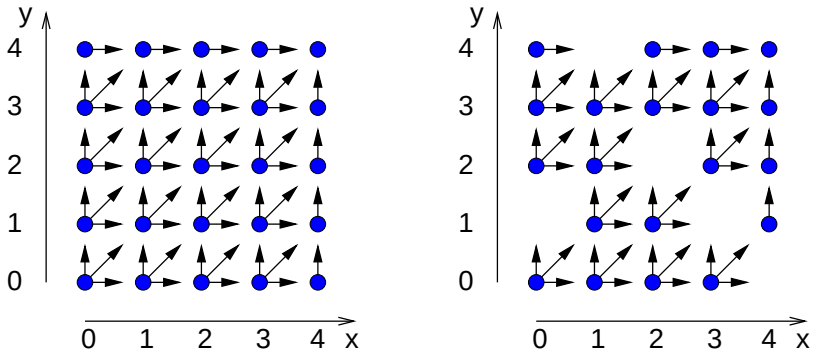
Prof. Dr. A.K. Hartmann
WiSe 2016/17, 5.04.4521
Blatt 4, 21.2.2017

Übungen zur Computerorientierten Physik

3 Zahl der Pfade

Betrachten Sie ein Gitter von $l \times l$ Plätzen, welche durch gerichtete ("Einbahnstraßen") Pfeile wie folgt verbunden sind (hier $l = 5$, siehe links, Plätze: Kreise):



(Der rechte Teil zeigt eine Variante, bei der einige Plätze nicht vorhanden sind, von ihnen gehen keine Pfeile aus, siehe unten)
Ziel ist es, die Anzahl der verschiedenen möglichen Pfade zu zählen, die von $(0, 0)$ nach $(l - 1, l - 1)$ führen. Dazu sollten Sie ein Array `num_paths[][]` aufsetzen, so dass `num_paths`$[x][y]$ die Zahl der Pfade von $(0, 0)$ nach (x, y) enthält. Das kann iterativ erfolgen, indem zuerst `num_paths`$[x][0] =$ `num_paths`$[0][y] = 1$ für alle x, y gesetzt wird. Danach können die Einträge für größere Werte von x, y unter Verwendung der Einträge bei kleineren Indizes berechnet werden (dynamisches Programmieren).
Ihre Aufgaben sind:

- Stellen Sie eine einfache rekursive Formel auf, nach der `num_paths`$[x][y]$ berechnet wird. Prüfen Sie z.B. ob `num_paths`$[1][1] = 3$ und `num_paths`$[2][2] = 13$ herauskommt.
- Laden Sie sich das Programmfragment `pathnumber_fragment.c` von StudIP.
- Lesen Sie das Programm durch und stellen Sie sicher, dass Sie alles verstehen.

  Achtung: Im Programm ist es erlaubt, dass Gitterplätze nicht besetzt sind, d.h. keine Pfade können darüber führen. Das wird mittels des Arrays `occupied[][]` realisiert. Das Hauptprogramm besetzt jeden Gitterplatz zufällig mit einer Wahrscheinlichkeit `p_occ`, außer den Platz $(0, 0)$, der immer besetzt ist. Die aktuelle Programmversion hat `p_occ` $= 1$, d.h. das Gitter ist vollständig belegt.

- Vervollständigen Sie die Funktion

  `void calc_paths(int l, double **num_paths, short int **occupied)`

  so dass die Anzahl der Pfade berechnet wird.

  Sie können hier die Variablen `occupied[][]` ignorieren, wenn Sie wollen.

- Compilieren Sie, z.B. aus der Shell mit

  ```
  cc -o pathnumber pathnumber_fragment.c -g -Wall
  ```

- Lassen Sie das Programm laufen und übergeben Sie die Gittergröße als Programmparameter, z.B. durch Eingabe in der Shell

  ```
  ./pathnumber 8
  ```

  Das sollte ergeben:

  ```
  # size num_paths
  8 48639.000000
  ```

- Führen Sie Simulationen für verschiedene Größen l durch und schreiben Sie die Ergebnisse in eine Datei, z.B. `num_l.dat`. Sie können das mit einer Shell Schleife erreichen:

  ```
  for l in 2 4 6 8 10 12 14 16; do ./pathnumber $l; done > path_l.dat
  ```

  Stellen Sie die sich ergebene Funktion mit einem Plot-Programm dar, z.B. `gnuplot`. Was beobachten Sie, wenn Sie die y-Achse logarithmisch skalieren?

- Zusätzliche Übung (Fortgeschrittene):

  Erweitern Sie die Pfadberechnung derart, dass `occupied[][]` berücksichtigt wird.

  Berechnen Sie nun für eine gegebene Gittergröße l die *mittlere* Anzahl der Pfade für unterschiedliche Besetzungswahrscheinlichkeiten `p_occ`. Mittelung bedeutet, dass Sie eine Schleife über verschiedenen unabhängige Realisierungen der Verteilung der Besetzungen durchführen, z.B. $r = 1 \ldots 100$. Das kann erreicht werden, indem der Zufallszahlengenerator (Vorgriff auf spätere Vorlesung ...) bei jedem Schleifendurchlauf mit dem Schleifenindex r initialisiert wird, die Pfadlängen aufsummiert werden und die Summe schließlich durch die Zahl der Realisierungen (z.B. 100) geteilt wird. Plotten Sie diese Funktion für verschiedene Größen, z.B. $l = 8, 16, 32$.

  Was beobachten Sie?

  Hinweis: Möglicherweise ist es besser den Mittelwert vom Logarithmus der Pfadanzahl zu berechnen, aber dann sollten Sie log(1) statt log(0) nehmen sofern, es keinen Pfad gibt.