Maxime KUOCH
Fabien LAMBERT--DELAVAQUERIE

UNIVERSITÉ
PARIS-EST
MARNE-LA-VALLÉE

PROJET DE PROGRAMMATION C : PROFILAGE

Maxime KUOCH
Fabien LAMBERT--DELAVAQUERIE

Table des matières

Maxime KUOCH
Fabien LAMBERT--DELAVAQUERIE

Présentation du programme

Composition

Répertoire bin : contient tous les fichiers objets .o et l'exécutable myprofiler créé

Répertoire include : contient tous les header .h du projet

Répertoire src : contient tous les fichiers sources .c

Répertoire doc : contient différents fichiers .log déjà testés créés par différents groupes tels que LOPES_LIN_1.log (groupe de LOPES MENDES Ailton et LIN Gerald), WADAN.log (groupe de WADAN Samy),profile_LIEGEY_CROHARE.log (groupe de LIEGEY Armand et CROHARE Jeanne) par exemple ainsi que ce rapport

Répertoire html créé par la documentation doxygen

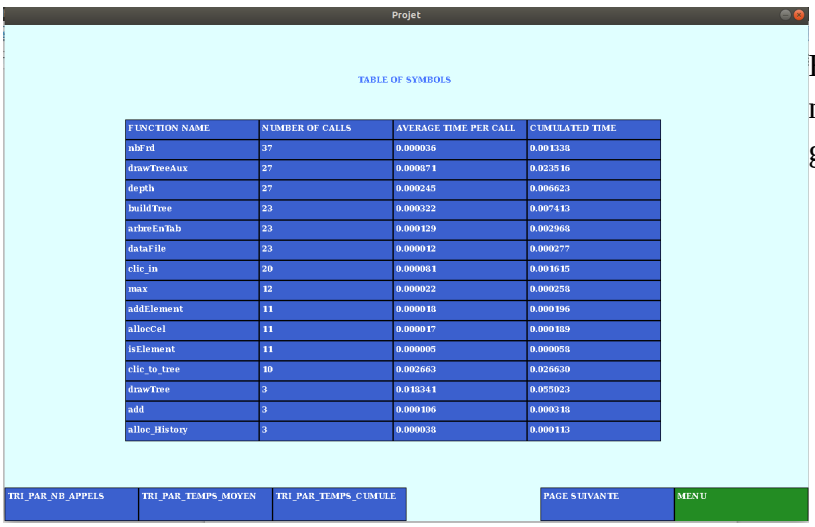
Fichier makefile qui compile le projet

Fichier KUOCH_LAMBERT.log créé par la macro de profilage

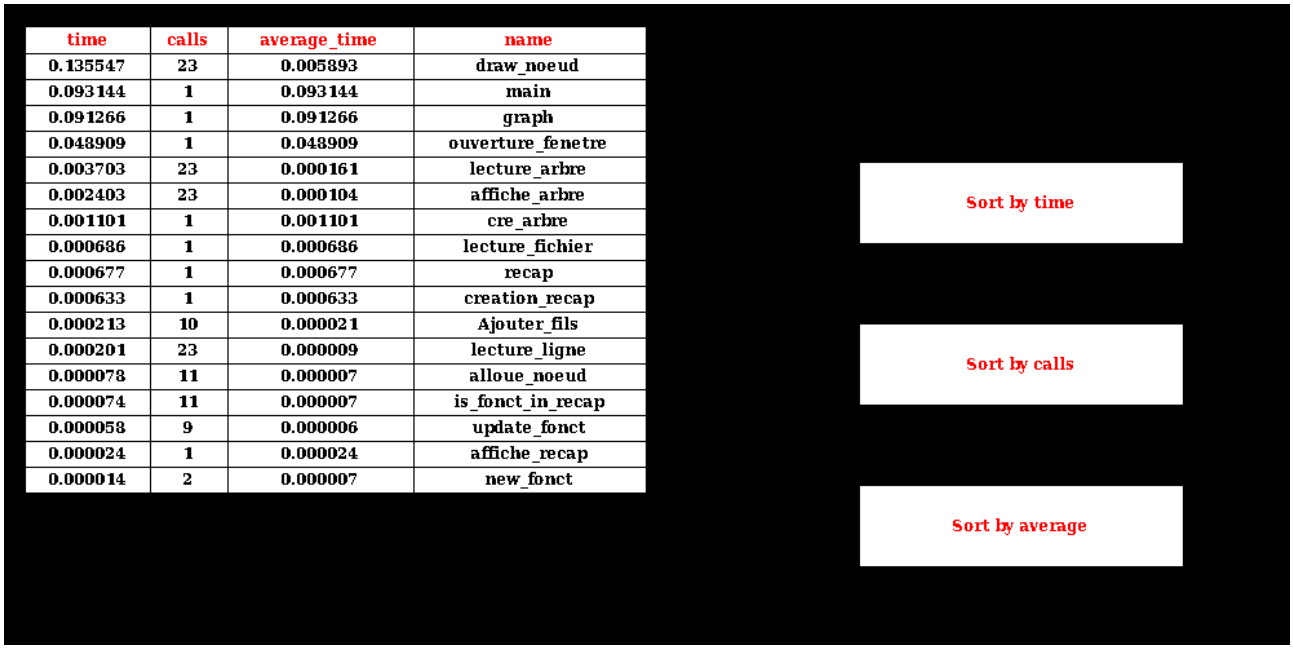
Fichier log_dev extrait depuis redmine qui correspond aux logs du projet

Pour les tests, nous avons donc utilisé les .log fournis par les groupes de LOPES MENDES Ailton et LIN Gerald, le groupe de WADAN Samy et le groupe de LIEGEY Armand et CROHARE Jeanne.

Nous leur avons d'ailleurs envoyé des .log créés par notre exécutable.



Exemple du tableau recap trié par nombre d'appels du .log du groupe LIEGEY CROHARE

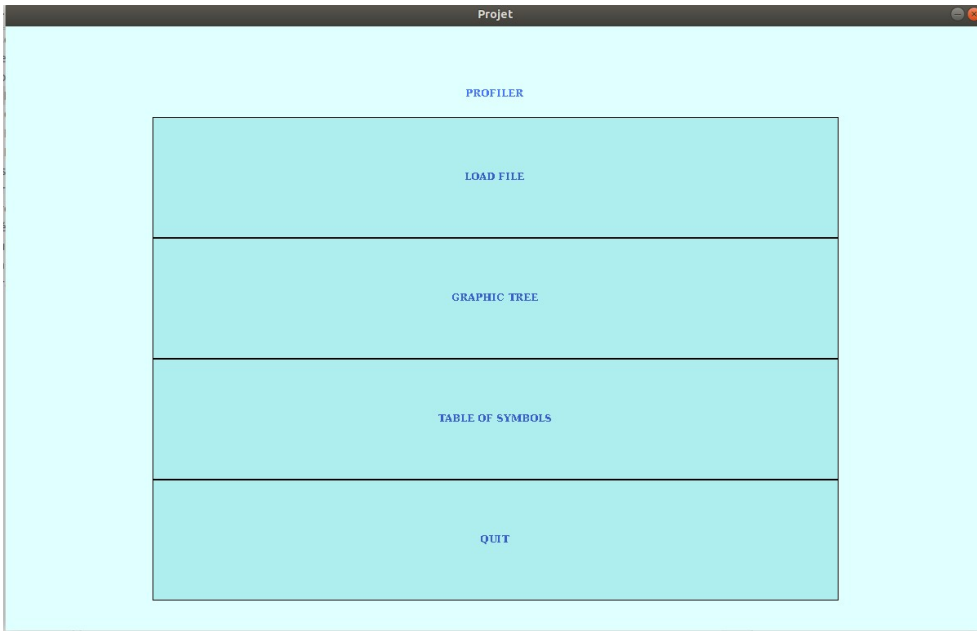


time	calls	average_time	name
0.135547	23	0.005893	draw_noeud
0.093144	1	0.093144	main
0.091266	1	0.091266	graph
0.048909	1	0.048909	ouverture_fenetre
0.003703	23	0.000161	lecture_arbre
0.002403	23	0.000104	affiche_arbre
0.001101	1	0.001101	cre_arbre
0.000686	1	0.000686	lecture_fichier
0.000677	1	0.000677	recap
0.000633	1	0.000633	creation_recap
0.000213	10	0.000021	Ajouter_fils
0.000201	23	0.000009	lecture_ligne
0.000078	11	0.000007	alloue_noeud
0.000074	11	0.000007	is_fonct_in_recap
0.000058	9	0.000006	update_fonct
0.000024	1	0.000024	affiche_recap
0.000014	2	0.000007	new_fonct

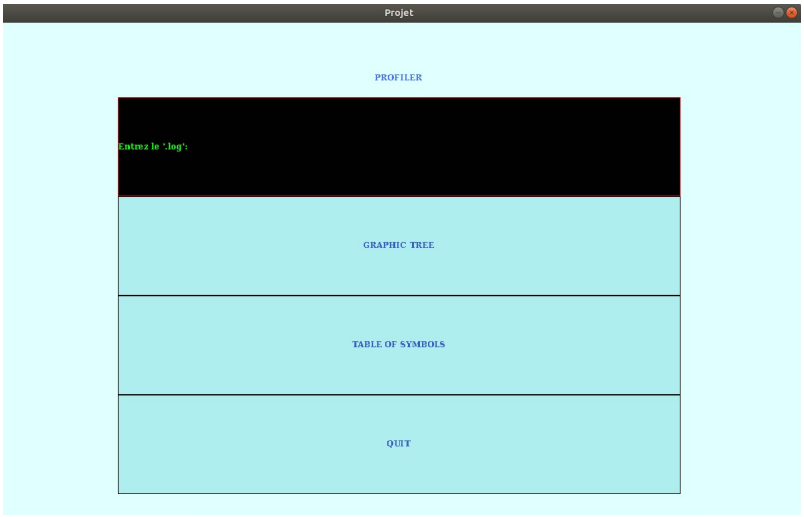
Tableau récap obtenu par le groupe de Armand et Jeanne avec un de nos .log

Exécution et Utilisation

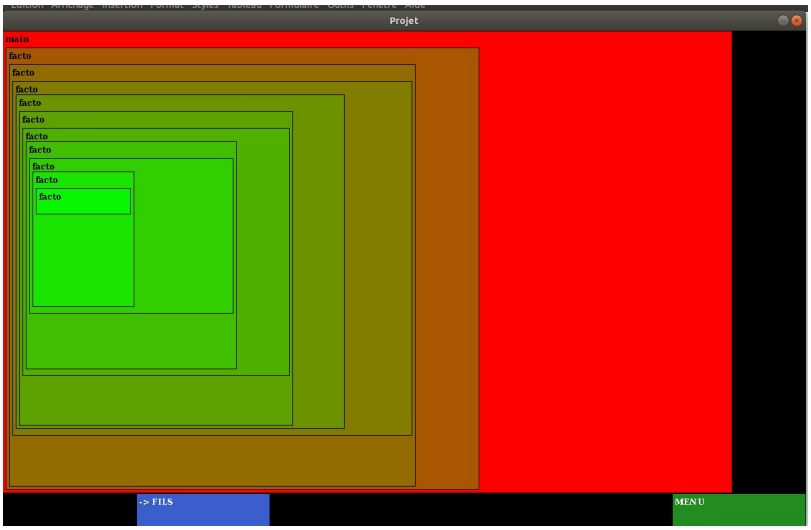
Une fois compilé, on utilise la commande ./bin/myprofiler. Cela ouvre l'interface graphique d'utilisation de la bibliothèque MLV.



Dans le menu obtenu, on peut donc load un fichier via une input_box qui vérifie l'entrée d'un .log et l'existence du dit fichier. Cela demande le chemin (doc/fichier.log).

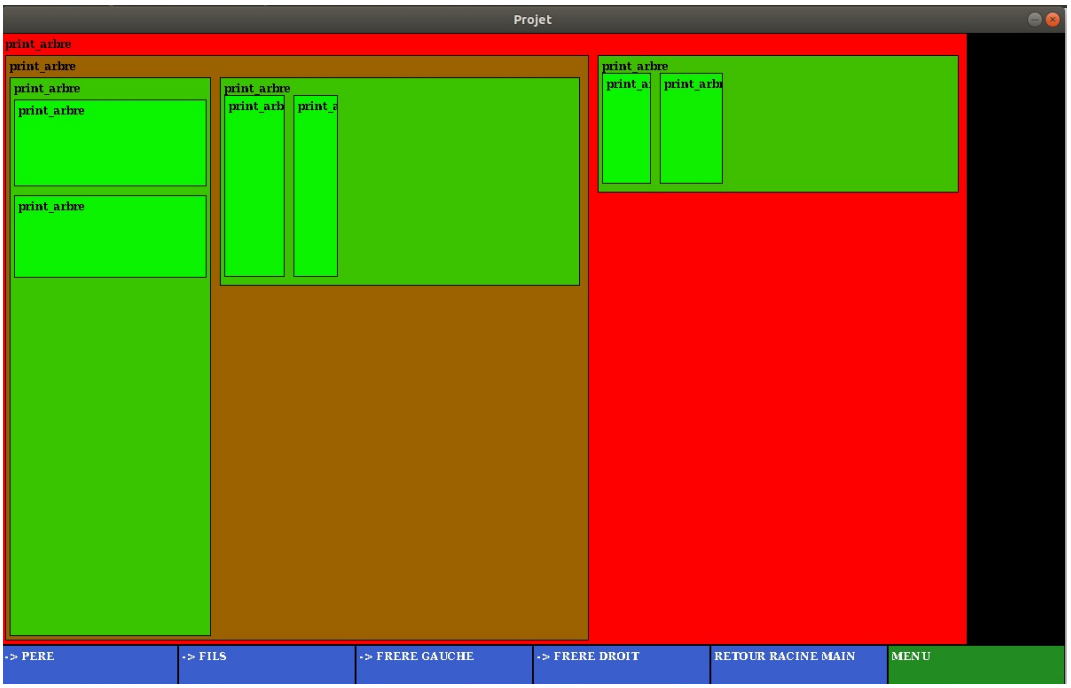


Une fois le fichier chargé, on peut accéder à la représentation graphique de l'arbre :



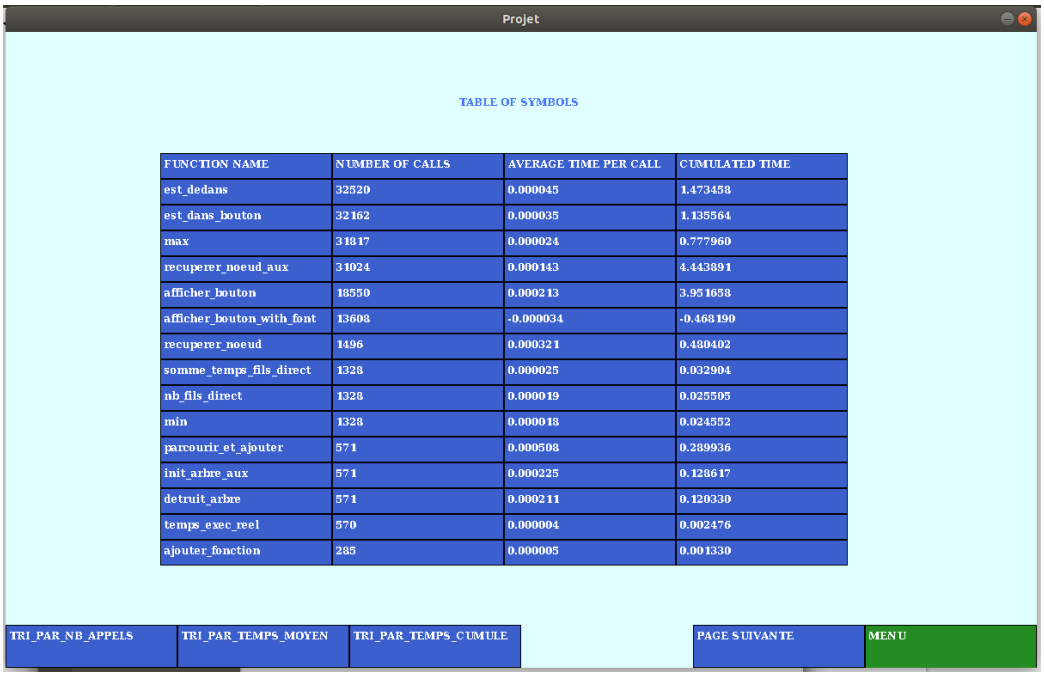
Exemple de la factorielle 10

On retrouve en bas des boutons permettant quand possible la navigation dans l'arbre qui s'actualise ainsi qu'un bouton en bas à droite pour retourner au menu.



Autre exemple (fichier : LOPES_LIN.log) dans lequel on a déjà navigué dans l'arbre, on a donc accès aux boutons PERE, FILS, FRERE GAUCHE, FRERE DROIT et RETOUR MAIN.

On peut aussi accéder à la table des symboles qui retient les éventuels déplacements effectués dans la partie graphique de l'arbre. On obtient un récapitulatif des appels des fonctions. Il y a 15 fonctions par page avec accès aux pages suivantes si nécessaire. Il y a aussi les boutons de tri qui font d'ailleurs revenir à la première page.



Pour quitter le programme, il faut revenir dans le menu et cliquer sur QUIT afin de s’assurer que profile.log obtenu par la macro soit valide.

Maxime KUOCH
Fabien LAMBERT--DELAVAQUERIE

Construction du code

Organisation

Comme indiqué dans le sujet, nous avons utilisé la plateforme redmine à l'adresse https://forge-etud.u-pem.fr/projects/profiler_kuoch_lambert-delavaquerie afin d'avoir plus facilement accès aux modifications de chacun. Le récapitulatif des différents commits est dans le fichier log_dev.

Nous avons organisé notre code selon différents modules.

-Le module Arbre qui représente l'arbre obtenu par .log, nous avons choisi assez tôt de mettre dans la structure des nœuds le nœud père afin de faciliter plus tard la navigation graphique dans l'arbre. Ce module gère la création des arbres.

-Le module draw qui sert à l'affichage graphique de l'arbre et de la table de symboles ainsi qu'à la récupération des actions utilisateur.

-Le module Recap qui contient une structure RECAP qui est un tableau de fonctions (structure qui contient le nom, le nombre d'appels, le temps moyen et le temps cumulé). Ce module gère les tris et la création des RECAPs.

-Le module Menu introduit assez tard, lorsque nous avons décidé d'avoir un exécutable qui fonctionne exclusivement via l'interface graphique. Il sert surtout à afficher le menu, à récupérer le chemin du .log à charger et à indiquer graphiquement que le programme agit toujours à la création de l'arbre ou du récap de manière plus agréable en cas de .log très lourds.

Difficultés et Bugs

Nous n'avons pas eu de grosses difficultés au début jusqu'à ce que nous essayons de profiler notre propre code. En effet, lors de certains boucles ou if la macro return ne fonctionnait pas sans accolades, un problème simple qui nous a tout de même gêné ainsi que le fait qu'avec des .log plus importants on obtenait des temps parfois négatifs. Nous en avons remarqué dans ces cas là qu'il s'agissait du fait que la macro renvoyait des temps avec plus de 6 chiffres après la virgule, nous avons donc forcé le fait d'avoir toujours ces 6 chiffres.

Pour le RECAP, nous avons choisi une TAILLE_MAX de 100 pour faciliter la gestion de celui-ci, cependant, si le .log contient plus de 100 fonctions aux noms différents, cela peut entraîner une erreur de segmentation.

Nous n’avons pas géré le cas où le .log n’est pas conforme ce qui entraîne des erreurs de segmentation lorsque l’on charge un fichier .log incompatible.

Dans l’affichage graphique de l’arbre, si le fils est tout petit, on ne l’affichait ni lui ni ses frères, nous avons donc forcé un text box de taille défini avec … afin d’afficher ses frères droits après celui-ci tout en restant lisible.

Améliorations possibles

Il faudrait sécuriser la partie récap en modifiant la fait que ce soit un tableau de taille défini. La vérification du fichier entré avant la création de l’arbre serait un plus. Au niveau de l’interface graphique, le fait de pouvoir cliquer sur la racine voulue dans l’arbre serait parfois plus agréable que l’utilisation des boutons qui, surtout pour naviguer entre les frères n’est pas intuitive.