Der Monte-Carlo-Algorithmus

Sonja Farghaly   Florian Gamböck   Bianca Zint

Fachbereich für Computerwissenschaften
Paris-Lodron-Universität Salzburg

Freitag, 27.01.2012

Gliederung

Gliederung

Gliederung

Gliederung

Gliederung

Anwendungsgebiete

- Kryptographie
- Hilfe für andere Programme

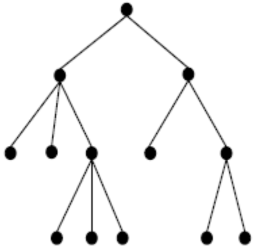
Vorteile

- Effizienz – Laufzeit, Speicherplatzbedarf
- Einfachheit – Implementierung, Verständnis

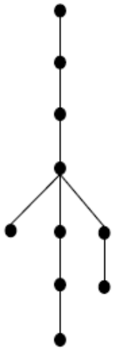
Beispiele

- Las-Vegas-Algorithmus
- Monte-Carlo-Algorithmus

Ablauf randomisierter Algorithmen



variable Laufzeit



konstante Laufzeit

Gliederung

Wichtige Personen

Janos Neumann de Margitta alias John von Neumann

Enrico Fermi

Stanislaw Marcin Ulam

Anwendungsgebiete

- Lösung von mehrdimensionalen Integralen
- Zuverlässigkeitsuntersuchungen
- Entscheidungsfindung
- Bestrahlungsplanung

Definition

- Suchproblem
- Entscheidungsproblem
- Lösung kann falsch sein

Fehlerwahrscheinlichkeit

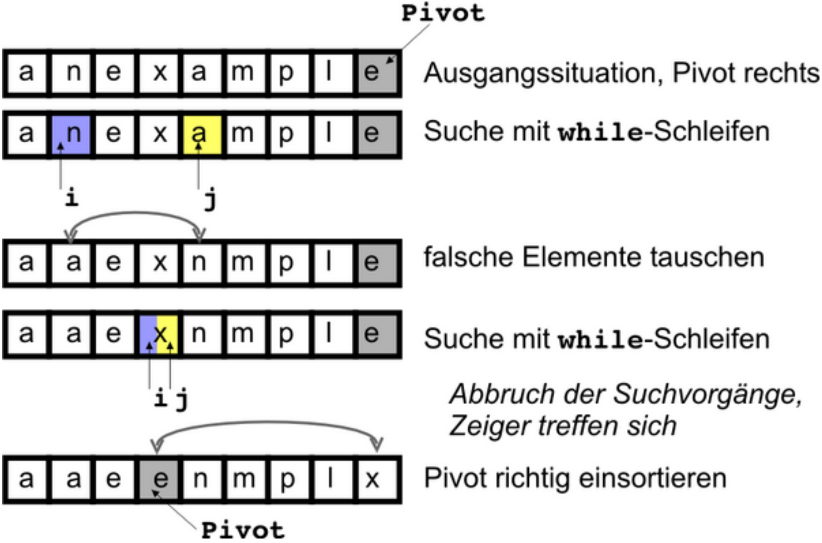
- Fehlerwahrscheinlichkeit: $p < 1 \;\rightarrow\; (1 - p)^x$
- Abstimmungsverfahren
- Onesided Error: Mindestens eine Art von Antwort ist sicher richtig
- Twosided Error: Die Antwort kann immer falsch sein

Gliederung

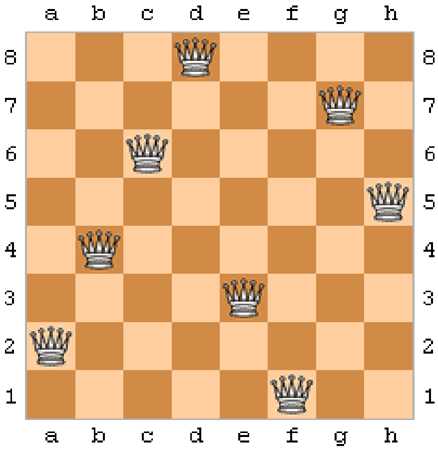
Las-Vegas: Gliederung in zwei Varianten

- „?“ nicht zulässig
    - QuickSort
- „?“ zulässig
    - Acht-Damen-Problem

QuickSort


Pivot
a n e x a m p l e
Ausgangssituation, Pivot rechts
a n e x a m p l e
Suche mit while-Schleifen
i
j
a a e x n m p l e
falsche Elemente tauschen
a a e x n m p l e
Suche mit while-Schleifen
i j
Abbruch der Suchvorgänge,
Zeiger treffen sich
a a e e n m p l x
Pivot richtig einsortieren
Pivot

Acht-Damen-Problem



Unterschied MC-A – LV-A

Monte-Carlo-Algorithmus

- darf versagen / falsch liegen
- unklar, wo versagt
- Rechenzeit konstant

Las-Vegas-Algorithmus

- darf höchstens „?“ ausgeben und nie versagen
- klar, wo versagt
- abhängig vom Worst Case unter Umständen hohe Rechenzeit

Merkregel

- MC: „mostly correct“, also nicht immer korrekt, also fehlerbehaftet.
- LV: „Laufzeit variabel“

Zusammenhang MC-A – LV-A

Umwandlung

- LV-A $\Rightarrow$ MC-A: immer möglich
- MC-A $\Rightarrow$ LV-A: nur mit Hilfsmittel möglich

Gliederung

Worum geht's?

- zufällige, aber gleichverteilte Erzeugung von Punkten innerhalb eines Quadrates
- wie viele Punkte sind innerhalb des eingeschlossenen Kreises?

$$\frac{\text{Punkte innerhalb des Kreises}}{\text{Erzeugte Punkte insgesamt}} = \frac{\text{Kreisfläche}}{\text{Quadratfläche}} = \frac{r^2\pi}{(2r)^2} = \frac{\pi}{4}$$

$$\Rightarrow \pi = 4 \cdot \frac{\text{Punkte innerhalb des Kreises}}{\text{Erzeugte Punkte insgesamt}}$$

Worum geht's?

- zufällige, aber gleichverteilte Erzeugung von Punkten innerhalb eines Quadrates
- wie viele Punkte sind innerhalb des eingeschlossenen Kreises?

$$\frac{\text{Punkte innerhalb des Kreises}}{\text{Erzeugte Punkte insgesamt}} = \frac{\text{Kreisfläche}}{\text{Quadratfläche}} = \frac{r^2\pi}{(2r)^2} = \frac{\pi}{4}$$

$$\Rightarrow \pi = 4 \cdot \frac{\text{Punkte innerhalb des Kreises}}{\text{Erzeugte Punkte insgesamt}}$$

Worum geht's?

- zufällige, aber gleichverteilte Erzeugung von Punkten innerhalb eines Quadrates
- wie viele Punkte sind innerhalb des eingeschlossenen Kreises?

$$\frac{\text{Punkte innerhalb des Kreises}}{\text{Erzeugte Punkte insgesamt}} = \frac{\text{Kreisfläche}}{\text{Quadratfläche}} = \frac{r^2\pi}{(2r)^2} = \frac{\pi}{4}$$

$$\Rightarrow \pi = 4 \cdot \frac{\text{Punkte innerhalb des Kreises}}{\text{Erzeugte Punkte insgesamt}}$$

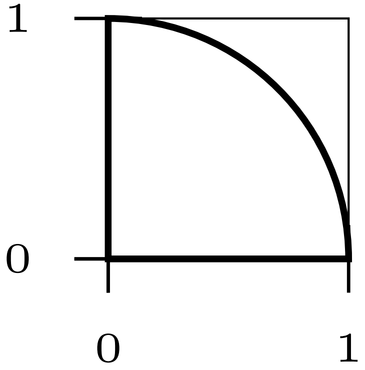
Worum geht's?

- zufällige, aber gleichverteilte Erzeugung von Punkten innerhalb eines Quadrates
- wie viele Punkte sind innerhalb des eingeschlossenen Kreises?

$$\frac{\text{Punkte innerhalb des Kreises}}{\text{Erzeugte Punkte insgesamt}} = \frac{\text{Kreisfläche}}{\text{Quadratfläche}} = \frac{r^2\pi}{(2r)^2} = \frac{\pi}{4}$$

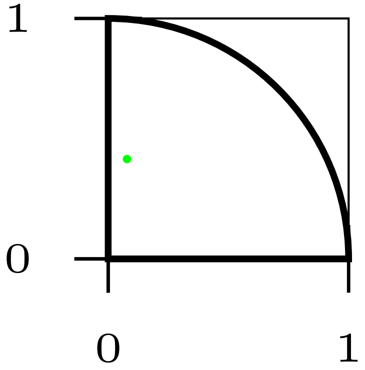
$$\Rightarrow \pi = 4 \cdot \frac{\text{Punkte innerhalb des Kreises}}{\text{Erzeugte Punkte insgesamt}}$$

Wie sieht das dann aus?



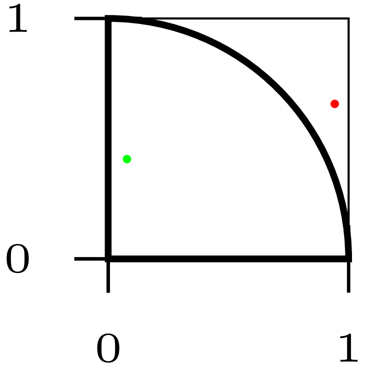
gut: 0
gesamt: 0
$\pi = 0.000$

Wie sieht das dann aus?



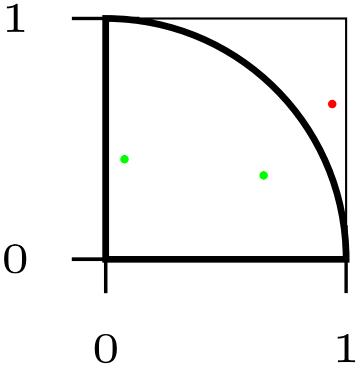
gut: 1
gesamt: 1
$\pi = 4.000$

Wie sieht das dann aus?



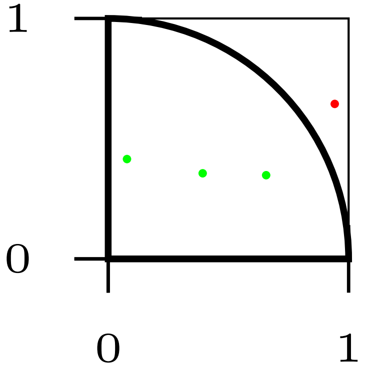
gut: 1
gesamt: 2
$\pi = 2.000$

Wie sieht das dann aus?



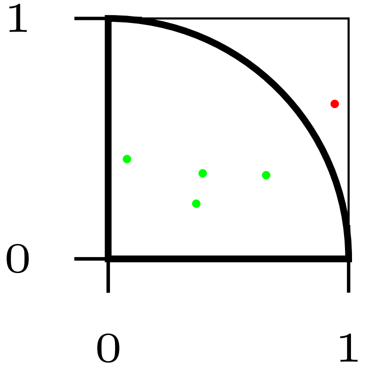
gut: 2
gesamt: 3
$\pi = 2.667$

Wie sieht das dann aus?



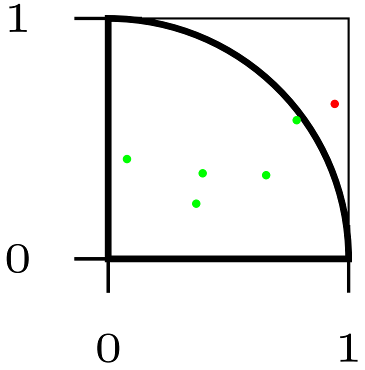
gut: 3
gesamt: 4
$\pi = 3.000$

Wie sieht das dann aus?



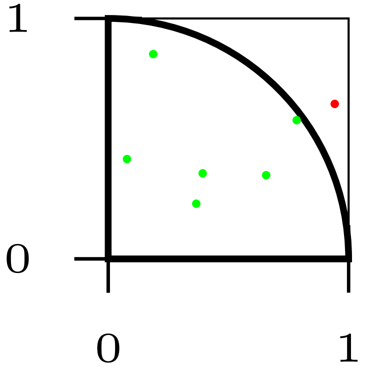
gut: 4
gesamt: 5
$\pi = 3.200$

Wie sieht das dann aus?



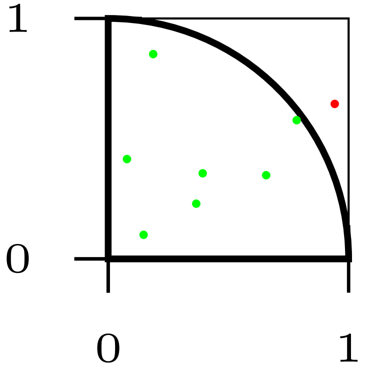
gut: 5
gesamt: 6
$\pi = 3.333$

Wie sieht das dann aus?



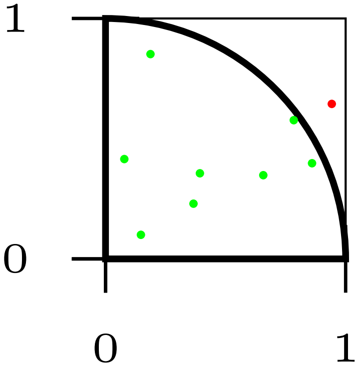
gut: 6
gesamt: 7
$\pi = 3.429$

Wie sieht das dann aus?



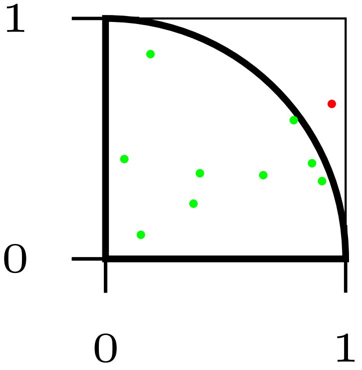
gut: 7
gesamt: 8
$\pi = 3.500$

Wie sieht das dann aus?



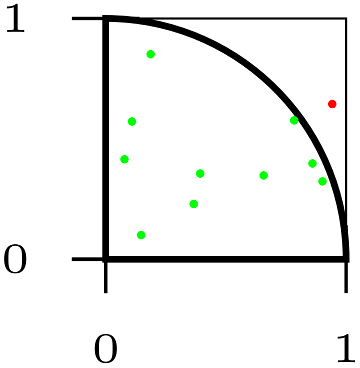
gut: 8
gesamt: 9
$\pi = 3.556$

Wie sieht das dann aus?



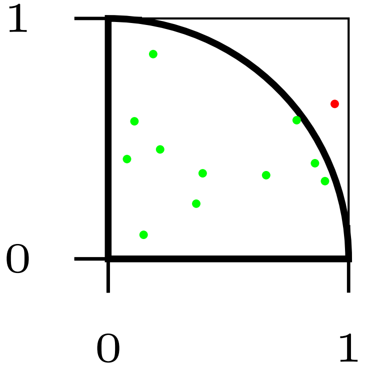
gut: 9
gesamt: 10
$\pi = 3.600$

Wie sieht das dann aus?



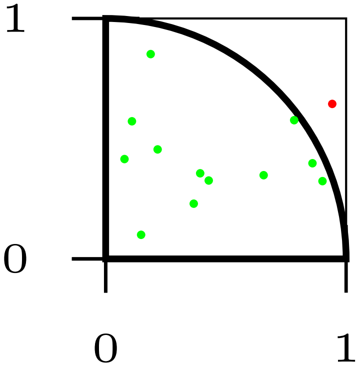
gut: 10
gesamt: 11
$\pi = 3.636$

Wie sieht das dann aus?



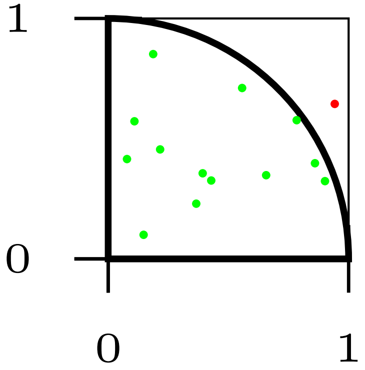
gut: 11
gesamt: 12
$\pi = 3.667$

Wie sieht das dann aus?



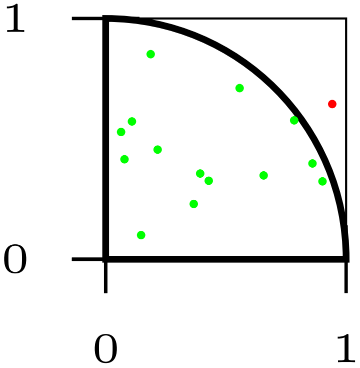
gut: 12
gesamt: 13
$\pi = 3.692$

Wie sieht das dann aus?



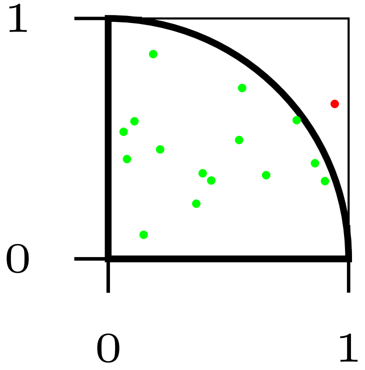
gut: 13
gesamt: 14
$\pi = 3.714$

Wie sieht das dann aus?



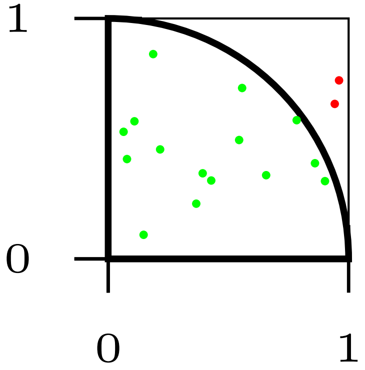
gut: 14
gesamt: 15
$\pi = 3.733$

Wie sieht das dann aus?



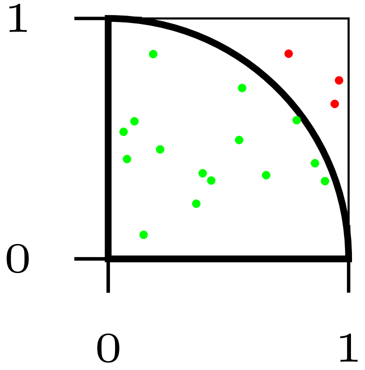
gut: 15
gesamt: 16
$\pi = 3.750$

Wie sieht das dann aus?



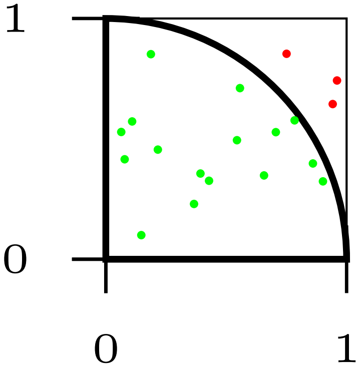
gut: 15
gesamt: 17
$\pi = 3.529$

Wie sieht das dann aus?



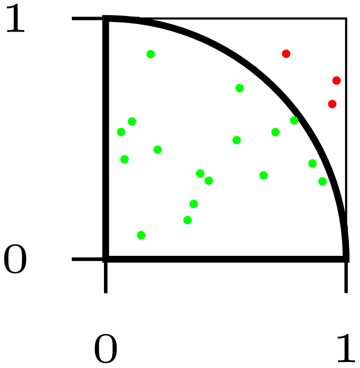
gut: 15
gesamt: 18
$\pi = 3.333$

Wie sieht das dann aus?



gut: 16
gesamt: 19
$\pi = 3.368$

Wie sieht das dann aus?



gut: 17
gesamt: 20
$\pi = 3.400$

Nur ein Bisschen Schwarze Magie

```
public double monteCarloPi(double tropfen) {
    double pi = 0, gut = 0, gesamt = tropfen;

    while (tropfen-- > 0) {
        double tropX = Math.random();
        double tropY = Math.random();

        if (tropX * tropX + tropY * tropY <= 1)
            gut++;
    }

    return pi = 4 * gut/gesamt;
}
```

Nur ein Bisschen Schwarze Magie

```
public double monteCarloPi(double tropfen) {
    double pi = 0, gut = 0, gesamt = tropfen;

    while (tropfen-- > 0) {
        double tropX = Math.random();
        double tropY = Math.random();

        if (tropX * tropX + tropY * tropY <= 1)
            gut++;
    }

    return pi = 4 * gut/gesamt;
}
```

Nur ein Bisschen Schwarze Magie

```
public double monteCarloPi(double tropfen) {
    double pi = 0, gut = 0, gesamt = tropfen;

    while (tropfen-- > 0) {
        double tropX = Math.random();
        double tropY = Math.random();

        if (tropX * tropX + tropY * tropY <= 1)
            gut++;
    }

    return pi = 4 * gut/gesamt;
}
```

Nur ein Bisschen Schwarze Magie

```
public double monteCarloPi(double tropfen) {
    double pi = 0, gut = 0, gesamt = tropfen;

    while (tropfen-- > 0) {
        double tropX = Math.random();
        double tropY = Math.random();

        if (tropX * tropX + tropY * tropY <= 1)
            gut++;
    }

    return pi = 4 * gut/gesamt;
}
```

Nur ein Bisschen Schwarze Magie

```
public double monteCarloPi(double tropfen) {
    double pi = 0, gut = 0, gesamt = tropfen;

    while (tropfen-- > 0) {
        double tropX = Math.random();
        double tropY = Math.random();

        if (tropX * tropX + tropY * tropY <= 1)
            gut++;
    }

    return pi = 4 * gut/gesamt;
}
```

Auswertung

Jeweils drei Durchläufe,
Wert von π und die benötigte Zeit sind Durchschnittswerte
(Vergleichswert: $\pi = 3.14159265358979$)

- Nach einem Schritt: $\pi = 2.\overline{6}$ in 0.156 Sekunden
- Nach zehn Schritten: $\pi = 3.2$ in 0.163 Sekunden
- Nach tausend Schritten: $\pi = 3.15\overline{3}$ in 0.172 Sekunden
- Nach einer Milliarde (10^9) Schritten: $\pi = 3.14151784$4 in 2 Minuten und 15 Sekunden
- Für genauere Ergebnisse ist viel mehr Zeit einzuplanen

Auswertung

Jeweils drei Durchläufe,
Wert von π und die benötigte Zeit sind Durchschnittswerte
(Vergleichswert: $\pi = 3.14159265358979$)

- Nach einem Schritt: $\pi = 2.\overline{6}$ in 0.156 Sekunden
- Nach zehn Schritten: $\pi = 3.2$ in 0.163 Sekunden
- Nach tausend Schritten: $\pi = 3.15\overline{3}$ in 0.172 Sekunden
- Nach einer Milliarde (10^9) Schritten: $\pi = 3.14151784$4 in 2 Minuten und 15 Sekunden
- Für genauere Ergebnisse ist viel mehr Zeit einzuplanen

Auswertung

Jeweils drei Durchläufe,
Wert von π und die benötigte Zeit sind Durchschnittswerte
(Vergleichswert: $\pi = 3.14159265358979$)

- Nach einem Schritt: $\pi = 2.\overline{6}$ in 0.156 Sekunden
- Nach zehn Schritten: $\pi = 3.2$ in 0.163 Sekunden
- Nach tausend Schritten: $\pi = 3.15\overline{3}$ in 0.172 Sekunden
- Nach einer Milliarde (10^9) Schritten: $\pi = 3.14151784$4 in 2 Minuten und 15 Sekunden
- Für genauere Ergebnisse ist viel mehr Zeit einzuplanen

Auswertung

Jeweils drei Durchläufe,
Wert von π und die benötigte Zeit sind Durchschnittswerte
(Vergleichswert: $\pi = 3.14159265358979$)

- Nach einem Schritt: $\pi = 2.\overline{6}$ in 0.156 Sekunden
- Nach zehn Schritten: $\pi = 3.2$ in 0.163 Sekunden
- Nach tausend Schritten: $\pi = 3.15\overline{3}$ in 0.172 Sekunden
- Nach einer Milliarde (10^9) Schritten: $\pi = 3.14151784$ in 2 Minuten und 15 Sekunden
- Für genauere Ergebnisse ist viel mehr Zeit einzuplanen

Auswertung

Jeweils drei Durchläufe,
Wert von π und die benötigte Zeit sind Durchschnittswerte
(Vergleichswert: $\pi = 3.14159265358979$)

- Nach einem Schritt: $\pi = 2.\overline{6}$ in 0.156 Sekunden
- Nach zehn Schritten: $\pi = 3.2$ in 0.163 Sekunden
- Nach tausend Schritten: $\pi = 3.15\overline{3}$ in 0.172 Sekunden
- Nach einer Milliarde (10^9) Schritten: $\pi = 3.14151784$ in 2 Minuten und 15 Sekunden
- Für genauere Ergebnisse ist viel mehr Zeit einzuplanen

Gibt es noch Fragen?

Danke für die Aufmerksamkeit!