TP3 IBI - Password Cracking

Ce programme a pour but de deviner un mot de passe à l'aide d'un algorithme génétique.

Auteurs :
Benoît ALCARAZ - 11608160
Louis GUILLOTIN - 11603130

Guide d'utilisation :
Le programme doit être placé dans le même dossier que l'exécutable unlock64.exe (ou équivalent en fonction du système d'exploitation) et le fichier config.conf.

Renseignez le numéro étudiant dans le champs student_id du fichier de configuration ainsi que les paramètres voulus.

Choix d'implémentation :

Chaque pas de la boucle suit le cycle suivant :
* step_run()
* step_generate_rank()
* get_best_children()
* step_mutate()
* heavy_mutation()

step_run() :
Simple fonction exécutant le programme "unlock" et attribuant le score résultant à chaque agent.

step_generate_rank() :
Genère une nouvelle population à partir du classement en fonction du score des 10% des meilleurs individus seulement.

Le choix de la génération à partir du rang plutôt que du score a été fait car l' écart de valeur entre les scores des agents n'est pas assez important. Seulement les 10% des meilleurs individus sont utilisés dans cette fonction car les agents n'appartenant pas à cet échantillon présentent souvent des scores trop faibles comparativement aux leilleurs éléments.

get_best_children() :
Cette fonction utilise les 10% meilleurs individus de la population afin de générer autant de nouveaux agents.

Le processus de couplage consiste à faire un individu d'une taille égale au minimum entre celles des deux parents et de choisir un character à la position i de l'un ou de l'autre avec une probabilité de 50%.

step_mutate() :
Fonction effectuant la mutation sur la population courante. Une mutation peut consister en l'une des actions suivantes :
* Ajouter un nouveau caractère à une position aléatoire.
* Retire un caractère à une position aléatoire.
* Inverser la position de deux caractères.
* Déplacer un caractère.
* Modifier un caractère.

heavy_mutation() :
Cette fonction applique d'importantes mutations (une série de 4 mutations à ajouter à la mutation classique) sur 5% de la population afin d'assurer l' hétérogénéité des agents.

Informations complémentaires :
À chaque itération, le meilleur élément est sauvegardé et réaffecté à la population suivante. Cela permet de s'assurer que l'algorithme ne régressera pas.

Résultats :

Le programme met en moyenne 200 itérations pour parvenir à la bonne solution avec une population de 100 agents. Le plus grand nombre de "restart" observé a été de 4 ce qui fait un maximum de 1200 itérations observé. Le minimum d' itérations observé a été de 64.