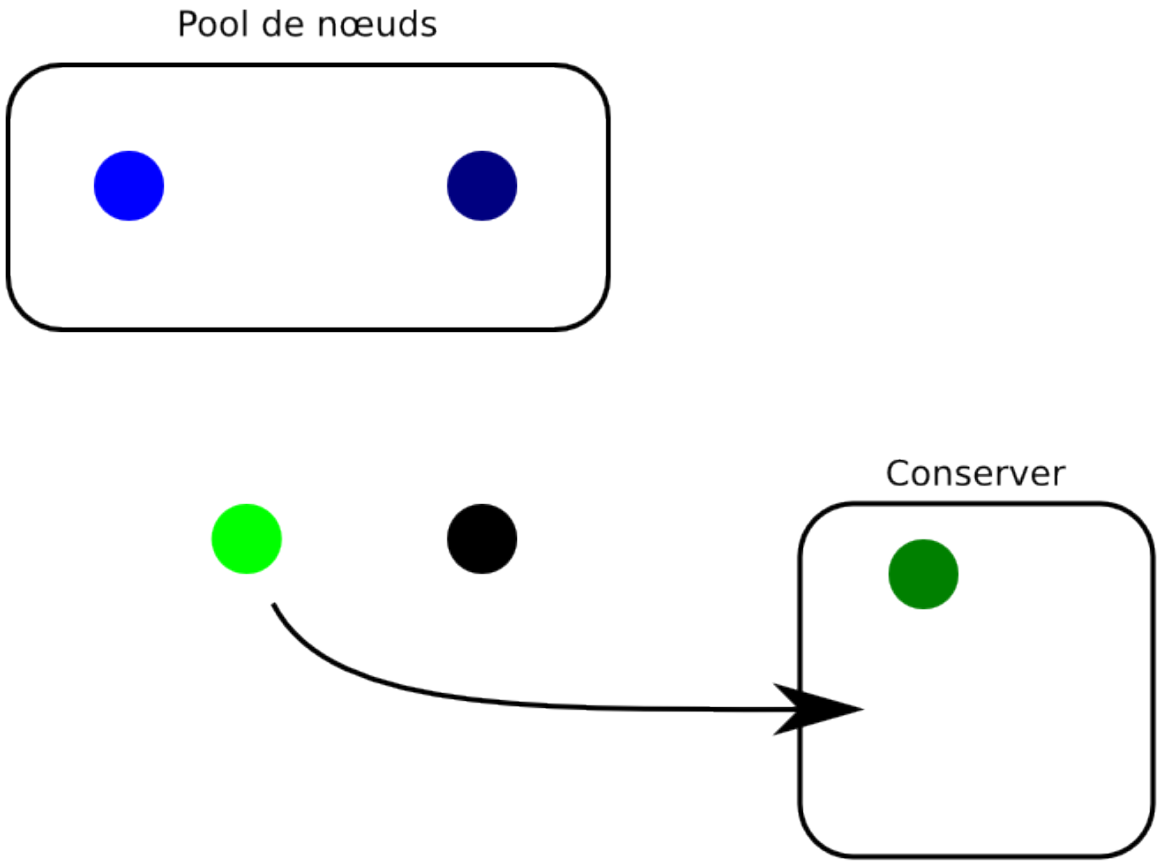
Rechercher une solution à un problème d'OC - schéma parallèle possible



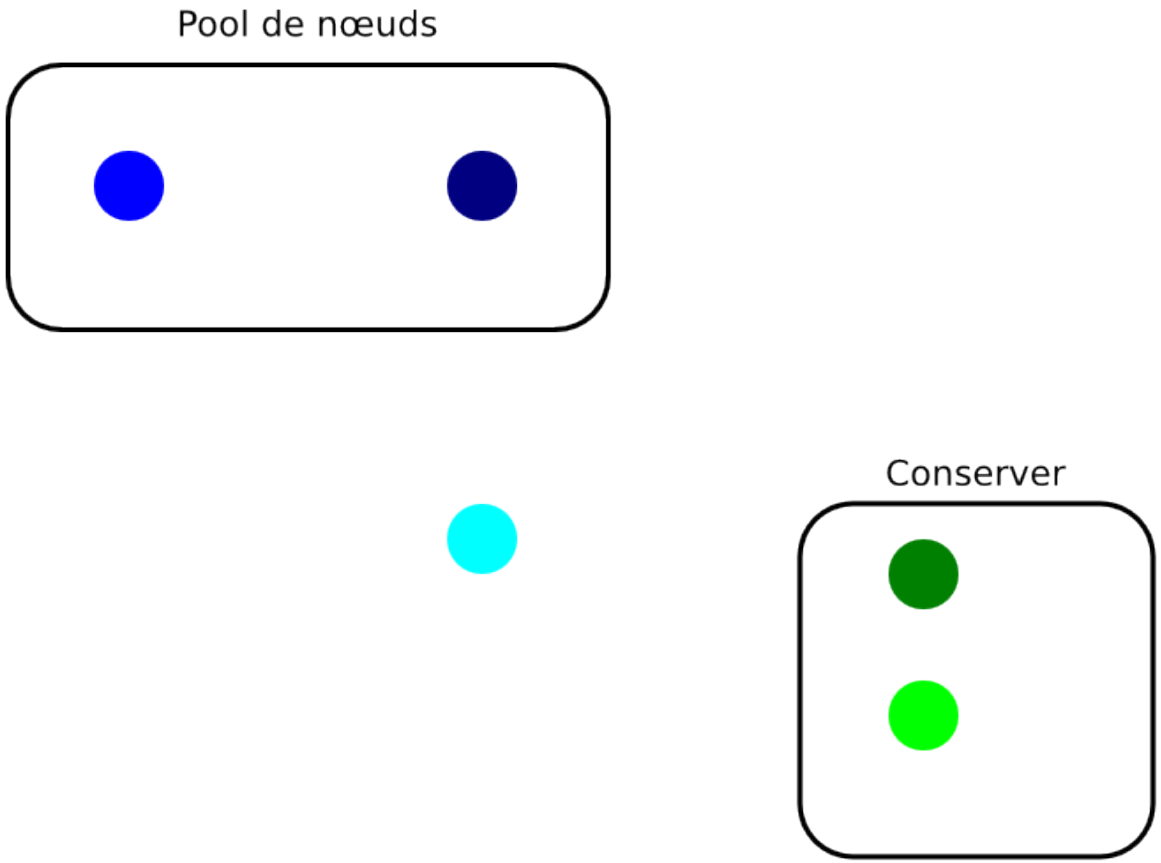
Rechercher une solution à un problème d'OC - schéma parallèle possible



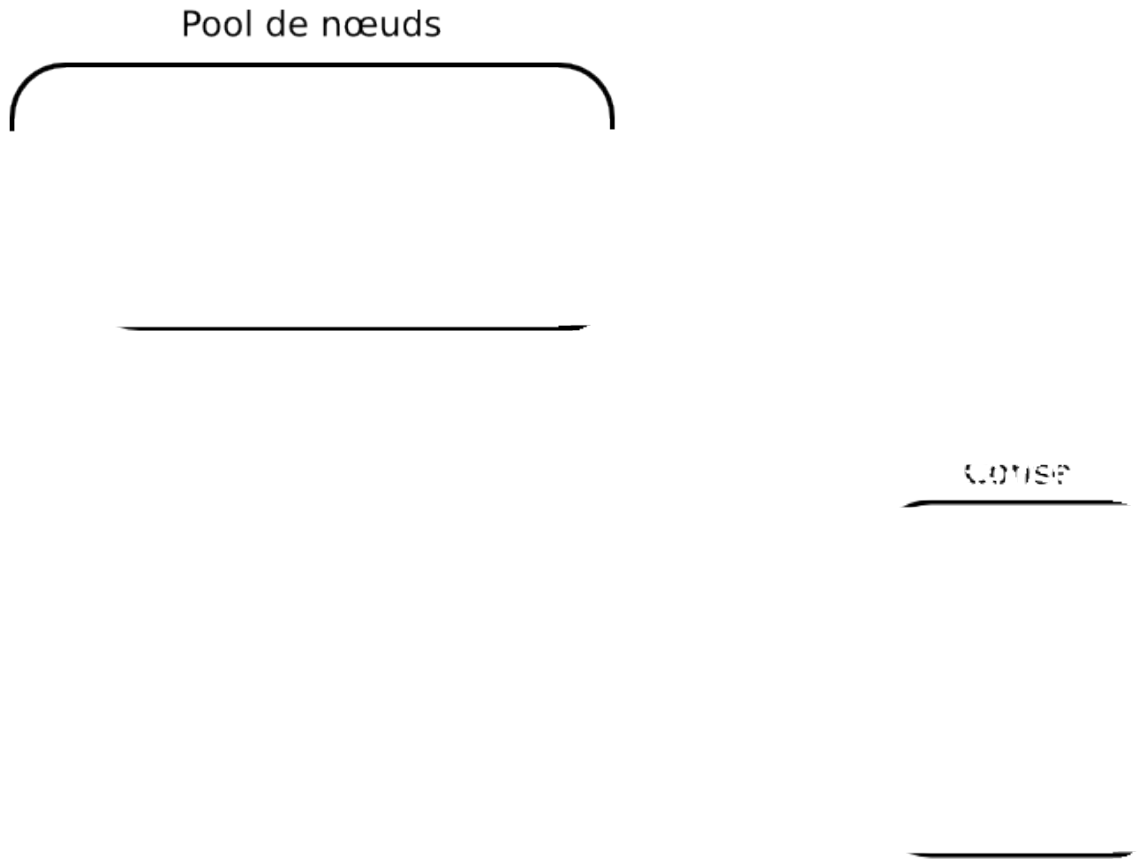
Rechercher une solution à un problème d'OC - schéma parallèle possible



Rechercher une solution à un problème d'OC - schéma parallèle possible



Rechercher une solution à un problème d'OC - schéma parallèle possible



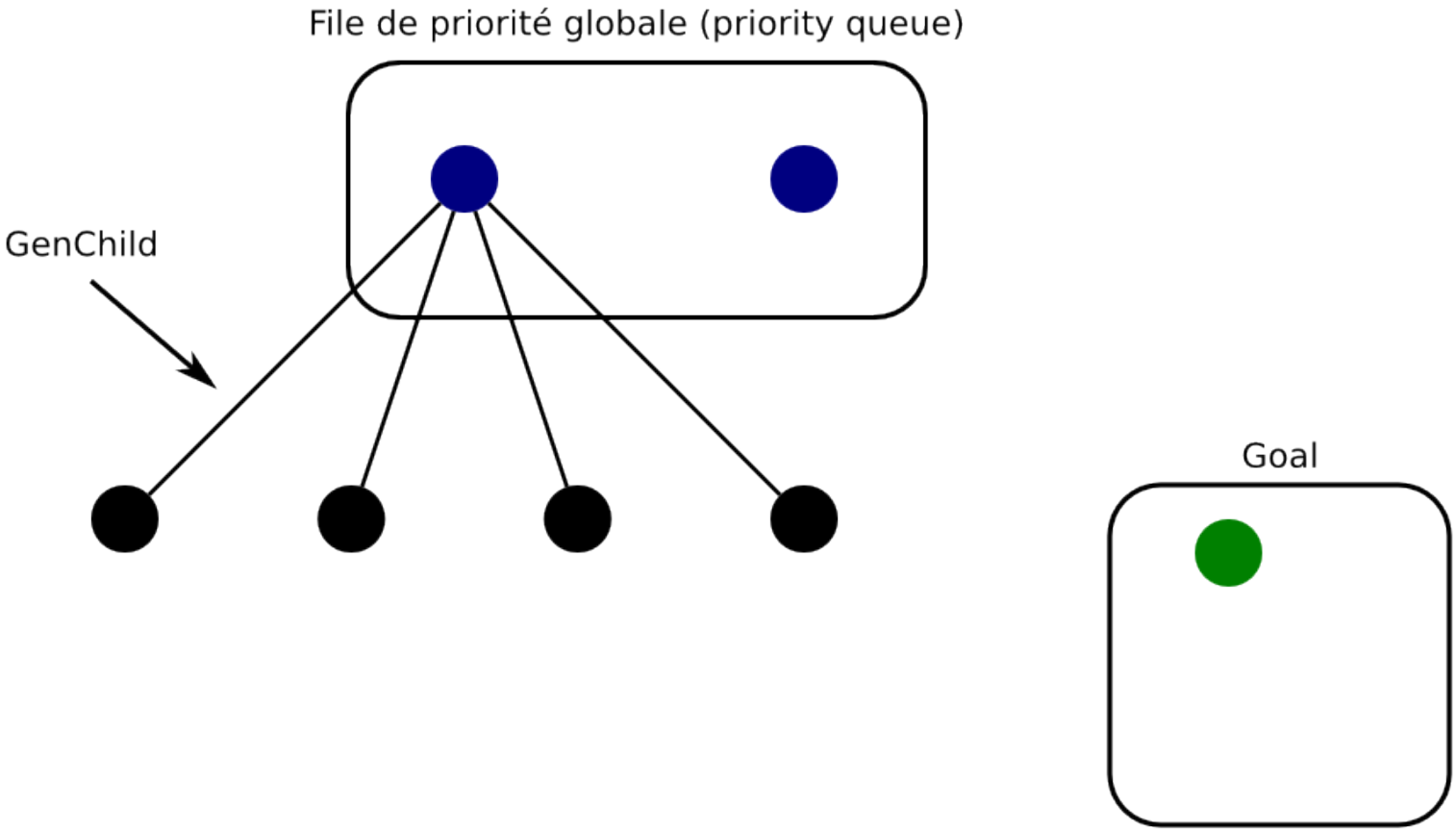
Plan

Bob++

Terminologie de bob++

- Le pool de nœuds est la file de priorité globale (« Priority Queue »)
- L'objectif à atteindre est le « Goal »
- La génération des fils est opérée par la fonction « GenChild »
- Important : l'utilisateur final n'est pas gêné par les aspects techniques de la gestion de la file de priorité globale ni par aucun aspect technique parallèle

Bob++



Comment se servir de bob++

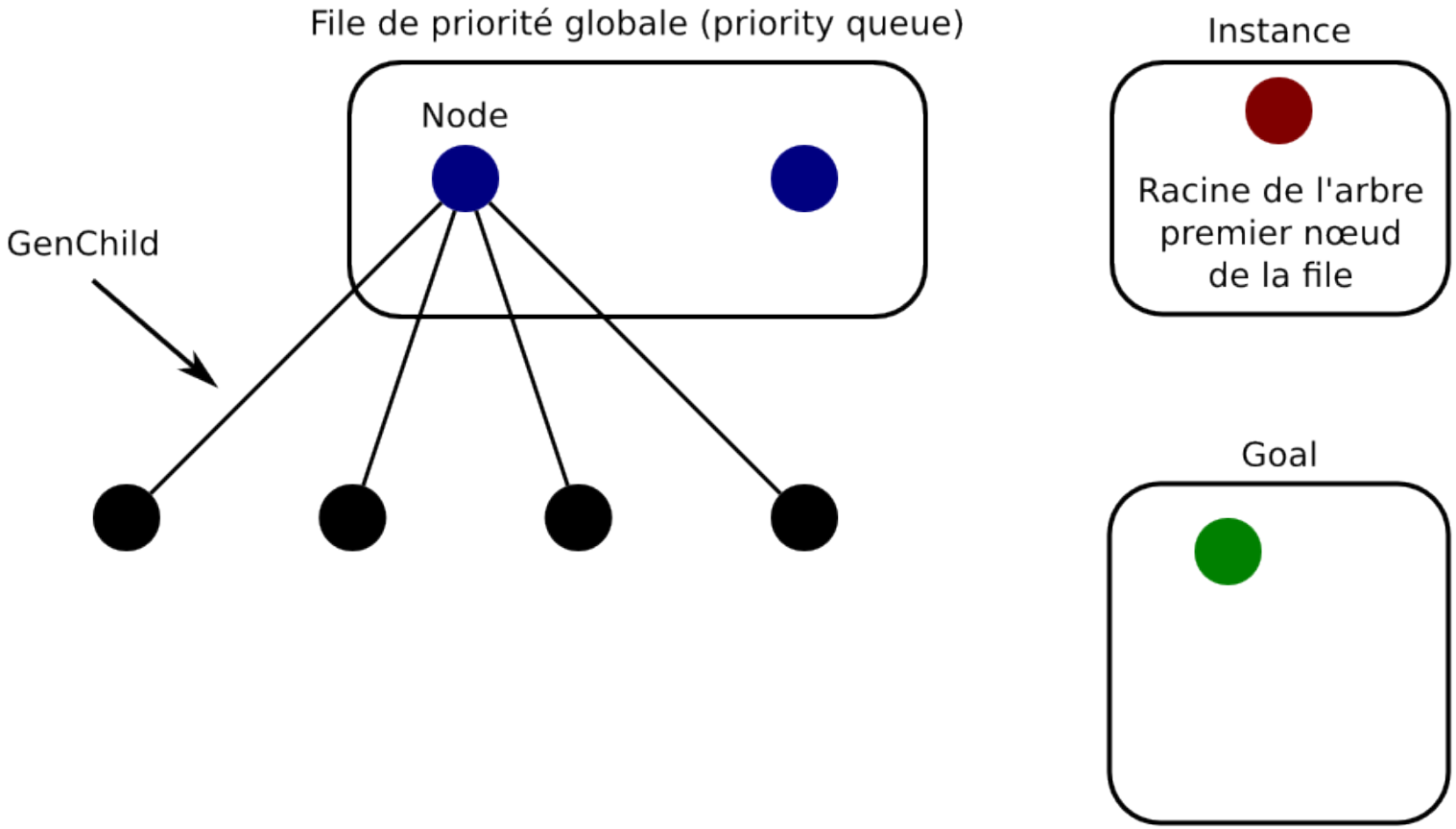
Implémenter ses objets et fonctions

- Données statiques du problème (hérite de « Instance »)
- Définir un nœud du problème (hérite de « Node »)
- Implémenter une fonction de génération des fils (hérite de « GenChild »)

Donner des directives à bob++

- Spécifier le type de problème *(exemple : « Branch & Bound »)*
- Spécifier l'objectif à atteindre selon le problème *(exemple : « Best Choice »)*
- Spécifier l'ordre de sélection dans la file de priorité globale *(exemple : « Depth First Search »)*
- *Les directives doivent avoir été prévues dans bob++*

Comment se servir de bob++



Liste des principales capacités actuelles de bob++

Types d'Algorithmes

- Branch & Bound (& Cut)
- Branch & Price
- Divide & Conquer (Costed)
- Interactions possibles avec
  - Solveurs PL
  - Générateurs de coupes

Liste des principales capacités actuelles de bob++

Types de « Goal »

- Meilleure solution réalisable
- Première solution réalisable
- Compter les (meilleures) solutions réalisables

Liste des principales capacités actuelles de bob++

Types de stratégies de sélection dans la « PQ »

- Parcours main gauche
- Parcours en largeur
- Meilleur d'abord
- Meilleur d'abord puis profondeur

Exemples

Exemple simple

Voir le fichier « queen.cpp » dans les exemples fournis avec les sources de bob++

Génération de colonnes pour HORUS

- Il existe un embryon d'algorithme pour une résolution par génération de colonnes
- Implémenter un générateur (une base à reprendre : VNS – ROADEF)
- Implémenter un algo de sélection de « bonnes » colonnes
- Implémenter les classiques bob++ (par exemple le « GenChild »)
- Le reste sera géré en interne par bob++

Plan

Bob++ et HORUS

- Amélioration de bob++ => bénéfique pour HORUS (en cours : MPI, CoinOR)
- Finalisation de l'algorithme de résolution par génération de colonnes
- Simplicité de modification et de mise à jour de l'algorithme