Prof. Dr. Henning Meyerhenke
Dr. Alexander van der Grinten

25.10.2018
Abgabe Dienstag, 30.10.2018, 10:00

Übungen zur Vorlesung

Algorithmische Methoden für Schwere Optimierungsprobleme

WS 2018/19

Blatt 1

AUFGABE 1:

(a) Zeigen Sie, dass es für jeden Graphen, der mit k Farben gefärbt werden kann, mindestens eine Permutation der Knoten gibt, so dass GreedyCol eine k-Färbung findet.

(b) Geben Sie einen Algorithmus an (in Pseudocode), der einen Graphen G als Input erhält und entweder eine 2-Färbung von G berechnet oder ermittelt, dass G nicht 2-färbbar ist.

AUFGABE 2:
Implementieren Sie den Algorithmus GreedyCol aus der Vorlesung. Schreiben Sie Ihre Implementierung in C++ und verwenden Sie NetworKit, um den Eingabegraphen zu verwalten.

Input. Ihr Programm sollte den Pfad eines Eingabegraphen im SNAP-Format (`https://snap.stanford.edu/data/`) als Parameter auf der Kommandozeile erhalten. Nutzen Sie NetworKits SNAPGraphReader um den Graphen einzulesen. In Moodle finden Sie einen Testgraphen (den Petersen-Graph, siehe Vorlesung). Weitere Graphen lassen sich auf der SNAP-Website downloaden.

Output. Geben Sie die Farben jedes Knotens nacheinander auf die Konsole aus. Geben Sie dabei jede Farbe in einer eigenen Zeile aus. Geben Sie ansonsten nichts auf `stdout` aus.

Verifizieren der Lösung. In Moodle finden Sie ein Programm `verify-col.cpp`, welches testet, ob eine gegebene Färbung zulässig ist. Nutzen Sie dieses Tool, um zu überprüfen, ob Ihre Färbungen korrekt sind.

Abgabeformat Geben Sie Ihr Programm als *eine* `.cpp` Datei ab (ohne zusätzliche Header o.ä.). Stellen Sie sicher, dass Ihr Programm korrekt kompiliert und dass die Ausgabe durch den Verifizierer gelesen werden kann.
Es folgen noch einige Hinweise und Tips:

- Sehen Sie sich `verify-col.cpp` als Inspiration für Ihren eigenen Code an. Dort werden die hier erwähnten Konzepte beispielhaft verwendet.
- Knoten werden in NetworKit durch `NetworKit::node` repräsentiert, was ein gewöhnlicher Integer-Datentyp ist. Dieser Integer liegt zwischen `0` und `graph.upperNodeIdBound()`.
- Nutzen Sie Datenstrukturen wie `std::vector`, um Ihre Daten zu verwalten. `https://en.cppreference.com/` ist eine gute Referenz der C++ Standardbibliothek.

- Nutzen Sie NetworKit's Methoden wie `forNodes()`, `forEdgesOf()` oder `forEdges()`, um über die Knoten und Kanten eines Graphen zu iterieren.
- Beachten Sie, dass der SNAPGraphReader die Knoten umnummeriert. Dies sollten Sie verhindern, indem Sie den entsprechenden Parameter im Konstruktur auf `false` setzen.
- Sie können die Lösung Ihres Programs direkt an den Verifizierer übergeben, indem Sie einen Befehl wie `./greedycol <graph> | ./verify-col <graph> /dev/stdin` nutzen.
- Falls Sie zusätzliche Nachrichten ausgeben wollen, nutzen Sie `stderr` (in C++ `std::cerr`); dies stellt sicher, dass der Output auf `stdout` korrekt vom Verifizierer verarbeitet werden kann.
- Setzen Sie die Environment-Variable `LD_LIBRARY_PATH` auf den Pfad Ihrer `libnetworkit.so` (d.h. auf das NetworKit-Verzeichnis), so dass diese Shared-Library zur Laufzeit gefunden werden kann. Siehe auch die C++ Slides.