Prof. Dr. A.K. Hartmann
Blatt 10, 3.3.2023

WiSe 2022/23, 5.04.4521

Übungen zur Computerorientierten Physik

10 Graphen und Perkolationsübergang

1. Laden Sie sich das Programm Paket `graphs.tar` vom `StudIP`. Es enthält die Dateien `list.c`, `list.h`, `graphs_lists.h`, `graphs_fragment.c`, `graphs_main.c`

   Sie können Compilieren mit

   ```
   cc -o graphs list.c graphs_fragment.c graphs_main.c -Wall -lm
   ```

2. Lernen Sie das Hauptprogramm `graphs_main.c` gut kennen und schauen Sie sich zumindetens die Aufrufstruktur der Funktionen in `graphs_fragment.c` an.

3. Komplettieren Sie die Funktion zur Erzeugung eines Zufallsgraphen: Für einen gegebenen Graphen werden `m` Kanten $\{i, j\}$ hinzugefügt, wobei $i \neq j$ und die Kante in dem Graphen noch nicht vorhanden sein darf

   ```
   /******************** gs_random_graph() ******************/
   /** Function adds exactly m randomly chosen edges to    **/
   /** the graph.                                          **/
   /** No self loops are allowed. No edge is allowed to    **/
   /*+ appear twice!                                       **/
   /** PARAMETERS: (*)= return-paramter                    **/
   /**         (*) g: graph                                **/
   /**             m: number of edges to be added          **/
   /** RETURNS:                                            **/
   /**     (nothing)                                       **/
   /*********************************************************/
   void gs_random_graph(gs_graph_t *g, int m)
   ```

   (2 Punkte)

4. Testen Sie für sich die Funktion mit den üblichen Methoden (debugger etc) sorgfältig.

   Messen Sie darüber hinaus die Gradverteilung (mittels Array, Code existiert im Hauptprgramm ist aber auskommentiert) ein paar (mindestens 2) Werte von $c = 2m/n$ (z.B. $c = 0.5$ und $c = 2$) und vergleichen Sie mit der analytisch bekannten Poissonverteilung

   $$p_k = \frac{c^k}{k!} e^{-c} .$$

   Diese können Sie in `gnuplot` direkt realisieren: `p(x)=c**x*exp(-c)/gamma(x+1)`, mit jeweils geeignet gesetztem Wert von c. (1 Punkt)

5. Komplettieren Sie die Funktion zur Berechnung der Zusammenhangskomponenten (Cluster). Orientieren Sie sich dabei an dem einfachen Clusteralgorithmus im Kapitel zur Perkolation (also KEINE Bäume, Pfadkomprimierung etc):

```
/*********************** gs_clusters() ********************/
/** Calculates for graph  the connected components:       **/
/** An array is updated, where for each node the id       **/
/** (0,1,...,num_clusters-1) of the cluster it belongs to  **/
/** is stored. Also, an array is created an returend, where**/
/** for each cluster the number of nodes is contains is    **/
/** stored.                                                **/
/** PARAMETERS: (*)= return-paramter                       **/
/**            g: graph                                    **/
/**       (*) id: for each node, id of cluster             **/
/** (*) num_cl_p: (p. to) number of clusters               **/
/** RETURNS:                                               **/
/**     array with sizes of each cluster (created)         **/
/************************************************************/
int *gs_clusters(gs_graph_t *g, int *id, int *num_cl_p)
```

   indem Sie den vorgegebenen Rahmen komplettieren.

   Hinweis: Für den Knoten `k` des Graphen `g` können Sie über alle Nachbarn iterieren durch:

```
neighb = g->node[k].neighbors;
while(neighb != 0)          /* loop over all neighbors */
{
   ..... neighb->info....        /* access node number */
  neighb = neighb -> next;
}
```

   Anmerkung: Im zweiten Teil des bereits vorhandene Codes wird noch für jeden Cluster die Anzahl der enthaltenen Knoten bestimmt. (5 Punkte)

6. Untersuchen Sie den Perkolationsübergang von Zufallsgraphen. Im Einzelnen

   - Bestimmen Sie für verschiedene Graphgrößen $N = 100, 200, 500, 1000$ mit `graph_main.c` jeweils die mitteler Größe der größten Komponenten $S_{\max}$ sowie den Anteil $s_{\max} = S_{\max}/N$ als Funktion der Konnektivität c und schreiben die die Ergebnisse in Dateien `graph_NNN.dat`, wobei sie `NNN` durch den Wert von N ersetzen.
   - Plotten Sie mit `gnuplot` die relative Größe $S_{\max}/N$ als Funktion von c. Wo sehen Sie den Perkolationsübergang c_{crit}? (2 Punkte)