Algorithmen und Komplexität **Probabilistische Algorithmen**

18. November

Dana Müller, Robert Stricker *Hochschule Zittau/Görlitz*

Ziel der Übung

Das Ziel dieser Übung ist es, dass in der Vorlesung vermittelte Wissen praktisch umzusetzen. Nach dieser Übung sollten die Fähigkeiten zur Erstellung von Zufallsgeneratoren und probabilistischen Algorithmen vorhanden sein.

Hinweise

Die Übung ist modular aufgebaut und besteht aus den drei Teilen Zufall, Algorithmen und Analyse. Es ist nicht zielführend nur einen dieser Abschnitte während der Übungszeit zu bearbeiten. Es wird empfohlen, nach jeder Aufgabe den Themenabschnitt zu wechseln.

Zur Erinnerung

Die Gleichung für die lineare Kongruenzmethode lautet:

$$x_{n+1} = (a \cdot x_n + c) \bmod m$$

Eine Indikatorfunktion ist definiert durch:

$$I\{A\} = \begin{cases} 1 & \text{falls A eintritt} \\ 0 & \text{falls A nicht eintritt} \end{cases}$$

Aufgaben

Zufall

1. Schreiben Sie ein Programm in einer von Ihnen gewählten Programmiersprache, welches Zufallszahlen nach der Kongruenzmethode erzeugt. Experimentieren Sie mit den Eingabeparametern! Als Hinweise seien hier der Überlauf des Zahlenbereiches und die Periodenlänge genannt.
2. Schreiben Sie unter Verwendung Ihrer Ergebnisse aus Aufgabe 1 ein Programm, welches eine Folge von Zufallszahlen ausgibt, deren Startwert beim Programmstart gewählt wird. (Stichwort: Randomize)
3. Machen Sie sich unter Verwendung Ihrer Ergebnisse aus Aufgabe 1 die optimale Parameterwahl klar. Wenden Sie das Verfahren zur Periodenlängenverlängerung an einem Modul Ihrer Wahl an. Optimieren Sie gegebenenfalls Ihre vorhandenen Prozeduren.

4. Implementieren Sie die Prozedur *rangeRandReal()*. Erzeugen Sie eine Folge von Zufallszahlen im Intervall [1,10] und bestimmen Sie die absoluten Häufigkeiten der aufgetretenen Zahlen.

5. Implementieren Sie die Prozedur *rangeRandGauss()*. Erzeugen Sie eine Folge von Zufallszahlen im Intervall [1,10] und bestimmen Sie die absoluten Häufigkeiten der aufgetretenen Zahlen. Nutzen Sie mindestens 1000 Zufallszahlen, beachten Sie die Periodenlänge!

Algorithmen

1. Erstellen Sie ein Programm, welches mit Hilfe der Methode des Zufallsregen den Näherungswert für das Integral

$$f(x) = \int_2^3 x^2 dx$$

ermittelt.

2. Erstellen Sie eine Prozedur ggT, welche den größten gemeinsamen Teiler zweier Zahlen ermittelt und zurückgibt.

Analyse

1. Berechnen Sie mittels der Indikatorfunktion die erwartete Anzahl für das Ereignis „Zahl“ beim Wurf einer fairen Münze.

2. Schreiben Sie eine Prozedur, welche ein Array aus Integer-Zahlen zufällig „mischt“.

Lösungen

Zufall

```
void kongruenz(int multiplikator,int inkrement,int start,int modul,int n) {
        int y = start;
        if (n == 0) {
                return;
        } else {
                y = (multiplikator * y + inkrement) % modul;
                System.out.println(y);
                start = y;
                kongruenz(multiplikator, inkrement, start, modul, (n - 1));
        }
}
```

```
public class Zufall {
        int start;
        int multiplikator;
        int inkrement;
        int modul;

        Zufall(int start, int multiplikator, int inkrement, int modul) {
                this.start = start;
                this.multiplikator = multiplikator;
                this.inkrement = inkrement;
                this.modul = modul;
        }

        public int randInt() {
                this.start = (this.multiplikator * this.start + this.inkrement)
                                % this.modul;
                return this.start;
        }

        public double randReal() {
                this.start = randInt();
                return this.start / ((double) this.modul);
        }

        public double rangeRandReal(int untereGrenze, int obereGrenze) {
                int n = 0;
                n = obereGrenze - untereGrenze + 1;
                return (untereGrenze + Math.floor(randReal() * n));
        }
```

```
        public int rangeRandGauss(int untereGrenze, int obereGrenze) {
                int n = 6;// Konstante zwischen 6 und 10, siehe [Knuth 1997]
                int sum = 0;
                for (int i = 0; i < n; i++) {
                        sum = (int) (sum + rangeRandReal(untereGrenze, obereGrenze));
                }
                return (sum / n);
        }
}
```

Algorithmen

1. Man berechnet den größten gemeinsamen Teiler mit Hilfe des euklidischen Algorithmus.

```
int ggt(int z1, int z2) {
                while (z2 != 0) {
                        if (z1 > z2) {
                                z1 = z1 - z2;
                        } else {
                                z2 = z2 - z1;
                        }
                }
                return z1;
        }
```

2. $$\int_2^3 x^2 dx = \left[\frac{1}{3}x^3\right]_2^3 = \frac{1}{3}\cdot 3^3 - \left(\frac{1}{3}\cdot 2^3\right) = \frac{27}{3} - \frac{8}{3} = \frac{19}{3}$$

   Die Prozedur sollte einen Näherungswert zu $\frac{19}{3}$ berechnen.

```
double integral_berechnen(int anzahl) {
                double integral;
                double t_innen = 0;

                for (int i = 0; i < anzahl; i++) {
                        double x = Math.random() + 2;
                        double y = Math.random() * 9;
                        if (y <= Math.pow(x, 2)) {
                                t_innen = t_innen + 1;
                        }
                }
                integral = (t_innen / anzahl) * 9;

                return integral;
        }
```

Analyse

1.

$$I\{A\} = \begin{cases} 1 & \text{wenn Z eintritt} \\ 0 & \text{wenn Z nicht eintritt} \end{cases}$$

$$\begin{aligned} E[X_Z] &= E[I\{Z\}] \\ &= 1 \cdot Pr\{Z\} + 0 \cdot Pr\{K\} \\ &= 1 \cdot \frac{1}{2} + 0 \cdot \frac{1}{2} \\ &= \frac{1}{2} \end{aligned}$$

2.
```
public int[] mischeArray(int[] x) {
        Zufall z = new Zufall(3, 61, 89, 1000);
        int[] zahlen = { 1, 2, 3, 4, 5, 6, 7, 8, 9, 10 };
        for (int i = 0; i < zahlen.length; i++) {
                int zz = (int) z.rangeRandReal(i, zahlen.length - 1);
                int tausch = zahlen[i];
                zahlen[i] = zahlen[zz];
                zahlen[zz] = tausch;
        }
        return zahlen;
}
```