Rapport de projet CSP : PC configurator

Nassim Lattab

Professeur : Christian Bessiere

1. Introduction

Ce projet consiste à développer un configurateur de PC qui garantit la compatibilité des composants via un problème de satisfaction de contraintes (CSP). Deux approches ont été mises en place : l'une avec un solveur CSP, et l'autre avec la propagation de contraintes (MAC).

2. Méthodologie

Filtrage interactif des configurations en fonction des choix de l'utilisateur.

2.1. Approche avec Solveur

- Utilisation de la bibliothèque `python-constraint`.
- Pré-calcul de toutes les solutions valides.
- Assure toujours une solution compatible mais coûteux en calcul (très combinatoire).

2.2. Approche sans Solveur (MAC)

- Réduction progressive des domaines via la propagation de contraintes.
- Vérification de la cohérence globale à chaque sélection de composant.
- Évite le backtracking en garantissant la consistance locale.
- Moins gourmand en calcul mais dépend de la diversité des composants.

3. Contraintes Gérées

- Compatibilité **CPU** - **Carte Mère** (socket).
- Compatibilité **Carte Mère** - **RAM** (type de mémoire).
- Compatibilité **Carte Mère** - **Boîtier** (format de carte supporté par le boîtier).
- Compatibilité **PSU** - **Boîtier** (format d'alimentation supporté par le boîtier).
- Compatibilité **PSU** - **GPU** (puissance minimale requise).
- Contraintes budgétaires (filtrage des solutions sous un seuil)*.

**La contrainte budgétaire est uniquement intégrée dans l'approche avec solver car elle repose sur une évaluation globale des configurations. Dans l'approche sans solver, la sélection étant progressive, il est difficile d'imposer une contrainte globale de budget. Une alternative pourrait être d'afficher en temps réel le coût cumulé et d'avertir l'utilisateur en cas de dépassement.*

4. Résultats et Analyse

Les deux approches ont des avantages et des limites en fonction du contexte d'utilisation.

- **Performances :** L'approche avec solver garantit une solution valide mais peut devenir très lente en raison de la croissance exponentielle du nombre de combinaisons possibles. L'approche sans solver est plus efficace car elle réduit progressivement les possibilités au lieu de tout générer dès le départ.
- **Contraintes budgétaires :** L'approche avec solver permet de filtrer immédiatement les configurations trop chères, tandis que l'approche sans solver doit gérer cela dynamiquement en affichant le coût cumulé après chaque choix.
- **Flexibilité utilisateur :** L'approche sans solver offre une interaction plus fluide où l'utilisateur voit les composants compatibles à chaque étape, contrairement à l'approche avec solver qui affiche uniquement les solutions finales.
- **Adaptabilité à des grandes bases de données :** Lorsque le nombre de composants est très élevé, l'approche avec solver devient impraticable (explosion combinatoire), alors que l'approche sans solver s'adapte mieux en limitant les domaines à chaque sélection.

5. Conclusion et Perspectives

Ce projet a permis de comparer deux paradigmes différents pour résoudre un CSP. L'approche avec solver explore exhaustivement toutes les combinaisons, garantissant une solution mais avec un coût élevé. À l'inverse, l'approche par propagation de contraintes avec réduction de domaine restreint dynamiquement les choix, réduisant la complexité. Cette comparaison souligne l'importance du choix de la méthode selon le contexte.

Une amélioration future pourrait consister à hybrider ces deux approches, conciliant exhaustivité et efficacité. De plus, l'intégration d'un moteur d'optimisation multi-critères (performance, volume sonore, etc.) permettrait d'affiner la sélection des configurations optimales selon différents besoins.

6. Instructions d'Exécution

- Prérequis : Python 3.x, pandas, constraint.
- Installation des dépendances :

```
pip install -r requirements.txt
```

- Lancer le programme :

```
python interactive_pc_builder_with_solver.py
python interactive_pc_builder_without_solver.py
```

- Sélectionner les composants et obtenir une configuration valide.

7. Annexe : Graphe des Contraintes

Le graphe ci-dessous illustre les principales relations de compatibilité entre les composants du PC configuré.

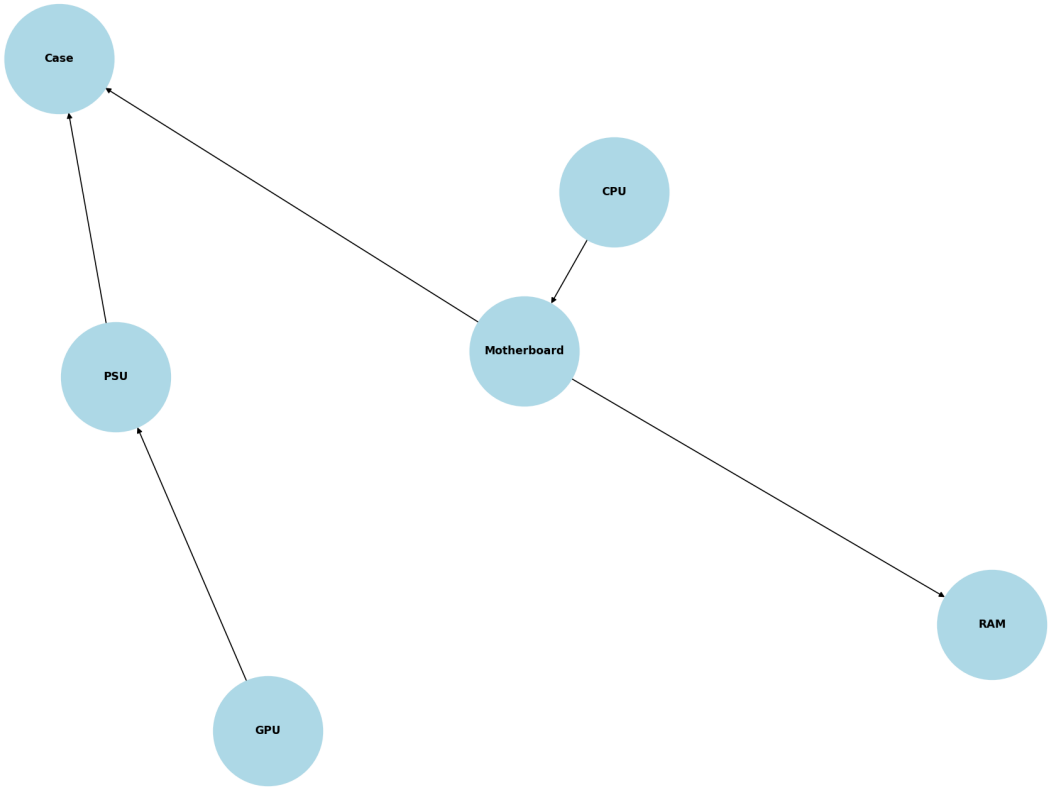

Figure 1. Graphe des contraintes du configurateur PC