Automne 2016

Algorithmes et structures de données pour ingénieur

GLO-2100

Travail à faire en équipes de 1 ou 2
À rendre avant la date indiquée
sur le site ENA du cours
(Voir modalités de remise à la fin de l'énoncé)

Travail Pratique #3

Investigation pour améliorer l'efficacité des algorithmes de plus court chemin

UNIVERSITÉ LAVAL
Faculté des sciences et de génie
Département d'informatique
et de génie logiciel

1 - Objectif général

Ce TP constitue la dernière partie de votre travail de session. Il reste encore du travail pour compléter votre application RTC, cependant nous allons nous contenter de rendre **efficace** ce qui a été fait jusqu'à présent. Ainsi, ce TP constitue une occasion de mettre en oeuvre vos aptitudes à investiguer un problème en vu d'améliorer son efficacité.

2 – Travail à faire

Au TP2, vous avez construit une structure de données "Graphe" et implémenté des algorithmes de plus court chemin. Ces algorithmes, Dijkstra et Bellman Ford, devraient respectivement être en $O(n^2)$ et $O(nm)$ selon les contraintes de l'énoncé (n sommets et m arcs). Cependant, nous sommes convaincu qu'il y a de meilleurs algorithmes ou implémentations des algorithmes précédents qui ont une meilleur complexité dans le cadre de notre application. Dans votre classe Reseau, on vous demande d'ajouter un algorithme de plus court chemin qui soit à la fois plus rapide que la version naïve de Dijkstra ($O(n^2)$) et celle de Bellman Ford ($O(nm)$). Votre démarche d'investigation pour arriver à cet algorithme devra être l'objet d'un rapport écrit à rendre. Cette démarche est aussi importante que la solution que vous aurez trouvé.

Dans votre réflexion, vous devez vous en tenir à des algorithmes non parallèles car un seul "thread" d'exécution sera à la disposition de l'utilisateur de votre application. Nous vous conseillons de commencer votre démarche d'investigation en examinant, la structure du réseau de transport de la RTC. En effet, c'est sur ce réseau que vous ferez vos tests, donc autant analyser ses paramètres (densité, existence de cycles, poids, etc) avant de commencer votre recherche. Veuillez consigner vos observations dans votre rapport. Aussi, en examinant attentivement les notes de votre cours, vous pourrez commencer à faire une élimination progressive des algorithmes de plus court chemin que vous avez vus mais qui ne sont pas applicables pour votre problème. Nous vous donnons comme piste de solutions potentielles l'algorithme de Bellman-Ford pour graphes acycliques avec l'utilisation du tri topologique, l'algorithme de Floyd-Warshall et celui de Dijkstra utilisant un monceau pour les nœuds non solutionnées. Vous pouvez bien sûr explorer d'autres pistes. Examinez de près toutes les pistes et justifiez le choix qui vous semble être le plus prometteur. Consignez les raisons et les résultats des étapes précédentes dans un rapport avant de continuer votre travail. Finalement, avant de passer à votre implémentation, sachez qu'il est possible d'augmenter l'efficacité d'une implémentation en évitant d'utiliser certains conteneurs ou algorithmes fournis par la STL. Ceci est dû au fait que la STL supporte parfois des opérations dont nous n'avions pas besoin dans notre code; ce qui rend son utilisation sous-optimale pour notre application. Il se peut également qu'un conteneur de la STL ne supporte pas certaines opérations que vous avez besoin. Par exemple, si vous optez pour l'algorithme de Dijkstra avec un monceau, il va falloir obligatoirement coder ce monceau vous même car la STL n'offre pas de méthode permettant la mise à jour d'un noeud interne du monceau en temps logarithmique.

Pour évaluer et comparer les performances des deux algorithmes du TP2 à celui que vous proposez dans ce travail, nous vous conseillons d'utiliser la fonction gettimeoffday() (voir la documentation à l'adresse http://linux.die.net/man/2/gettimeofday) de votre système linux qui est accessible en C++ à travers la librairie <sys/time.h>. Dans le fichier investigation.cpp qui vous est fourni avec cet énoncé, vous trouverez une fonction pour faire la différence en microsecondes (= 10^{-6} seconde) entre deux appels de gettimeoffday. Aussi, vous verrez que ce fichier contient du code pour calculer le temps d'exécution

moyen d'un algorithme de plus court chemin en faisant la moyenne des temps d'exécution sur un certain nombre de paires (origine, destination) du réseau de la RTC. Notez que vous devez utiliser cette dernière méthode pour mesurer le temps des différents algorithmes et ensuite consigner ces résultats dans votre rapport. **Vous êtes libre d'ajouter tout ce que vous voulez dans le fichier investigation.h, investigation.cpp, ainsi que le main.cpp fournis**. **Notez que nous vous fournissons également les fichiers reseau.h et reseau.cpp contenant les algorithmes de Belleman-Ford et Dijkstra et une méthode meilleurPlusCourtChemin() qui contiendra votre code pour votre meilleur algorithme de plus court chemin que vous aurez trouvé. Nous vous fournissons également la librairie statique libtp1.a contenant les modules compilés du TP1.** Nous avons mis du code bien documenté qui pourrait vous aider à vite avancer dans votre investigation mais vous devez aller plus loin. Ces fichiers fournis visent à vous aider pour votre TP3; ils ne constituent pas une solution au TP2.

Votre rapport d'investigation doit être un document PDF d'au plus 3 pages qui contiendra les éléments suivants.

1. La formulation, en vos propres mots, de ce que vous devez faire pour cette partie du TP. (environ une demie page)
2. Un résumé sur les algorithmes de plus court chemin que vous avez envisagés au début de votre investigation et les raisons qui ont conduit à éliminer certains de ces algorithmes. Notez également les raisons qui penchent en faveur de votre algorithme final dans le cadre du réseau de la RTC. (environ une demie page)
3. Une description complète des étapes algorithmiques qui ont conduit à votre implémentation finale. Par exemple, les structures de données que vous avez dû modifier dans un algorithme, les opérations que vous avez dû optimiser. En gros tout votre processus de planification depuis le moment où vous avez choisi une certaine direction de résolution du problème. (environ une page)
4. Une présentation des résultats obtenus suite à votre démarche d'investigation et votre analyse de ces résultats. Recommanderiez-vous votre propre algorithme pour trouver un plus court chemin dans le réseau de la RTC? (environ une page)

Portez une attention particulière aux points suivants :

- Pour obtenir une comparaison fiable du temps d'exécution des différents algorithmes, prévoyez, pour chaque algorithme, de calculer 20 fois le temps d'exécution moyen sur un certain nombre de paires (environ 1000) . En effet, vous observerez que le temps d'exécution retourné par gettimeofday() varie d'une fois à l'autre et il dépends de la quantité de processus actifs sur votre ordinateur. Faites donc toujours vos tests en vous assurant d'avoir fermé toutes les autres applications (mail, office, etc...) sur votre ordinateur.
- La qualité de votre rapport écrit est tout aussi importante que la qualité de votre code. Ne négligez pas l'un au dépend de l'autre.
- Votre rapport doit être clair et les arguments présentés doivent être pertinents.
- Vous avez la responsabilité de relire attentivement votre rapport afin de vous assurer qu'il soit sans faute.

Remarque pour les étudiants (et seulement ces étudiants) dans un programme de génie Comme mentionné sur le plan de cours, ce TP sera utilisé pour évaluer la Qualité 3 (Q3: Investigation) exigée par le BCAPG pour tout programme de formation ouvrant la possibilité de l'obtention du titre d'ingénieur. Si vous n'effectuez pas ce travail, cette Qualité ne pourra pas être évaluée pour ce cours et il est possible que cela ait des conséquences pour l'obtention éventuelle de cette Qualité à la fin de votre programme d'étude. Il est donc crucial que tout étudiant de génie effectue ce TP. Ainsi, toute équipe où il ya un étudiant en génie est obligé de compléter ce travail.

3 – Livrables et critères de correction

Vous serez évalué sur:
- Qualité d'investigation: il est important que votre démarche d'investigation soit cohérente et claire.
- Qualité du rapport: le contenu et la forme du rapport
- Efficacité du code : choix judicieux des structures de données, respect des spécifications sur la complexité des opérations demandées;

- Clarté du code : il est important que votre code soit clair, lisible et documenté (sans excès) afin de le corriger au mieux;

- Respect des prototypes fournis et de l'énoncé: il est capital de respecter les prototypes afin d'aider la correction.

- Critère de complétion: facteur multiplicatif sur la note globale, par exemple: « tout ce qui est demandé est implémenté » équivaut à 1.0, « la moitié » à 0.5, « rien » à 0.0) (le facteur multiplicatif n'est pas limité à ces trois exemples, il sera calculé.

- Critère de compilation: vous risquez de perdre entre 20% et 100% de la note si votre code ne compile pas (affecte donc le facteur multiplicatif) sur la machine virtuelle.

- Critère d'exécution: vous risquez de perdre un grand pourcentage de la note si votre code s'interrompt pendant l'exécution (affecte le facteur multiplicatif). La gravité de l'erreur détermine le pourcentage perdu.

4 - Fichiers à remettre

Dans la boîte de dépôt du portail du cours, vous devez remettre les fichiers de toutes les classes du projet et votre rapport. Il s'agit donc des fichiers suivants :

- reseau (.cpp et .h) contenant la classe Reseau mise à jour.
- investigation (.cpp et .h) contenant la classe GestionnaireInvestigation mise à jour.
- main.cpp contenant les expérimentations que vous avez faites pour votre rapport.
- rapport.pdf contenant votre rapport.

Vous devez insérer ces fichiers dans une archive ayant pour nom NomDeFamille_Prénom.zip. SVP, ne remettre aucun autre fichier. Bien que nous encourageons l'utilisation de Google Test pour vos tests unitaires, on vous demande de ne pas remettre vos testeurs (même si nous vous recommandons de bien tester votre code). Vous devez remettre votre travail dans la boîte de dépôt du portail du cours. Aucune remise par courriel ne sera acceptée. Tout travail remis en retard perdra 2 points par heure de retard. Donc, au bout de 10 heures de retard, la note maximale du TP sera de 80, et après 50 heures de retard, la note maximale sera de 0. Vous pouvez déposer autant de versions que vous le désirez. Seul le dernier dépôt sera corrigé.

Tout comme pour le TP3, ce travail est fait en équipe de 1 ou 2. Vous aurez jusqu'au 29 novembre pour vous former une équipe. Le dépôt de votre TP sera permis uniquement après cette date.

ATTENTION : vous avez la responsabilité de vérifier l'intégrité des fichiers que vous avez remis dans la boîte de dépôt. Donc, nous vous recommandons fortement que vous examiniez ce que vous avez remis afin d'en vérifier l'intégrité. Le mécanisme de téléversement du portail ne corrompt pas les fichiers. Cependant, il peut arriver que votre archive ait été corrompue si, lorsque vous l'avez créée, certains des fichiers étaient ouverts par d'autres applications (Eclipse par exemple...). Vérifiez donc l'intégrité de votre archive après l'avoir construite. SVP : ne pas déposer tout votre «workspace» !!

Bon travail!