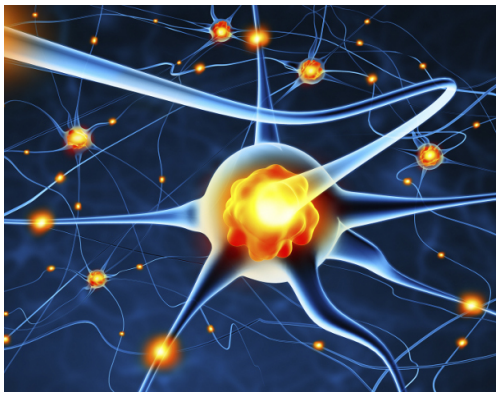
RAPPORT DE STAGE

Titre du rapport

Minatchy Jérôme
M1 Informatique
Année universitaire 2020/2021 à rendre pour le 3 Juin 2021



Université des Antilles
(Entreprise d'accueil)

Table des figures

Table des matières

Chapitre 1

Introduction

1.1 Présentation de la structure d'accueil

Durant la période de mon stage , j'ai été accueilli au **Laboratoire de Mathématiques Informatique et Application (LAMIA)** de l'Université des Antilles (UA).

Pour présenter cette structure, il me faut tout d'abord présenter l'université à laquelle il est rattaché.

1.1.1 L'université des Antilles

Bien que ce soit l'université dans laquelle j'ai fait toutes mes études, voici quelques chiffres que je ne connaissais pas et qui donnent la mesure de sa taille :

L'Université des Antilles s'organise autour deux pôles universitaires régionaux autonomes : le « Pôle Guadeloupe » et le « Pôle Martinique ».

Sur ces pôles, l'Université assure des missions d'*enseignement* et de *recherche*, assistées par des *administratifs et des techniciens*.

Administration et personnel technique

l'UA emploie 414 Administratifs et Techniciens (environ 200 personnes pour l'administation centrale et 100 répartis sur chaque pôle)

Enseignements

L'UA délivre des diplomes de la licence au doctorat dans de nombreux domaines. Au total, cela représente :

— 484 enseignants-chercheurs (environ 240 pour chaque pôle)

— 12 000 étudiants (environ 7000 pour la Guadeloupe , 5000 pour la Martinique)

Pour l'informatique, cela représente : - autour de 20 enseignants-chercheurs - autour de 120 étudiants

1.1.2 Le LAMIA

Le **Laboratoire de Mathématiques Informatique et Application (LAMIA)**, comme son nom l'indique, se concentre sur les recherches en informatiques et mathématiques.

Il compte une soixantaine de membres (Professeurs des Universités, Maitres de Conférences, ATER, Doctorants) répartis sur deux pôles (Guadeloupe et Martinique) au sein de trois équipes internes :

— Equipe **Mathématiques** (analyse variationnelle, analyse numérique, EDP, analyse statistique, mathématiques discrètes) ;

— Equipe Informatique **DANAIS** : Data analytics and big data gathering with sensors ;

— Equipe Informatique **AID** : Apprentissages Interactions Donnees ;

De plus, le LAMIA accueille en son sein un groupe de chercheurs associés travaillant en Epidémiologie clinique et médecine.

1.2 Contexte général

1.2.1 Qu'est ce que le sudoku

Le sudoku est un jeu représenté par une grille de 81 case découpé en 9 lignes et 9 colonnes et 9 sous-grilles 3 par 3. Le but du jeu est de remplir chaque ligne avec 9 chiffre allant de 1 à 9 en faisant en sorte qu'il n'y ai pas le même chiffre plusieurs fois sur la même ligne colonne ou dans la même sous grille.

5	3	4	6	7	8	9	1	2
6	7	2	1	9	5	3	4	8
1	9	8	3	4	2	5	6	7
8	5	9	7	6	1	4	2	3
4	2	6	8	5	3	7	9	1
7	1	3	9	2	4	8	5	6
9	6	1	5	3	7	2	8	4
2	8	7	4	1	9	6	3	5
3	4	5	2	8	6	1	7	9

FIGURE 1.1 – Exemple de Sudoku complet.

1.3 Contexte du problème

La résolution de sudoku est un sujet ou plusieurs solutions existent et où c'est dans la complexité [1] des différentes solution que réside la difficulté la résolution. Nous pouvons aussi rendre compte de résolution de sudoku avec des règles spécifiques.
tel que :

4	1	5	6	3	8	9	7	2
3	6	2	4	7	9	1	6	5
7	8	9	2	1	5	3	6	4
9	2	6	3	4	1	7	5	8
1	3	8	7	5	6	4	2	9
5	7	4	9	8	2	6	3	1
2	5	7	1	6	4	8	9	3
8	4	3	5	9	7	2	1	6
6	9	1	8	2	3	5	4	7

FIGURE 1.2 – Sudoku où l'on applique la règle de l'unicité des chiffres sur les diagonnales.

Dans ce mémoire nous allons éssentiellement parler de deux d'entre elles celle de la résolution de sudoku vu sous l'angle d'un problème d'optimisation linéaire grace a l'algorithme du simplex et une autre plus simple celle de l'algorithme du backtraking.

1. le nombre d'action réalisé durant la résolution

1.4 Méthodologie

1.4.1 Outils utilisés

Présentation de Python

FIGURE 1.3 – Logo de Python.

Python est un langage de programmation interprété [2] qui sera utiliser pour l'ensemble du projet. Nous avons choisi ce langage car il n'y a pas beaucoup de résolution

Présentation Qt

FIGURE 1.4 – Logo de Qt.

QT est une librairie [3] qui permet le création d'interface graphique en Python.Que nous utiliserons pour créer l'interface que l'on utilisera au cours du projet.

Présentation Cplex

FIGURE 1.5 – Logo de Cplex.

Cplex est une librairie[3] qui permet la modélisation et la résolution de problème d'optimisation linéaire [4].

2. Langage nécéssitant un programme informatique qui joue le rôle d'interface entre le projet et le processeur appellé interpréteur, pour exécuter du code.
3. Une librairie est un fichier contenant du code (généralement un ensemble de fonction et classes permettant de faciliter et/où de réaliser certain programme)
4. Terme que j'expliquerais plus tard dans mon rapport

Présentation de GitHub

FIGURE 1.6 – Logo de GitHub.

Nous pouvons définir GitHub comme une plateforme de développement de projet informatique en groupe. Elle simplifie grandement le développement de projets. Elle permet de versioner ses programmes et d'y apporter des modifications en temps réel à plusieurs.

Présentation de LaTex

FIGURE 1.7 – Logo de LaTex

Nous pouvons dire que LaTex est un langage de traitement de texte tel que le markdown qui permet de mettre en forme notre texte de manière s̈cientifiqueëcela veut dire que. LaTex permet une faciliter d'écriture des équations et de toutes les écriture mathématiques.Permet de par ses nombreux package une quasi-infinité de possibilitées. L'utilisation de cet outil permettra une synergie entre ceux-ci car LaTex peut-être utiliser avec un simple bloc-note c'est donc du texte ce qui permet une intéraction facilitée avec GitHub d'ailleurs ce rapport est écrit avec Latex et retrouvable sur GitHub.

1.5 Annonce du plan

1.5.1 Présentation des stratégies de résolution

Dans cette première partie je commencerais par vous présenter la première solution de résolution choisie qui est la résolution du sudoku en tant que problème d'optimisation linéaire.
Je vous expliquerai ce qu'est un problème d'optimisation linéaire puis vous décrirai l'algorithme du simplex qui nous permettra de le résoudre.
En deuxième grande sous partie de cette section je vous présenterai l'algorithme du backtraking.
En dernière sous partie de cette présentation je vous expliquerai pourquoi le choix de ces deux solutions.

1.5.2 Implémentation des stratégies

La première sous partie de cette grande sous partie commencera avec la modélisation et l'implémentation d'un sudoku en python et l'implémentation de l'interface graphique.
La seconde sera l'implémentation de la méthode résolution utilisant cplex.
Pour finir nous ferons l'implémentationde la méthode utilisant l'algorithme du backtracking.

1.5.3 Test du résolveur

Nous commencerons par établir nos méthodes de test, expliquer la raison du choix de ces test.
Nous continuerons avec l'implémentation de ceux-ci.
Nous finirons par présenter et analyser nos résultats.

1.5.4 Conclusion

Nous terminerons ce rapport par une conclusion où nous rappelerons brièvement la problématique.
Nous ferons le bilan des produits du projet de stage.
Nous ferons le bilan des apports du stage.
Et finirons par une ouverture en étudiants nos perspectives.

Chapitre 2

Déroulement

Ce chapitre, le plus volumineux du rapport, décrira l'ensemble des tâches que j'ai eu à effectuer au cours de ces deux mois.

2.1 Présentation des stratégies de résolution

2.1.1 Présentation de la résolution avec Cplex

Qu'est-ce qu'un problème d'optimisation linéaire

Nous allons dans cette première partie parler de la résolution de sudoku comme étant un problème d'optimisation linéaire. Mais tout d'abord nous devons définir ce qu'est un problème d'optimisation linéaire.

Par définition un problème d'optimisation linéaire est un problème dont la valeur à obtimiser ainsi que les contraintes qui y seront appliquer peuvent-êtres modélisées sous la forme d'une fonction linéaire cette description n'est pas des plus précises que l'on puisse c'est pour cela que nous allons l'étoffer par un problème connu.

Le problème du brasseur de bière peut être énoncer comme suit :

Un brasseur fabrique 2 types de bières : blonde et brune.
3 ingrédients : maïs , houblon , malt.
Quantités requises par unité de volume :
Bière blonde : 2,5 kg de maïs, 125 g de houblon, 17,5 kg de malt
Bière brune : 7,5 kg de maïs, 125 g de houblon, 10 kg de malt
Le brasseur dispose de 240 kg maïs, 5 kg houblon , 595 kg malt
Prix vente par u.v. : blonde 9 euros , brune 15 euros Le brasseur veut maximiser son revenu .
Quelle quantité de bières blondes et/ou brunes doit-il produire pour cela?

Nous pouvons modéliser le revenu du brasseur comme étant une fonction linéaire tel que :
Soit x^1 le nombre de volumes d'unité de bière blonde et x^2 le nombre d'unité de volume de bière brune
Nous avons :

Le revenu que nous de vons maximiser : $R = x^1 * 9 + x^2 * 15$

Les contraintes peuvent êtres représentées comme suit :

La quantité maximale de maïs : $M \geq x^1 * 2,5 + x^2 * 7,5$
La quantité maximale de houblon : $H \geq x^1 * 0,125 + x^2 * 0,125$
La quantité maximale de malt : $Ma \geq x^1 * 17,5 + x^2 * 10$

Comment résoudre un problème d'optimisation linéaire

Maintenant que nous avons vu ce qu'est un problème d'optimisation linéaire et illustré ceci par un exemple nous allons voir comment peut-on le résoudre plus particulièrement comment avec l'algorithme du simplex en théorie puis avec Cplex.

La résolution avec cplex est des plus simples il suffit de mettre en place nos contraintes et notre fonction à maximiser le code sera donner et expliciter en annexe :

```
Version identifier: 20.1.0.0 | 2020-11-10 | 9bedb6d68
CPXPARAM_Read_DataCheck                          1
Tried aggregator 1 time.
No LP presolve or aggregator reductions.
Presolve time = 0.02 sec. (0.00 ticks)

Iteration log . . .
Iteration:     1   Scaled dual infeas =             0.000000
Iteration:     2   Dual objective     =           528.000000
Solution status =  1 :
optimal
Solution value  =  528.0
Row 0:  Slack =   0.000000  Pi =   1.200000
Row 1:  Slack =   0.000000  Pi =  48.000000
Row 2:  Slack = 105.000000  Pi =  -0.000000
Column 0:  Value =  12.000000 Reduced cost =   0.000000
Column 1:  Value =  28.000000 Reduced cost =   0.000000
```

FIGURE 2.1 – Résultat de l'optimisation du problème du brasseur

Nous voyons que nous avons pour revenu maximum 528 euro avec 12 unité de volume de bière blonde produite et 28 uinité de bière brune produites.

2.1.2 Présentation de la résolution par backtracking

L'algorithme du backtracking est plus précisément une famille d'algorithme qui sont utilisé le plus souent pour résoudre des problmèmes de satisfaction de contrainte. Le backtracking est la solution la plus simple à comprendre il s'agit tout simplement de tester toutes les valeurs possible sur chaque case jusqu'a obtenir une sortie de sudoku remplie et valide.Cette algorithme agit comme un parcours d'arbre illustrons cela avec une image :

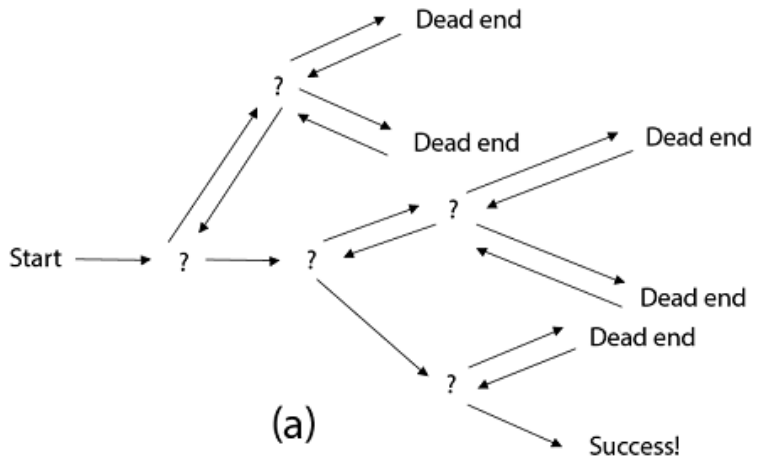


FIGURE 2.2 – Illustration de l'algorithme du backtracking.

Comme l'illustre cette image nous essayons toutes les possibilitées et retournons en arrière si nous tombons dans une impasse dans nous passons de case vide [1] en case vide[1] en revenant en arrière si peu importe la le chiffre essayer dans cette case allant de 1 à 9, nous donne un sudoku invalide. Le problèlme de cette algorithme qui reviens le plus souvent est dù au fait que nous essayons toutes les valeurs possible ce qui nous donne une très grande quantité d'action et donc ce qui augmente considérablement notre temps de calcul nous pouvons toutefois régler ce problème en évitant de tester les possibiliter qui nous amène de façon logique à une impasse ou éviter de tester des possibilité d'une case si il est possible de déduire sa valeur de façon logique.

2.1.3 Pourquoi utiliser ces deux algorithme

Pourquoi utiliser l'algorithme du simplex

La résolution avec l'algorithme du simplex permet une implementation intuitive et facililité dù à l'utilisation de Cplex. Cplex nous permet aussi une meilleur étude de la complexité de notre algorithme car cous pouvons voir les différentes itération de recherche de notre outil de résolution. Cette méthode de résolution nous permet aussi de revoir le problème de plusieur façon différente de par le fais que la résolution ne dépends que de nos contraintes.

Pourquoi utiliser l'algorithme utilisant le backtracking

La résolution avec avec backtracking est l'algorithme le plus naturel qui viennent à l'esprit dans la résolution de problème avec contrainte ce qui permet une facile implémentation de la gestion des contraintes malgré leur nombre.C'est un algorithme facilement améliorable car le but tout au long de son implémentation sera de tester le moins de valeur menant à une impasse possible. Cette algorithme a aussi pour avantage de n'avoir besoin daucune librairie extérieur ce qui permet une liberté totale au niveau du code et donne lieu à une certaine transparence quand à son fonctionnement contrairement aux au code que nous utilisons via cplex qui peuvent différé légèrement de leurs fonctionnements théoriques cités plus haut.

2.1.4 Implémentation des Stratégies de résolution

2.1.5 Modélisation et implémentation d'un sudoku en Python et de l'interface graphique

Nous allons stocker dans une liste [2] contenant 9 listes[2] qui représenteront nos ligne qui elles contiendront 9 entiers allant de 0 à 9. Nous aurons donc besoin de coder des fonctions pour faire le lien entre l'interface et notre résolveur du fait de notre modélisation du sudoku nous pouvons découper cela en plusieurs fonction utilitaire :

- Une fonction qui nous permettra de stocker le contenu de notre interface graphique dans une liste comme celle décrite précédemment
- Une fonction permettant de transférer le contenu d'une liste comme celle décrite plus haut dans notre interface graphique
- Une fonction permettant de copier le contenu de la zone de saisi de notre interface dans la zone d'affichage de notre interface
- Une fonction permettant de copier le contenu de la zone d'affichage de notre interface dans la zone de saisi de notre interface

1. Une case vide est comme dit plus haut une case contenant la valeur 0
2. structure de donnée incluse de base dans python qui permet de contenir plusieurs variables différentes

Chapitre 3

Conclusion

3.1 Rappel de la problèmatique

3.2 Réponse apportées

3.3 Piste d'amélioration

3.4 Les apports du stage

3.4.1 les apports a l'entreprise

3.4.2 les apports personels

3.5 Perspectives

Chapitre 4

Remerciements

Bibliographie

Annexe A

Annexes