# Liste des projets C++ 2014/2015

28 octobre 2014

Ce document présente une liste non-exhaustive de sujets proposés pour le projet de deuxième année en C++. Les groupes souhaitant traiter un autre sujet sont libres de le faire, après discussion avec moi-même. Globalement, la difficulté sera surtout fonction de la profondeur avec lequel le sujet sera traité, et elle est donc souvent très modulable. Chaque groupe sera constitué de deux personnes. Un étudiant de chaque groupe doit m'envoyer un mail sur durut.matthieu@gmail.com avec les deux noms des gens du groupe, et une liste de souhaits composée de 3 sujets avec un ordre de préférence. A défaut de réponse de votre part au 28 novembre 2014, vous serez groupés aléatoirement.

Je déconseille les projets "finance", peu formatteurs. Par expérience, il faut un bien meilleur niveau et un investissement en temps supérieur pour réussir à fournir un projet "finance" faisant montre de la même technicité que sur d'autres projets.

Chaque groupe sera encadré par un des chargés de TP. Deux "permanences" seront prévues pendant lesquelles vous pourrez prendre rendez-vous avec un encadrant pour obtenir de l'aide sur vos projets. Il vous incombe de vous mettre en relation avec cet encadrant pour vous accorder sur un créneau horaire pour obtenir un suivi.

Au fur et à mesure de l'attribution des sujets, je mettrai à jour un fichier d'attribution des sujets/encadrants sur pamplemousse.

Il est tout à fait autorisé (conseillé même) d'utiliser des bibliothèques tierses (SDL, OpenGL, GSL, Boost, QT, ... ). J'attends cependant qu'une part significative de votre projet soit vôtre. La tricherie est punie par la note minimale.

## 1 Jeux

1. **Shoot'em up** (Difficulté 1 à 2/6) Le but du projet est de développer un jeu de type shoot them up, c'est à dire où des éléments (les "méchants") qui vont apparaître à l'écran doivent être éliminés par un clic de souris suffisamment rapide. Ce projet reprendra l'ébauche de ce qui aura été traité en TD sur le même thème.

2. **Tower Defence** (Difficulté 3/6) Le but du projet est une implémentation du jeu en-ligne.

3. **Puissance4** (difficulté 3/6) Le but du projet est un revival du jeu de "plateau". Le projet peut proposer une IA basée sur une stratégie MiniMax. Une référence sur le MiniMax peut être trouvée ici : nvansteenkiste.googlepages.com/miniMax.pdf

4. **Jeu de dames** (Difficulté 2 à 3/6). Même remarque que pour les autres jeux. La difficulté varie selon l'IA créée ou non.

5. **Bomberman** (difficulté 3/6) Le but du projet est le développement du jeu bomberman. La difficulté du projet est modulable par ajout progressif d'options, évènements, ...

6. **SlimeSoccer** (Difficulté 2/6). Jeu disponible sur ZeBest-3000.com. Il s'agit d'un petit jeu où deux joueurs vont s'affronter dans un pseudo-foot.

7. **BoulderDash** (Difficulté 4/6). Jeu de réflexion (première version 1984). Un mineur doit récupérer des diamants en creusant la terre. On pourra se faire une idée du jeu avec un remake : Treasures Caves.

8. **Initiation à OpenGL** (Difficulté 6/6). Ce projet propose une initiation avec la libraire OpenGL, qui permet de réaliser des applications graphiques 3D. Une bonne expérience préalable de la programmation est nécessaire. Des tutoriaux spécifiques peuvent-être étudiés à profit à l'adresse suivante : nehe.gamedev.net, ou en parcourant le "red book" (Schreiner, Woo, Neider, Davis). Un exemplaire est empruntable auprès de moi-même.

9. **Doom-Like** (Difficulté 7/6). Projet réservé aux personnes ayant préalablement des compétences très solides en C++ et des bonnes notions en OpenGL ou DirectX . Le but du projet est de créer une ébauche de Doom-like en utilisant des "skins" au format .md2 provenant de Quake2. La référence pour ce projet est la page perso de David Henry : http ://tfc.duke.free.fr/.

# 2 Finance

1. **Réplication de portefeuilles** (Difficulté 3/6) Certains portefeuilles financiers complexes sont très lourds à manipuler statistiquement, que l'on souhaite simuler des trajectoires, les pricer ou les hedger. Une stratégie à la mode consiste à projeter (au sens Hilbertien) ce portefeuille complexe sur un espace sensiblement plus petit engendré par des instruments financiers élémentaires. Ce projet propose la réalisation d'un outil de réplication de portefeuille par les payoffs et est relié avec d'autres cours de finance comme économétrie de la finance (3ème année). On pourra au besoin augmenter la difficulté en créant un dialogue entre le programme et une feuille Excel

par exemple.

2. **Pricer** (Difficulté 2/6) Sujet archi-classique qui consiste à créer un programme pour déterminer le prix d'une option financière. Dans la plupart des cas où les formules fermées ne s'appliquent pas, on calculera ces prix par méthode de Monte-Carlo (cf Td correspondant).

3. **Estimation de sensibilités** (difficulté 2/6) Calcul de sensibilités par méthodes de différences finies. Techniques d'accélération de convergence des estimateurs.

4. **Pricing par EDP** (difficulté 3/6) Estimation de prix de produits dérivés par résolution d'EDP dans le cadre Black-Scholes-Merton.

# 3 Applications scientifiques

1. **Classification supervisée** (Difficulté 5/6). L'objectif est de travailler sur la reconnaissance de chiffres écrits à la main, en utilisant le jeu de données ultra-classique NIST (http ://yann.lecun.com/exdb/mnist/ par exemple).

2. **Prévision de séries temporelles par méthodes non-paramétriques (plus proches voisins L2)** (Difficulté 2/6). L'objectif de ce projet est l'introduction à des modèles statistiques non-paramétriques. Les expériences pourront être menées sur des jeux de données publics comme NN5.

3. **Clustering par K-Means** (Difficulté 2/6) Ce projet consiste à développer un algorithme de classification de vecteur en grande dimension pour les regrouper en "paquets" (clusters en anglais) homogènes. Une version multi-thread est aussi envisageable, augmentant sensiblement la difficulté du projet.

4. **Filtre à spam et modèles bayésiens** (Difficulté 4/6) Ce projet consiste à développer un filtre à spam aux moyens de réseaux bayésiens naïfs. Ce projet ne nécessite pas de connaissances particulières mais demande une bonne intuition statistique.

5. **Autour des fractales** (Difficulté 4/6) Ce projet consiste à étudier l'ensemble de Mandelbrot, qui est un sous-ensemble du plan complexe $\mathbb{C}$. Cet ensemble est célèbre pour sa caractéristique "fractale", c'est-à-dire qu'un dessin de l'ensemble est tel que chaque sous-partie du graphe est identique au graphe global. Récursivement, on peut "zoomer" progressivement dans le graphe et toujours observer les mêmes formes. La page wikipedia sur l'ensemble de Mandelbrot illustre bien cette caractéristique. Le but du projet est d'abord de tracer cet ensemble, au moyen par exemple de la bibliothèque de fonctions graphiques SDL. Le calcul de l'ensemble en lui-même est simple. Une fois l'ensemble tracé, il s'agira de permettre à l'utilisateur du programme de se déplacer et de zoomer dans le graphe, pour

observer la propriété fractale. De nombreuses extensions sont ensuite envisageables : la colorisation du graphe, le calcul d'autres ensembles comme l'ensemble de Julia, le Buddhabrot (une fractale dont la forme rappellerait Bouddha lui-même), l'utilisation d'outils de simulation statistique comme l'algorithme de Metropolis-Hastings...

`http://en.wikipedia.org/wiki/Mandelbrot_set`
`http://en.wikipedia.org/wiki/Julia_set`
`http://en.wikipedia.org/wiki/Buddhabrot`
`http://www.steckles.com/buddha/`

6. **Réseaux de neurones** (Difficulté 5/6) Le programme devra permettre de faire fonctionner interactivement un petit réseau neuronal. Par exemple, ce réseau pourra être constitué dŠune zone d'entrée (un carré de 8 cases de côté), de 3 couches intermédiaires et dŠune couche de sortie linéaire de 10 cases.

   Votre réseau devra être implémenté pour un usage concret précis qui est laissé à votre libre choix : classification d'un jeu de données en entrée (mode non supervisé), reconnaissance de motifs (mode supervisé) ou problématiques de modélisations de fonctions (à des fins d'estimation ou de prise de décision).

   L'utilisateur prépare les leçons, listes de couples entrés/sorties dans le cas d'un réseau supervisé ou jeu de données à analyser pour faire converger le réseau (mode non supervisé). Le réseau apprend ces leçons et la progression de son apprentissage devra être visible. Ensuite, l'utilisateur peut introduire de nouvelles entrées et observer le comportement du réseau. Vous pouvez simuler la vieillesse en détruisant des neurones pour voir son impact sur la mémorisation de la leçon. Il peut être également intéressant de prévoir des seuils paramétrables pour éviter le sur-apprentissage ou changer la fonction d'activation.

   Vous trouverez une description générale des réseaux de neurones sur wikipedia pour plus de détails sur leur fonctionnement et les différents usages que l'on peut en faire (http ://fr.wikipedia.org/wiki/Réseau_de_neurones).

7. **Bandits multibras** (difficulté de 2 à 4/6) Un agent évoluant dans un environnement aléatoire doit cumuler un maximum de récompenses en choisissant au fil du temps la meilleure politique, c'est à dire la meilleure réaction possible à ses observations.

8. **Markov Chain Monte Carlo et Metropolis-Hastings** (difficulté 4/6) Utilisation d'algorithmes MCMC, comme Metropolis-Hastings. Applications au problème du voyageur de commerce.

9. **Akinator** (difficulté 6/6) Implémentation d'un algorithme convaincant (et éventuellement de l'interface) de classification pour mimer le jeu en ligne Akinator : http ://en.akinator.com/personnages/jeu. Akinator s'autorise à vous poser 20 questions fermées (réponse = oui ou non) sur une personnalité que vous devez lui faire deviner. On passera par des arbres de classification pour l'algo de prévision.

10. **Correcteur d'orthographe par approche statistique** (Difficulté 5/6) Construction d'un correcteur orthographique par différentes stratégies de Text-Mining (filtres bayésiens, autres classifieurs, distance de Levenshtein, etc.)

11. **Recherche de chemins par l'algorithme A*** (difficulté 3/6) Etant donné un environnement 2D constitué de cases carrées, dont certaines sont impraticables, le but de l'algorithme est de trouver un chemin court entre deux points donnés en utilisant une structure de graphe pour décrire les cases praticables.

12. **Garbage Collector** (difficulté 6/6) Implémentation d'un garbage collector C++ monothread "Stop the world". On pourra dans un premier temps se référrer à la wikipedia : http ://en.wikipedia.org/wiki/Garbage_collection_(computer_science) pour une première introduction aux techniques les plus faciles d'implémentation.

# 4 Autres applications

1. **Recherche de trajet dans le métro** (Difficulté 2 à 3/6) Il s'agit d'implémenter une recherche de trajet optimal (cette notion devant être définie au préalable) au sein d'un graphe. L'application la plus classique de ce type de problématique est la recherche de trajet dans le métro : comment aller d'un point A à un point B en minimisant la durée de trajet / le nombre de correspondances / etc.

2. **Modélisation de propagation d'épidémie dans une ville** (Difficulté 2 à 3/6) On se propose ici d'implémenter des déplacements d'individus dans une ville. La ville elle-même est constituée de rues (lieux de passage), d'établissements (lieux de rencontre) et de zones vides. Une proportion x initiale d'habitants contracte un virus d'épidémie. Il faut définir les règles de propagation du virus et modéliser ensuite sa propagation au sein de la ville. Un objectif intéressant de ce vaste sujet peut être de comparer la vitesse de propagation et/ou de résorbation du virus en fonction des paramètres choisis, et également de comparer avec les résultats fournis par les modèles statistiques d'épidémiologie. Compte tenu des délais du projet, une interface graphique ne sera pas exigée. La qualité de l'IA sera favorisée.

3. **Générateur de malédictions** (Difficulté 2/6) Ce projet consiste à générer aléatoirement des malédictions à partir de "briques élémentaires" (sujets, verbes, COD, COI, etc.)

4. **Modélisation d'une fourmilière** (difficulté 3/6) Le projet consiste en la simulation de l'organisation d'une colonie de fourmis. La simulation s'effecture à l'extérieur de la fourmilière sur la recherche de nourriture et son rapatriement ainsi que sur la défense du lieu de résidence. Plusieurs types d'individus seront implémentés : reines, soldates, récolteuses,

femelles, mâles, oeufs, etc. Le sujet peut être étendu par des guerres entre colonies.

5. **Apprentissage par Renforcement pour Tetris** (Difficulté 6/6) Je recommande un groupe de trois personnes motivées pour ce sujet. Il s'agit de s'inspirer des bases de la théorie d'apprentissage par renforcement pour créer une Intelligence artificielle sur une version simplifiée de Tetris. Le travail pourra largement s'inspirer de : http ://www.math.tau.ac.il/m̃ansour/rl-course/student_proj/livnat/tetris.html

6. **ASCII Art Converter** (Difficulté 4/6) L'art ASCII consiste à réaliser des dessins comme une combinaison de charactères (lettres, ponctuations, etc.). Le but du projet est de réaliser un ASCII Art Converter, par exemple par des techniques de Vector Quantization. On peut se donner une idée de belle réalisation ASCII Art ici : https ://www.youtube.com/watch ?v=Dgwyo6JNTDA

Les projets doivent être réalisés en binôme. Pour les sujets les plus ambitieux, les projets peuvent être réalisés par groupe de 3 étudiants avec mon accord préalable. Avant la soutenance, chaque binôme devra fournir un rapport ainsi que les fichiers sources commentés.

Le rapport détaillera :

1. une description de ce que fait le programme,
2. les problèmes rencontrés et les solutions choisies,
3. une description de l'architecture du programme,
4. un guide d'utilisation du programme (si nécessaire).

Quelques points qui rentreront en compte dans la notation (liste non exhaustive) :

1. Les fichiers sources doivent être présentés clairement et commentés convenablement.
2. Le programme doit compiler et être exécuté sans erreur.
3. Le projet doit pouvoir être ouvert avec Visual Studio, les libraires externes utilisées doivent être automatiquement jointes.
4. Le projet doit respecter la description du sujet donnée plus haut.
5. Un rapport de 10 pages maximum doit être rendu en même temps que le code source du projet.
6. En dehors des suivis, l'autonomie est également appréciée. Les assistants de l'ENSAE ne font pas les projets à votre place...
7. Les projets sont à rendre sous la forme d'un repository Git, hébergé par exemple sur GitHub.

Bon courage !