AU : 2017/2018

Université de Sousse

Ecole Nationale d'Ingénieurs de Sousse

Rapport de Projet Module Algorithmique Avancé

Réalisé par :

Hamza Slama

Mohamed Khemiri

Filière :

2ème Année

Génie Informatique Appliquée

Encadré par

Mr Walid Chainbi

Période : 1er semstre

Sommaire

Introduction Générale

Dans la vie, l'être humain tend toujours à choisir le chemin le plus simple. La plupart des problèmes ont au moins une solution. Dans le cas où il existe plusieurs solutions qui ramènent au même résultat, il faut poser quelques questions, Quelle est la différence entre une solution et une autre ? Quelle est la plus évidente ? Quelle est la plus simple ? Quelle est la plus rapide ?.

Afin d'illustrer la notion de l'algorithme, partons de l'exemple suivant : avez-vous déjà indiqué un chemin à un touriste égaré ? Avez-vous fait chercher un objet à quelqu'un par téléphone ? Si oui, vous avez déjà fabriqué des algorithmes.

Un algorithme, c'est une suite d'instructions, qui une fois exécutée correctement, conduit à un résultat donné. Si l'algorithme est juste, on atteint le résultat ciblé, et le touriste se retrouve là où il voulait aller. Si l'algorithme est faux, le résultat est, disons, aléatoire, et décidément, cette saloperie de répondeur ne veut rien savoir.

Alors, dans notre domaine, apprendre l'algorithmique, c'est apprendre à manier la structure logique d'un programme informatique. Cette dimension est présente quelle que soit le langage de programmation.

Bref, Nous allons présenter dans ce rapport une application basée sur un algorithme bien spécifique : le sac à dos. Donc, dans une première partie, nous parlons du problème du sac a dos : historique , principe et méthode de résolution. Puis dans une seconde partie, Nous présentons notre application. Ensuite, nous citons les techniques de développement et de réalisation, et nous terminons par la conclusion et les perspectives.

Chapitre 1
Problème de SAC A DOS

1. Introduction

Le problème du sac à dos fait partie des problèmes d'optimisation combinatoire les plus étudiés ces cinquante dernières années, en raison de ces nombreuses applications dans le monde réel. En effet, ce problème intervient souvent comme sous-problème à résoudre dans plusieurs domaines : la logistique comme le chargement d'avions ou de bateaux, l'économie comme la gestion de portefeuille ou dans l'industrie comme la découpe de matériaux.

2. Historique

Le problème du sac à dos est l'un des 21 problèmes NP-complets de Richard Karp, exposés dans son article de 1972. Il est intensivement étudié depuis le milieu du XX^{e} siècle et on trouve des références dès 1897, dans un article de George Ballard Mathews. La formulation du problème est fort simple, mais sa résolution est plus complexe.
Les algorithmes existants peuvent résoudre des instances pratiques de taille importante. Cependant, la structure singulière du problème, et le fait qu'il soit présent en tant que sous-problème d'autres problèmes plus généraux, en font un sujet de choix pour la recherche.

3. Principe

D'une façon générale, le problème du sac-à-dos consiste à remplir un sac dont la capacité est fixée, avec un sous ensemble d'objets de poids et de profit connus, de manière à satisfaire les deux conditions suivantes :

a. Le poids cumulé du sous-ensemble d'éléments choisi ne dépasse pas la capacité du sac.

b. Le profit généré par le sous-ensemble d'éléments choisi est maximal. Dans ce cas, il faut savoir quels sont les objets qu'on mettra dans le sac pour maximiser ce profit. [1]

En pratique, le problème du sac-à-dos possède de nombreuses applications. Il apparait dans plusieurs situations comme sous-problème aidant à la résolution d'autres problèmes (cf., Gilmore et Gomory [2], et Hifi et Roucairol [3]). Ce qui fait de lui un modèle théorique particulièrement intéressant.

Les données du problème peuvent être exprimées en termes mathématiques. Les objets sont numérotés par l'indice i variant de1à n. Les nombres wi et pi représentent respectivement le poids et la valeur de l'objet numéro i La capacité du sac sera notée W.

Il existe de multiple façon de remplir le sac à dos. Pour décrire l'une d'elle if faut indiquer pour chaque élément s'il est pris ou non. On peut utiliser un codage binaire. L'état d'i-ème élément vaudra xi = 1 si l'élément est mis dans le sac, ou xi = 0 s'il est laissé de côté. Une façon de remplir le sac est donc complètement décrite par un vecteur appelé vecteur contenu ou simplement contenu X = ($x1$, $x2$, … , xn) ; et le poids associé, ainsi que la valeur associée à ce remplissage, peuvent alors être exprimés comme fonction du vecteur contenu. Pour un contenu X donné, la valeur totale contenue dans le sac est naturellement :

$$z(X) = \sum_{\{i,x_i=1\}} p_i = \sum_{i=1}^{n} x_i p_i$$

Figure 1.Somme des valeurs

De même, la somme des poids des objets choisis est :

$$w(X) = \sum_{\{i,x_i=1\}} w_i = \sum_{i=1}^{n} x_i w_i$$

Le problème peut être reformulé comme la recherche d'un vecteur contenu X= ($x1$, $x2$, … , xn) (les composants valant 0 ou 1), réalisant le maximum de la fonction valeur totale z(X), sous la contrainte :

$$w(X) = \sum_{i=1}^{n} x_i w_i \leq W$$

c'est-à-dire de la somme des poids des objets choisis ne dépasse pas la capacité du sac à dos[4]

4. Méthodes de résolution

Il existe deux grandes catégories de méthodes de résolution de problèmes d'optimisation combinatoire : les méthodes exactes et les méthodes approchées. Les méthodes exactes permettent d'obtenir la solution optimale à chaque fois, mais le temps de calcul peut être long si le problème est compliqué à résoudre. Les méthodes approchées, encore appelées heuristiques, permettent d'obtenir rapidement une solution approchée, donc pas nécessairement optimale. Nous allons détailler un exemple d'algorithme de résolution de chaque catégorie.

4.1 Méthode approchée :

Une méthode approchée a pour but de trouver une solution avec un bon compromis entre la qualité de la solution et le temps de calcul.

4.2 Méthode exacte :

Pour trouver la solution optimale, et être certain qu'il n'y a pas mieux, il faut utiliser une méthode exacte, qui demande un temps de calcul beaucoup plus long (si le problème est difficile à résoudre). Il n'existe pas une méthode exacte universellement plus rapide que toutes les autres. Chaque problème possède des méthodes mieux adaptées que d'autres.

5. Conclusion :

Dans ce chapitre nous étudions l'algorithme de sac à dos. Par la suite nous identifions les méthodes de résolutions Dans le chapitre suivant, nous passons à la réalisation de l'application.

Chapitre 2

Application KnapSack

1. Introduction

Après avoir terminé la présentation du problème du sac a dos , nous traitons dans ce chapitre les détails liés au développement de l'application. Nous commençons par décrire les outils de développement et les langages utilisés pour la mise en œuvre. Nous étalerons par la suite les fonctionnalités de l'application en se basant sur quelques captures d'écrans de l'exécution.

2. Les outils et les langages utilisés :

1.1. L'outil utilisé (Android Studio) :

Android Studio est un environnement de développement pour développer des applications Android. Il est basé sur IntelliJ IDEA.

Android Studio permet principalement d'éditer les fichiers Java/Kotlin et les fichiers de configuration XML d'une application Android.

Il propose entre autres des outils pour gérer le développement d'applications multilingues et permet de visualiser la mise en page des écrans sur des écrans de résolutions variées simultanément.[5]

Logo Android Studio

1.2.Langage utilisée (Kotlin) :

Kotlin [6] est un langage de programmation orienté objet et fonctionnel, avec un typage statique qui permet de compiler pour la machine virtuelle Java et JavaScript. Son développement provient principalement d'une équipe de programmeurs chez JetBrains basée à Saint-Pétersbourg en Russie (son nom vient de l'île de Kotline, près de St. Pétersbourg).

Google annonce pendant la conférence Google I/O 2017 que Kotlin devient le second langage de programmation supporté officiellement sur Android[1] après Java.

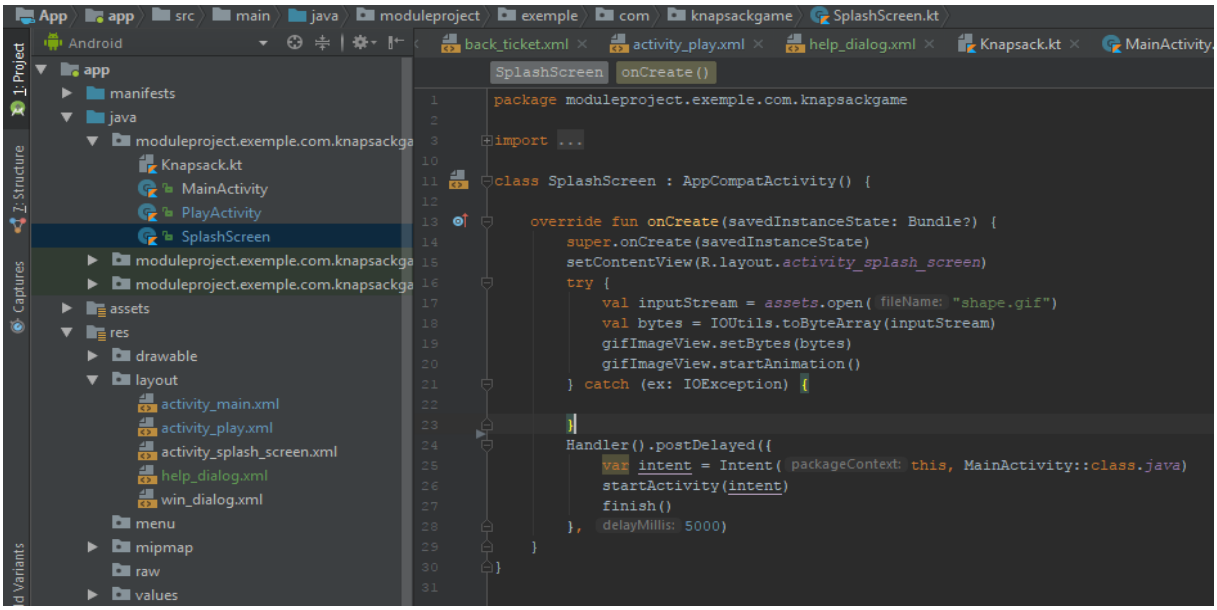


Figure 2.Splash Screen

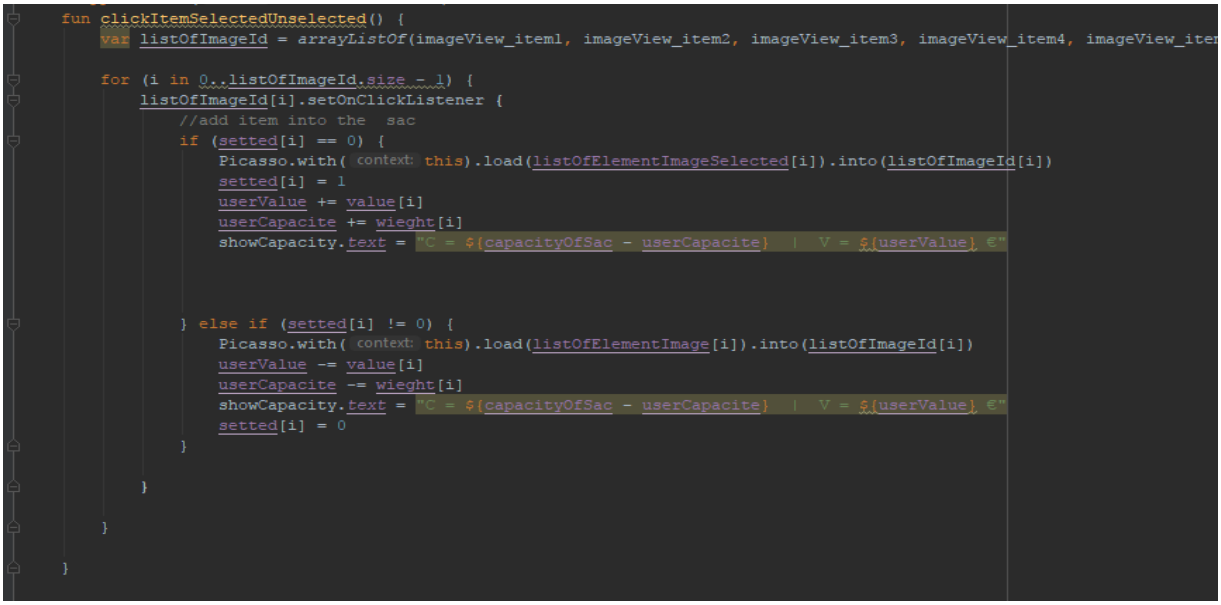


Figure 3.Ajouter des elements

3. Principe

- Le principe du jeu est très clair :

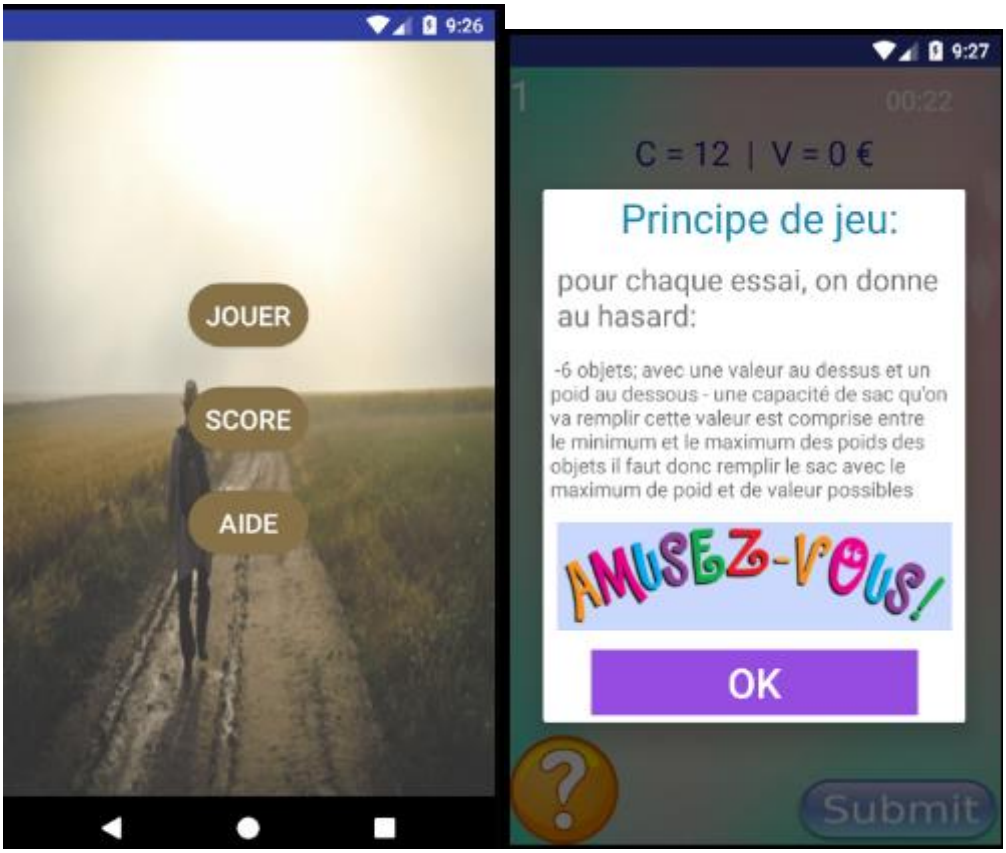
Pour chaque essai, on donne au hasard :

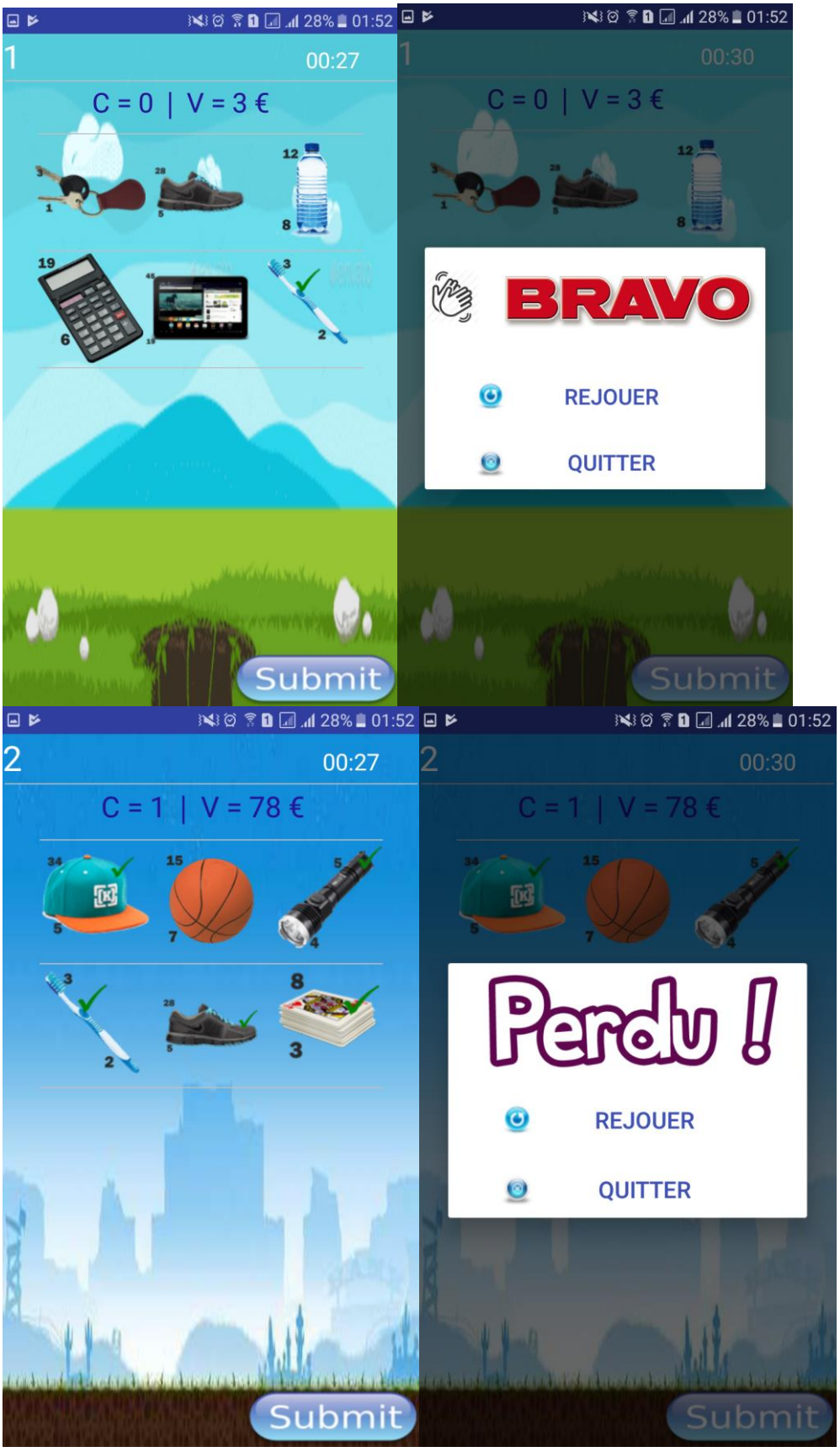
-6 objet ayant chacun une valeur au-dessus et un poids au-dessous.

- une capacité de sac qu'on va le remplir. Cette capacité est comprise entre le minimum des poids des objets et le maximum des sommes des poids des objets (au hassard).

Il faut donc remplir ce sac avec la plus haute valeur possible sans dépasser la capacité donnée.

4. Captures d'écran




1
00:27
C = 0 | V = 3 €
Submit
1
00:30
C = 0 | V = 3 €
BRAVO
REJOUER
QUITTER
Submit
2
00:27
C = 1 | V = 78 €
Submit
2
00:30
C = 1 | V = 78 €
Perdu !
REJOUER
QUITTER
Submit

Conclusions

A travers ce mini-projet, nous avons eu l'occasion d'améliorer notre capacité de travail collectif. Nous avons eu une expérience très importante qui améliore notre acquis d'analyse et de résolution, en découvrant un nouvel algorithme sac à dos et puis l'appliquer dans notre application grâce au logiciel Android Studio.

A l'état actuel, l'application développé en cours de la réalisation de ce mini-projet peut être améliorée par l'ajout d'autre au stage a notre jeux , pour que les enfants adaptent a penser d'une manière optimale

Références :

[1]https://tel.archives-ouvertes.fr/tel-00439824/document

[2]Gilmore P.C and Gomory R.E. Multistagecuttingproblems of two and more dimensions, Operations Research 13 : 94-119, 1965.

[3]Hifi M and Roucairol C. Approximate and exact algorithms for constrained (un)weightedtwo-dimensionaltwo-stagedcutting stock problems, Journal of CombinatorialOptimization, 5 : 465-494, 2001.

[4]http://www.academia.edu/20263835/Probl%C3%A8me_du_sac_%C3%A0_dos

[5] https://www.silicon.fr/kotlin-programmation-android-175165.html?inf_by=5a246631671db80f548b4a23

[6] https://kotlinlang.org