Projet 1A : Synthèse de musique électronique

Projet réalisé par :
Jules Dasseux
Sami Moustachir
Oualid Hariss
David De Souza

Table des matières

Introduction

Suite à l'essor de l'informatique et au développement d'internet, la sécurité informatique est devenue une question majeure dans ce domaine. Que ce soit pour la vie privée ou pour les entreprises, la sécurité des données et l'intégrité des machines est un point crucial pour une utilisation seraine de ces outils.

Afin de défendre nos ordinateurs et autres outils informatiques contre les programmes malveillants, des logiciels permettant de détecter et contrer ces entités indésirables ont été mis au point. Les développeurs de virus informatique exploitant sans cesse les nouvelles failles de sécurité de nos systèmes et créant de nouveaux virus, ces logiciels de défense doivent constament être mis à jour et devenir plus performants.

C'est dans ce cadre que s'inscrit notre projet. Afin de détecter si un logiciel est un virus ou non, nous avons transcrit son code en musique. L'idée est que, comme chaque compositeur a son style, chaque codeur possède une manière propre d'écrire des virus. En les transcrivant en musique, nous pourrions donc déceler si le logiciel douteux est apparenté à un autre virus ou non.

Dans ce rapport, nous allons donc vous présenter notre projet. Tout d'abord nous reviendrons sur quelques définitions afin de mieux contextualiser notre projet, puis nous présenterons le programme réalisé, et finalement nous reviendrons sur l'organisation du projet.

I) Mise en contexte et définitions

a) Langage assembleur

En informatique il existe différents langages, s'étalant sur différents niveaux entre la machine et l'homme, qui permettent aux programmeurs de définir des instructions que l'ordinateur doit réaliser pour le fonctionnement des différents programmes qu'il doit exécuter.

Au plus bas niveau se trouve le langage machine. Difficilement lisible, il est uniquement composé d'instructions en binaire (c'est-à-dire soit des 0 soit des 1). La forme d'encodage des instructions dépend alors du processeur sur lequel elles vont être exécutées.

Le langage assembleur lui se situe juste au-dessus. Très proche du langage machine, il permet de traduire de manière plus compréhensible les instructions du langage binaire. Au même titre que pour le langage machine, il va dépendre du processeur considéré. Ainsi, une même instruction ne sera pas transcrite de la même manière en fonction du processeur.

Lors de l'exécution d'un programme, celui-ci va donc faire réaliser au processeur un certain nombre d'opérations élémentaires, que l'on peut lire et identifier une par une grâce au langage assembleur.

Ces opérations vont se classer selon leur type, en plusieurs catégories. Par exemple nous pouvons retrouver les instructions d'arithmétique binaire, les instructions logiques ou encore les instructions de contrôle des données.

b) Virus informatique

Un virus informatique est un programme, souvent assez léger, qui se loge au sein d'un autre programme. Lorsqu'il s'enclenche, le virus va exécuter les instructions prévues par son concepteur. De manière générale, le virus infecte d'autres programmes afin de se répliquer. C'est cette propriété qui leur a donné leur nom, par analogie avec la médecine, leur véritable nom étant CPA, acronyme de Code Auto-Propageable.

Souvent le virus va avoir un comportement nuisible. D'une simple gène lors de l'utilisation de l'ordinateur à la destruction de données et la prise de contrôle de systèmes, le champ d'applications et les effets d'un virus sont très variables.

Pour cette raison les virus peuvent être classés selon leur mode de propagation. On peut alors trouver : les vers (qui se propagent via un réseau), les chevaux de Troie (qui créent des brèches dans la sécurité des systèmes) et les bombes logiques (qui s'enclenchent suite à un événement défini).

Lors de l'infection, le virus s'insère dans une application hôte. Lors de sa réplication, afin d'éviter d'infecter à nouveau un même fichier, le virus va inclure au code de l'application une clé d'identification qui lui permettra de reconnaitre les fichiers déjà infectés des autres. Cette clé s'appelle la signature virale.

c) Antivirus

Afin de lutter contre les virus, des programme appelés antivirus ont été développés. Leur rôle est d'identifier et, dans la mesure du possible, de neutraliser les virus.

Afin de détecter les virus, les antivirus vont se fonder sur trois axes principaux de recherche. Tout d'abord, les signatures virales. En effet, en détectant ces signatures dans les programmes que l'antivirus scanne, les fichiers infectés peuvent être identifiés. Cependant, cette méthode a des limites. En effet, cela nécessite que la base de données de l'antivirus sur ces signatures soit à jour. De plus, cela ne fonctionne pas avec les virus encore non détectés (c'est-à-dire nouveaux) et certains virus sont maintenant capable de camoufler voir de rendre illisible leur signature (ces virus sont dits polymorphes).

Un autre moyen est de controler l'intégrité des fichiers en vérifiant certaines de leurs caractéristiques, comme la taille ou les dates de modifications. Ainsi, si un fichier vient à être anormalement altéré, l'antivirus prévient l'utilisateur. Là encore, de nombreux virus passent ce controle, en ne modifiant pas la taille ou la date d'utilisation aux yeux de l'antivirus.

Finalement vient la méthode heuristique. En analysant le fonctionnement des applications, si l'antivirus détecte un comportement proche de celui d'un virus connu, une alerte est émise. Pouvant avoir des résultats très puissants, cette méthode permet parfois même de détecter denouveaux virus. Cependant, il peut arriver que de fausses alertes se déclenchent.

Après avoir détecté la menace, l'antivirus va tenter de la neutraliser. Pour cela, il peut tenter de supprimer le code du virus dans le fichier infecté, voir complètement supprimer le fichier. Le fichier peut également être mis en quarantaine, c'est-à-dire être déplacé vers une zone du disque dur sous le contrôle de l'antivirus et où il ne pourra pas être exécuté.

II) Notre projet

a) Principe

Afin de chercher d'autres moyens de détecter des virus, nous allons analyser le code résultant de l'exécution d'un virus. En récupérant ces instructions en langage assembleur, nous allons tenter de les transcrire en musique. L'idée étant que des virus parents devraient avoir des codes proches, les instructions utilisées devraient également être semblables, et en les transcrivant en musique, ils devraient avoir des sonorités proches. De plus, tout comme chaque compositeur possède un style qui lui est propre, chaque hackeur va également posséder ses propres habitudes en matière de codage, et il serait donc potentiellement possible d'apparenter des virus et de leur reconnaitre ainsi une même source.

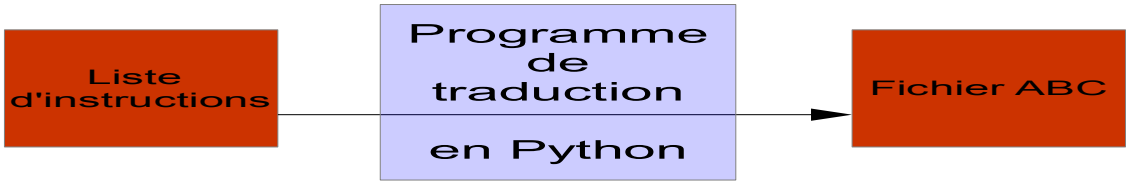

Illustration 1: Schéma du principe du projet

Nous avons ainsi créé un programme qui traduit des instructions (en langage assembleur) en musique. Le programme prend ainsi en entrée un fichier texte contenant une liste d'instructions récuppérées lors de l'exécution d'un programme et crée en retour un autre fichier texte au format ABC (format décrivant une manière d'écrire la musique par ordinateur). Grâce à un second programme, nous pouvons ensuite écouter ce fichier.

Afin de pouvoir dicerner chaque instruction, nous avons décidé d'associer à chacune une note et un instrument différents. De plus, chaque famille d'instruction s'est vue attribuer des instruments proches. Par exemple, les instruments utilisés pour les instructions de contrôle des données sont le violon, le violocelle et la contrebasse.

b) Résultats

En janvier, nous avions déjà aboutit à un premier programme. Celui-ci traduisait de manière aléatoire les instructions en notes. Suite a une première lecture, il créait une liste de traduction des instructions, puis avec cette liste il créait une partition en abc. De ce fait, la partition obtenue à partir d'un même fichier changeait à chaque lecture par le programme. De plus, le seul instrument utilisé était le piano.

Le programme final lui ne souffre plus de ces problèmes, grace au dictionnaire de traduction des instructions. Plusieurs instruments ont été implémentés. Le fichier en abc peut ensuite être converti en fichier MIDI grâce à l'application libre abc2midi. Nous avons pu ainsi traduire la liste d'instructions d'un virus en musique.

c) Perspectives

Afin d'améliorer le programme, nous avons envisagé plusieurs pistes.

Pour le rendre plus simple d'utilisation, une interface graphique serait souhaitable. Elle permettrait de rendre le programme plus ergonomique et mieux compréhensible.

D'autres informations du code fourni pourraient éventuellement être utilisées, comme par exemple les adresses mémoires.

Le programme pourrait éventuellement resortir la partition abc directement au format MIDI. Ceci en simplifierait également l'utilisation, puisqu'alors il ne serait plus nécessaire de faire appel à un autre programme.

III) Organisation du projet

a) Mise en place du groupe

Dès le début nous nous sommes répartit les tâches selon deux axes : un axe déchiffrage du code et un axe écriture de la partition. Le premier groupe devait s'occuper de la partie du programme qui s'intéressait à l'exploitation du code fourni en entrée du programme. Le but était d'identifier les données exploitables et de choisir une structure de données pour la base du programme en python. Le second groupe a eu pour mission de s'intéresser à la partie du programme qui écrit la partition en ABC. Ici, il fallait voir comment se manipule le langage ABC et trouver des manières satisfaisantes d'écrire le programme.

Afin de pouvoir communiquer sur l'avancement du projet, nous avons également créé un groupe sur facebook.

Nous avons également plannifié le projet avec des objectifs à atteindre, selon la chronologie suivante : en novembre, familiarisation avec le langage ABC, en janvier, écriture d'un premier programme simple, puis en mai/juin, d'un programme final.

b) Déroulement du projet

Au début du projet, nous avons effectué peu de réunions, la première phase consistant en un apprentissage de l'écriture ABC. Puis nous avons commencé à communiquer plus, via le groupe facebook et en réunions. Nous n'avons pas fait de comptes rendu pour chaque rencontre, mais nous notions à la fin des réunions les objectifs fixés sur la page du groupe.

Nous avons été confrontés à la difficulté de la gestion d'emplois du temps différents. Il était parfois difficile de trouver de bons horaires pour les réunions, ce qui explique que celles-ci n'aient pas été régulières.

Nous avons effectué une réunion par skype pendant les vacances de décembre. Nous nous sommes également servi de cet outil pendant une autre réunion, à laquelle un membre du groupe ne pouvait assister.

Pour la création du programme, nous avons initalement utilisé un googledoc pour l'écrire, puis nous l'avons ensuite mis au format python.

De manière générale, nous avons pu tenir les objectifs que nous nous étions fixés initialement. En revanche, nous n'avons pas pu implémenter dans le programme certaines des idées supplémentaires que nous avons eu durant l'année.

Conclusion

A travers ce projet, nous avons pu constater que de nombreux domaines sont en jeu dans la réalisation d'un programme, tant sur le plan technique que sur le plan humain.

D'une part, nous avons pu nous familiariser avec certains concepts et applications en informatique. En effet, la création d'un programme nous aura montré qu'un solide socle de connaisances est nécessaire avant de pouvoir commencer l'écriture, mais il faut également prévoir une architecture globale afin de pouvoir subdiviser les tâches et les rendre plus simples à traiter.

D'autre part, nous avons pu nous atteler à la gestion d'un projet. Les aspects organisationnels nous ont montré que, même sur un groupe restreint, des difficultés peuvent surgir, et qu'il faut pouvoir s'adapter afin de les surmonter.

Dans l'ensemble, cette expérience fut enrichissante, tant sur le plan professionnel que personnel, car nous avons pu voir l'aboutissement de notre projet.

Bibliographie

Site internets :

- http://www.commentcamarche.net/contents/15-introduction-a-l-assembleur, *site consulté le 11/06/2014.*
- http://www.commentcamarche.net/contents/1235-virus-informatique, *site consulté le 11/06/2014*
- http://docs.oracle.com/cd/E19253-01/817-5477/817-5477.pdf, *x86 Assembly Language Reference Manual, site consulté le 07/06/2014*
- http://abc.sourceforge.net/standard/abc2-draft.html, *documentation sur le langage ABC, site consulté le 07/06/2014*
- http://ifdo.pugmarks.com/~seymour/runabc/abcguide/abc2midi_guide.html, *documentation sur l'application abc2midi, site consulté le 07/06/2014*