Sujet 1 : Retrouver un (grand) nombre à partir d'un multiple et de certains de ses chiffres

— Encadrant : Charles Bouillaguet (`charles.bouillaguet@sorbonne-universite.fr`)
— Nombre d'étudiants : 2 ou 3
— Description :

Si on sait que 45420684447526892839533159105846521497309925920929484958214185689521621434478663631127446975601883137031825407528784402330058238493985291643232038581418818289496160207331092823517215925169021514237770639469776177582306224920733458840231643843646372583211005763472461013590501463880337426817066958163756358327 est un multiple de 13217703905079126550753185793722191818605252415566110015450161611346911291512 59 ?, peut-on retrouver les chiffres manquants ?

La question à un intérêt cryptographique : le système de chiffrement/signature RSA est déployé dans toutes les cartes bleues européennes et il repose sur la difficulté calculatoire de trouver un diviseur de grands entiers (disons à 350 chiffres). Mais si jamais on parvenait à apprendre une partie des chiffres du diviseur, par exemple avec des techniques de rétro-ingénierie matérielle appliquées à la carte bleue, le problème deviendrait-il facile ?

En 1985, Rivest et Shamir (le R et le S de RSA) on trouvé une technique qui fonctionne si les 2/3 des chiffres du diviseur sont connus. Ceci a été amélioré par Coppersmith en 1996 : connaître la moitié des chiffres (consécutifs) suffit. La technique de Coppersmith consiste à trouver un polynôme à coefficients entiers qui s'annule sur le nombre recherché. Plus la proportion de chiffres manquant est grande, et plus son degré sera élevé. Trouver des racines de polynômes est "facile" (en tout cas, on dispose d'algorithmes efficaces pour le faire). La fabrication de ce polynôme repose sur l'utilisation d'un algorithme célèbre nommé LLL, qui permet de trouver une combinaison linéaire (à coefficients entiers) de norme faible de vecteurs à coefficients entiers.

Le projet consiste à :

1. comprendre et implanter cet algorithme (donc la partie qui produit le polynôme). Cela peut être soit en python directement (en utilisant la bibliothèque fpylll), soit dans le logiciel libre de calcul formel SageMath qui utilise python.
2. mesurer ses performances et vérifier s'il est capable de faire ce qu'il prétend faire, c'est-à-dire factoriser avec la moitié des chiffres connus des nombres de 3-400 chiffres.

Des notions d'arithmétique modulaire sont souhaitables.

Sujet 2 : Automatisation de la cryptanalyse des cryptosystèmes classiques à l'aide d'algorithmes modernes

— Encadrant : Valérie Ménissier Morain (`valerie.menissier-morain@lip6.fr`)
— Nombre d'étudiants : 2 ou 3
— Description :

Les cryptosystèmes classiques ont été utilisés de l'Antiquité à la seconde guerre mondiale pour protéger aussi bien des affaires privées que des secrets militaires et civils, des affaires d'état, des puissants et des humbles : chiffrements par substitition, par transposition, homophoniques, de Vigenère, de Playfair, ADFGVX, etc. Le cassage d'un texte chiffré par un tel système de chiffrement s'effectuait donc manuellement avec une technique plus ou moins aboutie selon les époques.

Depuis une trentaine d'années un certain nombre de techniques modernes ont été utilisées pour essayer d'automatiser la cryptanalyse de ces cryptosystèmes. Qu'il s'agisse de recuit simulé, d'algorithmes génétiques ou plus récemment de "hill climbing" et autres méta-heuristiques propres à la fouille de donnnées, ces techniques combinent une progresssion plus ou moins guidée et plus ou moins aléatoire avec une fonction d'évaluation de la qualité de la solution ("fitness function").

Le but de ce projet est de systématiser cette exploration d'une part en comparant plusieurs techniques sur un même cryptosystème et d'autre part d'essayer une même technique sur plusieurs systèmes de chiffrement. Le point de départ sera le "hill climbing" appliqué à la cryptanalyse des substitutions mono-alphabétiques en testant plusieurs fonctions d'évaluation et en faisant varier le nombre d'itérations.

Ensuite l'exploration ira en s'élargissant en alternant les phases de familiarisation avec de nouveaux cryptosystèmes et de nouvelles méta-heuristiques, et vous progresserez dans votre comparaison expérimentale de ces techniques sur ces systèmes de chiffrement. Cette seconde phase pourra être exécutée en parallèle selon les affinités de chacun.

Sujet 3 : Challenge Unesco

- Encadrant(e) : Thibaut Lust (`thibaut.lust@lip6.fr`)
- Nombre d'étudiants : 2
- Description :

  Vous êtes un fan de voyage, c'est la fin de l'année scolaire et vous avez trois semaines de vacances devant vous ! Vous décidez de saisir cette opportunité pour réaliser votre rêve : visiter les endroits les plus incroyables de la terre. Votre lieu de départ est connu sous forme LAT/LONG et vous disposez d'un hélicoptère parfait (y compris le pilote) qui voyage à la vitesse constante de 85km/h et possède un réservoir illimité. Vous décidez de vous limiter à la visite de sites UNESCO inscrits au patrimoine de l'humanité, qui sont de trois types : endroits culturels (tels que les centres historiques), les endroits naturels (tels que les parcs nationaux) et les endroits mixtes qui représentent les deux. Le nombre de sites culturels visités doit être égal au nombre de sites naturels visités (à un près, les sites mixtes comptent aussi bien pour naturel que pour culturel). Le challenge est de trouver un itinéraire, partant de n'importe quel endroit, qui va vous permettre de visiter un maximum de sites intéressants, et de revenir à votre point de départ. Le temps maximum de votre voyage est de trois semaines, et vous supposez que vous allez passer 7 heures par site, comprenant la visite, les repas/boissons et le repos (que vous pourrez aussi réaliser dans l'hélicoptère). Vous évaluerez la qualité de votre itinéraire de la façon suivante :
  - Chaque site UNESCO visité compte pour 10 points.
  - Chaque pays différent visité compte pour 20 points.
  - Chaque site qui est listé comme "en danger" rapporte 30 points en plus.

  Votre but est de trouver un itinéraire permettant d'obtenir le score le plus élevé. Les méthodes à utiliser sont basées sur les métaheuristiques et le langage de programmation est le langage C.

Sujet 4 : Algorithmique et calcul haute-performance pour l'algèbre linéaire

— Encadrant : Mohab Safey El Din (`mohab.safey@lip6.fr`)
— Nombre d'étudiants : 2 ou 3
— Description :
Le calcul matriciel est une composante fondamentale de nombreux problèmes scientifiques et ses déclinaisons algorithmiques et logicielles sont intensivement utilisées dans les sciences en général et celles du numérique en particulier (notamment en cryptologie, codes correcteurs d'erreurs, imagerie, etc.). Il est donc essentiel de disposer d'implantations haute-performance pour ces types de calcul, et on attend de celles-ci qu'elles exploitent au mieux les opérateurs arithmétiques les plus récents disponibles dans les processeurs modernes.

Dans le cadre de ce projet, on s'intéressera à l'exploitation de techniques dites de "vectorisation" qui permettent d'effectuer plusieurs calculs arithmétiques en même temps sur des architectures CPU modernes. L'objectif sera d'obtenir, dans un premier temps, un code vectorisé pour le produit de matrices et l'élimination de Gauss. Dans un deuxième temps, on s'intéressera à des implantations haute-performance de l'élimination de Gauss.

Dans les deux cas, on implantera des algorithmes de complexité asymptotiquement meilleure que celle des algorithmes classiques.

Enfin, pour terminer, et si le temps le permet, on s'intéressera à des techniques de parallélisation du code.

Les étudiants auront accès à des serveurs de calculs pour mener leur expérimentation. Ils développeront des compétences en algorithmique mathematique, des techniques de programmation haute-performance (en langage C) et seront initiés à l'usage d'outils collaboratifs comme `git`.

Sujet 5 : Calcul de pseudospectres

— Encadrant : Stef Graillat (`stef.graillat@lip6.fr`)
— Nombre d'étudiants : 2
— Description :
Le ε-pseudospectre d'une matrice A est défini comme le sous-ensemble du plan complexe consistant en toutes les valeurs propres de toutes les matrices situées à une distance ε de A. C'est un outil très utilisé en théorie du contrôle et en automatique pour tester la robustesse de la stabilité d'un système.
Considérons maintenant une matrice $A \in M_n(\mathbb{C})$. Nous notons par $\Lambda(A)$ son spectre. Étant donné $\varepsilon > 0$, le ε-pseudospectre de la matrice $A \in M_n(\mathbb{C})$ est l'ensemble $\Lambda_\varepsilon(A)$ défini par

$$\Lambda_\varepsilon(A) = \{z \in \mathbb{C} : z \in \Lambda(X) \text{ avec } X \in M_n(\mathbb{C}) \text{ et } \|X - A\|_2 \leq \varepsilon\}.$$

On peut montrer que

$$\Lambda_\varepsilon(A) = \{z \in \mathbb{C} : \sigma_{\min}(A - zI) \leq \varepsilon\},$$

où $\sigma_{\min}$ représente la plus petite valeur singulière. Cela donne un algorithme de calcul du pseudospectre connu sous le nom de `GRID`.
Cet algorithme est massivement parallèle. En effet, il revient à calculer de manière indépendante une SVD (décomposition en valeurs singulières) de $A - zI$ pour chaque point z de la grille. Un tel algorithme devrait donc pleinement tirer parti des architectures parallèles. Néanmoins, il nécessite beaucoup de calcul de valeurs singulières en des points qui ne font pas partie du pseudospectre. Une méthode basée sur un algorithme de prédiction-correction (suivi de trajectoire) a été proposée pour pallier ce problème.

Le travail pourra se dérouler de la manière suivante.

1. Étude théorique des pseudospectres et de la décomposition en valeur singulière (complexité, algorithme de calcul en particulier).
2. Implantation de l'algorithme `GRID` en Python. Proposer une version parallèle de cet algorithme.
3. Implantation en Python de l'algorithme de prédiction-correction. Comparaison en terme de performance et de parallélisation avec `GRID`.
4. Étendre ces algorithmes à la notion de pseudospectres par composante.

Sujet 6 : Génération de problèmes éthiques par construction graphique

— Encadrant : Gauvain Bourgne (`gauvain.bourgne@lip6.fr`)
— Nombre d'étudiants : 2 ou 3
— Description :
Ce sujet s'inscrit dans le cadre de la mise en place d'une plate-forme de comparaison de modèles de raisonnement éthiques (mécanismes permettant de déterminer si une décision dans un contexte donné est jugée admissible ou non selon différents principes éthiques). Il s'agit de faciliter la construction et la représentation lisible de dilemmes éthiques en permettant de jouer sur certains de leurs paramètres pour illustrer et comparer différentes approches. Il s'agit en premier lieu de définir un format de représentation, dont la partie factuelle indiquant les déroulés possibles, sera formalisée sur la base de standard de la planification tel que le langage PDDL (Planning Domain Definition Language. La génération de dilemme à proprement parler se fera de manière graphique. Dans un premier temps, on se concentrera sur des variantes du classique dilemme du trolley, en permettant à l'utilisateur de multiplier et recombiner les différents éléments (voies, personnes sur celles-ci, embrayages...) sur un schéma. Ce schéma devra alors être traduit automatiquement dans le formalisme préalablement défini, et l'option devra être donnée de simuler les différents mécanismes de raisonnement éthiques existants pour indiquer leur recommandations. Il sera ensuite possible de monter d'un cran d'abstraction en permettant de représenter graphiquement les chaines causales pour simuler n'importe quel type de problème.

Sujet 7 : Implantations efficaces de calculs sur les polynômes à une variable : FFT

— Encadrant : Vincent Neiger (`vincent.neiger@lip6.fr`)
— Nombre d'étudiants : 2 ou 3
— Description :
Les polynômes sont un objet mathématique fondamental, qui apparaît de manière omniprésente dans de nombreux problèmes et applications provenant de contextes scientifiques variés. Par conséquent, les calculs sur les polynômes (addition, multiplication, division, PGCD, solutions d'équations...) forment une composante essentielle de solutions algorithmiques et logicielles conçues pour répondre aux besoins de ces contextes d'utilisation.

Les sciences du numérique fournissent un champ naturel d'applications pour ce type de calculs, par exemple concernant la cryptologie et les codes correcteurs d'erreurs. Par ailleurs, ces domaines demandent souvent de résoudre des instances dont la taille est particulièrement importante. Il est donc essentiel de disposer d'implantations haute-performance pour ce type de calculs. Ces implantations doivent se concentrer sur les meilleurs algorithmes connus, et exploiter au mieux les techniques les plus récentes disponibles via les processeurs modernes.

Dans le cadre de ce projet, on s'intéressera à la FFT (Fast Fourier Transform), sous-routine principale des algorithmes rapides pour la multiplication et autres opérations plus complexes sur les polynômes. On s'attachera à rendre le code efficace du point de vue des accès à la mémoire, en se basant sur le principe de localité temporelle et spatiale des données. On exploitera des techniques de "vectorisation" qui permettent d'effectuer certains calculs en parallèle sur des vecteurs de données.

Les étudiants auront accès à des serveurs de calculs pour mener leurs expérimentations. Ils développeront des compétences en algorithmique mathematique, des aspects de programmation haute-performance en langage C/C++, et seront initiés à l'usage d'outils collaboratifs comme Git.

Sujet 8 : Algorithmes de budget participatif

— Encadrant : Fanny Pascual (`fanny.pascual@lip6.fr`)
— Nombre d'étudiants : 2 ou 3
— Description :
De plus en plus en plus de communes, dans le monde entier, consacrent une part de leur budget à des projets proposés par les habitants, et plébiscités par ceux-ci. Le but de ce projet est d'implémenter des algorithmes de budget participatif.

Un algorithme de budget participatif prend en entrée une liste de projets proposés par les habitants, les coûts de chacun de ces projets, les préférences des votants sur les projets, et le budget disponible B. Il retourne alors une affectation du budget disponible aux projets proposés. On peut distinguer deux variantes.
— Dans la première, un projet est soit pas financé soit totalement financé. L'algorithme retourne alors un sous-ensemble des projets dont la somme des coûts est au plus B.
— Dans la seconde, un projet peut être partiellement financé. L'algorithme retourne alors une affectation du budget aux projets, de manière à ce que la somme des fonds affectés aux projets soit d'au plus B.
Une première étape à ce projet consistera à implémenter un algorithme de résolution du problème du sac-à-dos, qui, étant donné un ensemble d'objets ayant chacun une valeur et un poids, et une capacité B, retourne un sous-ensemble d'objets de poids au plus B et de valeur total maximale.

Une deuxième étape consistera à implémenter, et éventuellement à proposer, divers algorithmes de budget participatif.

Une interface graphique, permettant de voir les préférences des votants et le résultat de l'algorithme (projets sélectionnés) sera appréciée. Selon les envies et les connaissances des étudiants, une application web peut également être proposée.

Pour en savoir plus sur les algorithmes de budget participatif : `https://en.wikipedia.org/wiki/Participatory_budgeting_algorithm`

Sujet 9 : Algorithme de Berlekamp – Massey

— Encadrant : Jérémy Berthomieu (`jeremy.berthomieu@lip6.fr`)
— Nombre d'étudiants : 2
— Description :
Soit $\mathbf{u} = (u_i)_{i \in \mathbb{N}}$ une suite que l'on sait récurrente linéaire. Le but du projet est de déterminer un algorithme afin de retrouver la relation de récurrence minimale vérifiée par cette suite. On

étudiera ensuite l'algorithme (quasi-)optimal de Berlekamp - Massey introduit par Berlekamp en 1968 et par Massey en 1969.
En pratique, l'algorithme de Berlekamp – Massey est utilisé en théorie des codes correcteurs d'erreurs pour décoder les codes BCH, utilisés entre autres dans les CD, DVD, SSD ou encore certains codes-barres bidimensionnels.
Plus précisément, étant donné un message représenté sous la forme d'un vecteur de $\mathbb{K}^k$, où $\mathbb{K}$ est un corps, la théorie des codes correcteurs permet d'anticiper les erreurs apparaissant au cours de sa transmission en y ajoutant de la redondance. En général, on considère alors les messages comme des vecteurs (dits *mots du code*) d'un sous-espace vectoriel C de $\mathbb{K}^n$ de dimension $k \leq n$. Après transmission du message, il *suffit* de trouver le vecteur de C le plus proche de celui reçu pour corriger (ou *décoder*) les erreurs de transmission.
Une implantation des ces algorithmes sera effectuée en C.

Sujet 10 : Jeu de société Quoridor : implémentation du jeu et d'IA joueuses

— Encadrant : Parham Shams (`parham.shams@lip6.fr`)
— Nombre d'étudiants : 2 ou 3
— Description :
Quoridor est joué sur un plateau de 81 tuiles carrées (9x9). Chaque joueur y est représenté par un pion qui part de la tuile centrale du côté du plateau situé face à lui (dans un jeu à deux, les joueurs se font face). L'objectif est d'être le premier joueur à atteindre n'importe laquelle des neuf tuiles situées sur le bord opposé du plateau. Le jeu Quoridor comporte aussi vingt murs. Ce sont des plaquettes de bois longues de deux tuiles qui peuvent être placées dans l'interstice entre les tuiles. Les murs restreignent le déplacement de tous les pions, qui doivent les contourner. Quand vient son tour de jouer, un joueur peut choisir soit de déplacer son pion d'une case (dans toutes les directions), soit (si possible) de placer un mur. Les murs sont plaçables entre deux paires de tuiles. Ils ne peuvent se chevaucher. Ils ne doivent pas non plus interdire l'accès aux lignes d'arrivées des pions adverses.

Le but de ce projet est donc d'implémenter le jeu dans un premier temps avant de s'intéresser à implémenter des IA joueuses. Plus précisément on pourra commencer par des IA ayant un comportement random, puis s'intéressant seulement à la recherche d'un plus court chemin (via l'algorithme A*). On pourra ensuite selon l'envie des étudiants aller plutôt vers des algorithmes minimax, alphabeta, Monte-Carlo ou encore de l'apprentissage voire des algorithmes génétiques.
Il est recommandé d'avoir une connaissance du langage Python.

Sujet 11 : Continuous-Time Bayesian Network (CTBN) : sampling & learning

— Encadrant : Pierre-Henri WUILLEMIN (`pierre-henri.wuillemin@lip6.fr`)
— Nombre d'étudiants : 2 ou 3
— Description :
Les réseaux bayésiens (BNs) sont un modèle probabiliste qui s'appuie sur un graphe (orienté sans cycle) pour représenter une distribution jointe d'un grand nombre de variables aléatoires.Ce modèle à la fois numérique (distribution) et qualitatif (graphe) est un point de contact intéressant entre probabilités, statistiques et intelligence artificielle. Il permet d'implémenter des outils de raisonnement, de calcul de fiabilité, d'explications causales, mais aussi d'apprentissage statistiques et des outils de classification (machine learning, etc.).

Ce projet s'inscrit dans la continuité d'un projet de l'année dernière. Son but est d'étudier des algorithmes basés sur un modèle issus des BNs permettant la représentation de processus stochastique en temps continu (représentant des chaînes de Markov factorisée, en temps continu). Les différentes tâches de ce projet seront :
1- état de l'art sur le domaine sur le modèle [2002], l'inférence approchée [2008] et son apprentissage [2020]
2- prise en main et améliorations d'une implémentation (issue du projet précédent) du modèle CTBN dans la librairie aGrUM (http ://agrum.org) qui propose un grand nombre des composantes de base pour l'implémentation d'un tel modèle en python
3- étude et implémentation d'un algorithme de sampling efficace de CTBN
3- étude et implémentation d'un algorithme d'apprentissage de CTBN
4- productions de quelques notebooks (jupyter) de présentation des résultats du projet
Bibliographie :
[2002] U. Nodelman, C. R. Shelton, and D. Koller. Continuous Time Bayesian Networks. In Proceedings of the Eighteenth Conference on Uncertainty in Artificial Intelligence, pages 378–387, 2002. [2008] El-Hay, T., Friedman, N., & Kupferman, R. (2008). Gibbs Sampling in Factorized Continuous-Time Markov Processes. ArXiv, abs/1206.3251. [2020] Alessandro Bregoli, Marco Scutari and Fabio Stella Constraint-Based Learning for Continuous-Time Bayesian Networks, in PGM'20

Sujet 12 : Logiciel de QCMs personnalisés pour l'apprentissage des drapeaux nationaux

- Encadrant : Nawal Benabbou (`nawal.benabbou@lip6.fr`)
- Nombre d'étudiants : 2 ou 3
- Description :
  L'objectif premier de ce projet est de créer un logiciel avec une interface graphique plaisante, permettant d'aider l'utilisateur à apprendre des drapeaux nationaux par le biais de QCMs. Pour accélérer son apprentissage, les questions posées ne doivent pas être choisies au hasard. À la place, il convient de tenir compte de ses réponses aux précédents QCMs pour pouvoir se concentrer sur ses erreurs. Pour ce faire, il faut stocker dans un fichier toutes les réponses de l'utilisateur après chaque QCM réalisé. Ces réponses doivent ensuite être analysées avant chaque nouveau QCM à réaliser, en utilisant par exemple une mesure de similarité qui aidera à définir les prochaines questions à lui poser. Il est important de souligner que ces calculs doivent être réalisés de manière efficace pour limiter le temps d'attente de l'utilisateur. Par ailleurs, l'utilisateur de ce logiciel doit pouvoir choisir la difficulté des questions (nombre de réponses possibles), le type de questions (drapeau vers pays ou pays vers drapeau) et doit pouvoir suivre ses statistiques. En seconde partie de ce projet, on s'intéressera à étendre les possibilités du logiciel en permettant à l'utilisateur d'apprendre d'autres choses, comme par exemple le nombre de frontières à traverser pour pouvoir passer d'un pays à un autre. Les étudiants pourront proposer d'autres extensions qu'ils estiment pertinentes.