

Kap. 4.5: Skiplisten
Kap. 5: Hashing

Professor Dr. Petra Mutzel

Lehrstuhl für Algorithm Engineering, LS11

15. VO

30. Mai 2006

Motivation

„Warum soll ich heute hier bleiben?“

Skiplisten & Hashing

„Was ist daran Besonderes?“

Einfacher und schneller als alles bisherige!

Überblick

- Wiederholung Skiplisten

- Analyse von Skiplisten

- Einführung in Hashverfahren

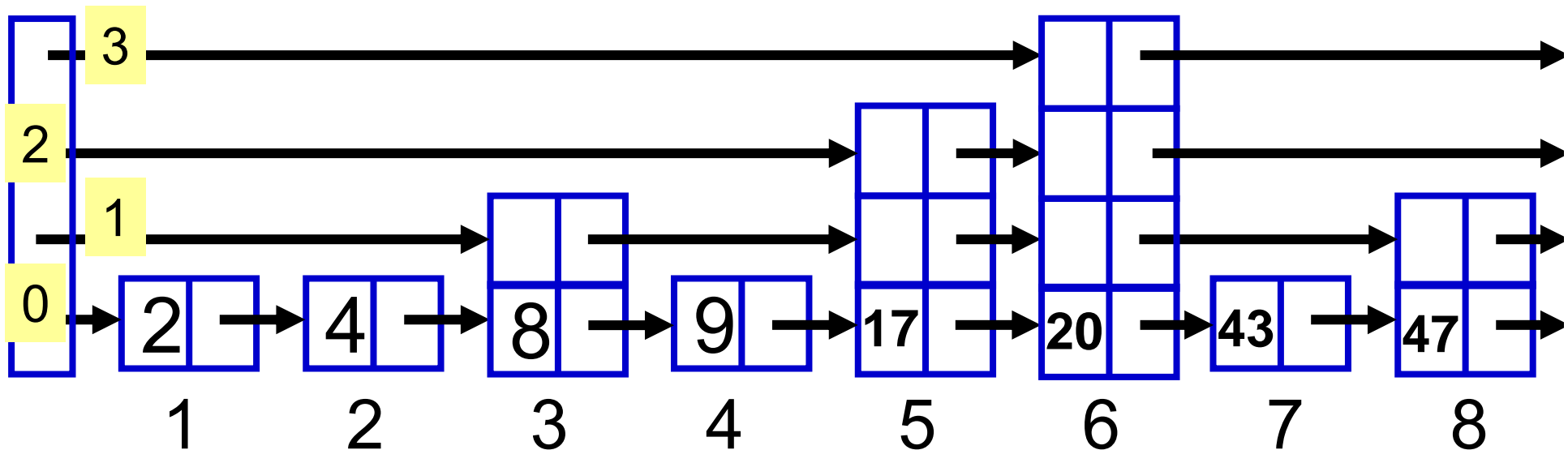
- Einfache Hashfunktionen

Randomisierte Skiplisten

- **Idee:** Die Verteilung der Höhen in der Skipliste entspricht ungefähr der Verteilung in einer perfekten Skipliste.

Wie erreicht man diese Verteilung?

Niveau i des Zeigers



Randomisierte Skiplisten

- Elemente c der Liste besitzen Höhe $h(c)$
- Diese Höhe wird zufällig bestimmt, so dass die Verteilung derjenigen einer perfekten Skipliste entspricht.
- Jedes Element c enthält einen Schlüssel und $h(c)+1$ Zeiger auf nachfolgende Listenelemente
- Anfang und Ende der Liste: je ein Pseudo-Element ohne Daten, gleiche Höhe wie maximales Element in der Liste; Schlüssel des Endelements ist größer als alle anderen

Eigenschaften randomisierter Skiplisten

Beobachtung: Skiplisten besitzen kein Gedächtnis, d.h. Elemente, die eingefügt und wieder entfernt wurden haben keinen Einfluss auf das Aussehen der aktuellen Skipliste.

Dies ist bei den anderen Datenstrukturen, wie z.B. den binären Suchbäumen anders.

Analyse randomisierter Skiplisten

Beobachtung: Wir können keine Worst Case Schranke garantieren, denn theoretisch ist es möglich, dass ein Listenelement eine beliebig hohe Höhe erhält.
Dies ist jedoch „unwahrscheinlich“.

Lösung: Wir analysieren die Erwartungswerte und zeigen, dass große Abweichungen von diesen Werten extrem unwahrscheinlich sind.

Analyse randomisierter Skiplisten

Lemma: Die Wahrscheinlichkeit, dass ein Element Höhe

- gleich h erhält, ist $(\frac{1}{2})^{h+1}$.
- mindestens h erhält, ist $(\frac{1}{2})^h$.

Beweis: Die Höhe 0 eines Elements ist nach dem 1. Münzwurf festgelegt. Die Wahrscheinlichkeit hierfür ist $\frac{1}{2}$. Nach dem h -ten Münzwurf ist Höhe $h-1$ festgelegt mit Wahrscheinlichkeit $(\frac{1}{2})^h$.

Für Mindesthöhe h muss h Mal „Zahl“ fallen.

Analyse randomisierter Skiplisten

Lemma: Der erwartete Höhe für Element ist 1.

Beweis: Die Erwartungswert für die Höhe ist

$$\sum_{0 \leq h < \infty} h \left(\frac{1}{2}\right)^{h+1} = 1.$$

Warum summiert sich diese Formel zu 1?

Unendliche geom. Reihe für $|x| < 1$: $\sum_{0 \leq h < \infty} x^h = 1/(1-x)$.

Differenzieren von L.S. und R.S.: $\sum_{0 \leq h < \infty} h x^{(h-1)} = 1/(1-x)^2$.

Beide Seiten „mal x^2 “: $\sum_{0 \leq h < \infty} h x^{(h+1)} = x^2/(1-x)^2$.

Unser Fall: $x = (1-x) = 1/2$

Analyse randomisierter Skiplisten

Lemma: Die Wahrscheinlichkeit, dass eine randomisierte Skipliste eine gewisse Höhe h überschreitet ist $\text{Prob}(H(n) \geq h) \leq \min \{ 1, n (1/2)^h \}$

Beweis: Die Wahrscheinlichkeit, dass ein Element mindestens Höhe h erhält ist $(1/2)^h$.

Sei A_i das Ereignis, dass das i -te Element eine Höhe von mind. h hat. Die Wahrscheinlichkeit, dass eines von n Elementen die Höhe mindestens h hat, ist:

$$\text{Prob}(\cup_{1 \leq i \leq n} A_i) \leq \sum_{1 \leq i \leq n} \text{Prob}(A_i) = n (1/2)^h.$$

Analyse randomisierter Skiplisten

Lemma: Die erwartete Höhe einer randomisierten Skipliste mit n Elementen ist $E(H(n)) \leq \lfloor \log n \rfloor + 2$.

Beweis: Die Erwartungswert für die Höhe ist

$$\begin{aligned}
 E(H(n)) &= \sum_{0 \leq h < \infty} h \text{Prob}(H(n)=h) = \\
 &\text{Prob}(H(n)=1) + \text{Prob}(H(n)=2) + \text{Prob}(H(n)=3) + \dots \\
 &\quad + \text{Prob}(H(n)=2) + \text{Prob}(H(n)=3) + \dots \\
 &\quad \quad + \text{Prob}(H(n)=3) + \dots \\
 &= \sum_{1 \leq h < \infty} \text{Prob}(H(n) \geq h) = \sum_{1 \leq h \leq \lfloor \log n \rfloor + 1} \text{Prob}(H(n) \geq h) + \sum_{\lfloor \log n \rfloor + 2 \leq h < \infty} \text{Prob}(H(n) \geq h) \\
 &\leq \sum_{1 \leq h < \lfloor \log n \rfloor + 1} 1 + \sum_{1 \leq i < \infty} n (1/2)^{\lfloor \log n \rfloor + 1 + i} \leq (\lfloor \log n \rfloor + 1) + \sum_{1 \leq i < \infty} (1/2)^i
 \end{aligned}$$

Analyse randomisierter Skiplisten

Lemma: Die erwartete Anzahl der Zeiger in einer randomisierten Skipliste mit n Elementen ist $E(Z(n)) \leq 2n + \lfloor \log n \rfloor + 3$.

Beweis: Das i -te Element hat eine durchschnittliche Höhe von 1, besitzt also 2 Zeiger.

Hinzu kommen die Zeiger vom Anfangselement, das sind $E(H(n)) + 1 \leq \lfloor \log n \rfloor + 3$ viele.

Analyse randomisierter Skiplisten

Lemma: Der erwartete Platzbedarf für eine randomisierte Skipliste mit n Elementen ist $O(n)$.

Beweis: Dies folgt direkt aus dem letzten Lemma.

Analyse randomisierter Skiplisten

Theorem: Die erwartete Rechenzeit für jede der Operationen SEARCH, INSERT und DELETE beträgt $O(\log n)$.

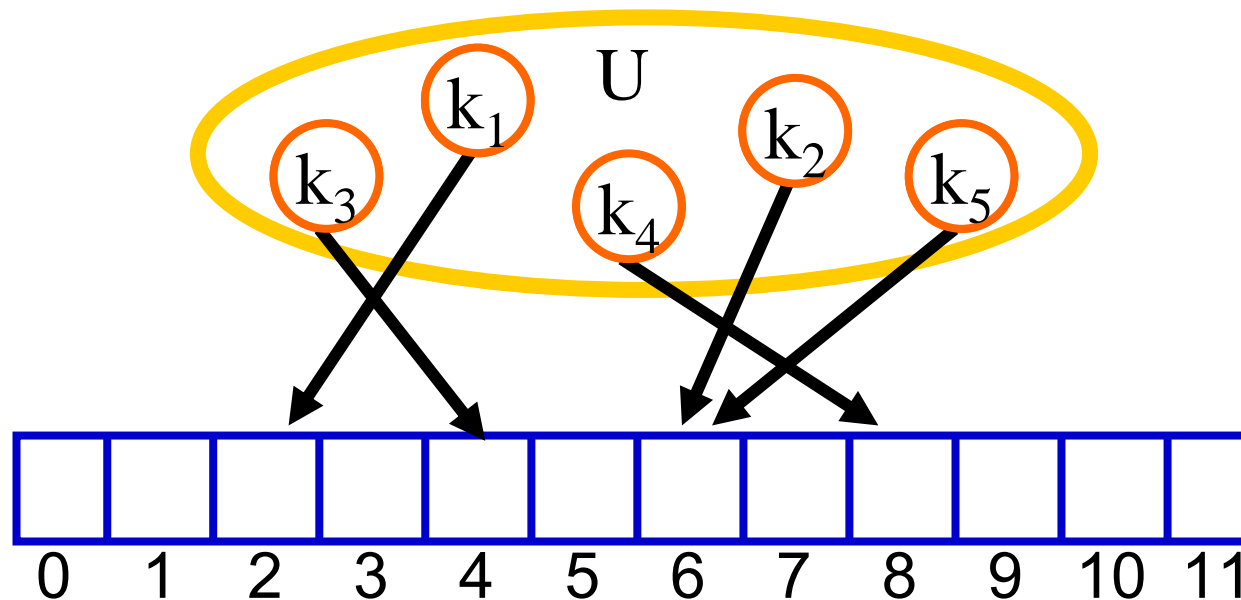
Beweis-Idee: Man zeigt, dass die erwartete Suchpfadlänge logarithmisch beschränkt ist.

- Die Idee ist, den Suchpfad rückwärts zu laufen.
- Jeder Schritt geht mit Wahrscheinlichkeit $\frac{1}{2}$ eine Ebene weiter nach oben oder waagrecht zurück.
- Man zeigt, dass es im Durchschnitt gleich viele Schritte nach oben wie nach links gibt.
- Wir wissen, dass die erwartete Höhe $\lfloor \log n \rfloor + 2$ ist.

Kap. 5: Hashing

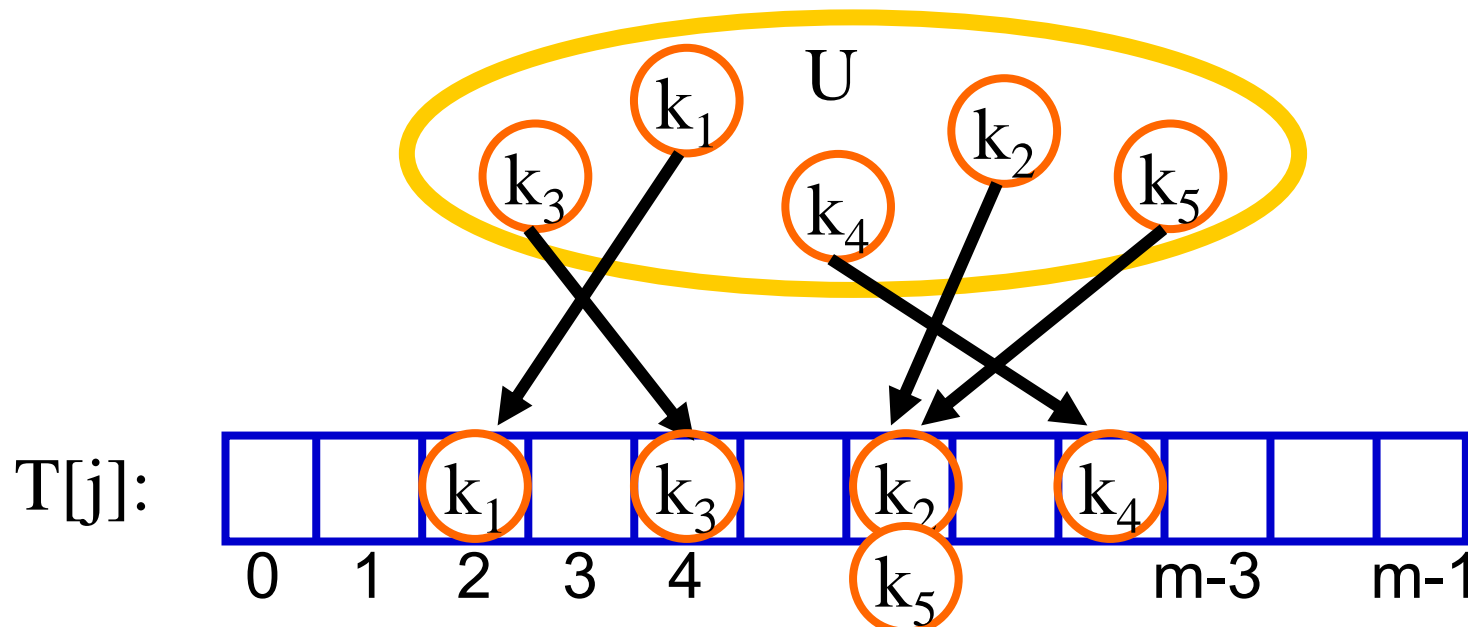
Idee von Hashing

- **Idee:** Ermittle die Position eines Elements durch eine arithmetische Berechnung statt durch Schlüsselvergleiche.
- Hashverfahren unterstützen die Dictionary-Operationen **Suchen**, **Einfügen** und **Entfernen** auf einer Menge von Elementen.



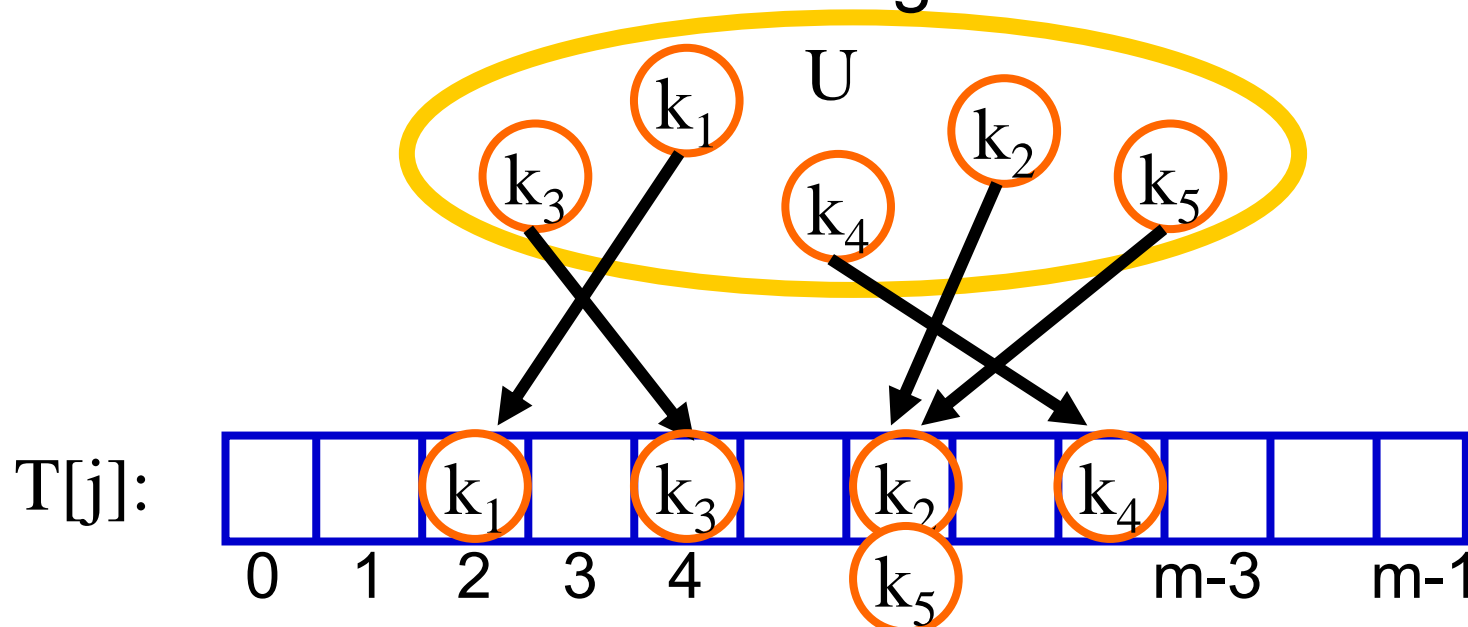
Idee von Hashing

- Sei U das Universum aus dem die Schlüssel kommen können, sei m die Tabellengröße, n die Anzahl der einzufügenden Schlüssel. Wir wählen eine Hashfunktion $h: U \rightarrow \{0, \dots, m-1\}$.
- Die Schlüssel werden in Hashtabelle $T[h(i)]$ gespeichert.



Idee von Hashing

- Hashtabelle wird als ein Array $T[0], \dots, T[m-1]$ der Größe m realisiert. Sie speichert die Einträge mit Hashadressen $0, \dots, m-1$.
- Probleme:
 - Was sind gute Hashfunktionen?
 - Kollisionsbehandlung



6.1 Zur Wahl der Hashfunktion

- Ziel: Hashadressen sollten möglichst gleich verteilt in $\{0, \dots, m-1\}$ sein.
- Hashfunktionen sollten Häufungen fast gleicher Schlüssel möglichst gleichmäßig auf den Adressbereich streuen.

- Generalannahme: Schlüssel sind nicht-negative ganze Zahlen.

- Lösung für character strings: deren ASCII-Code ist Nummer in $[0, \dots, 127]$,
- z.B. $p \cong 112$, $t \cong 116 \rightarrow pt \cong 112 \cdot 128 + 116 = 14452$
- Allgemein: für String (s_1, \dots, s_l) : $k = \sum_{1 \leq i \leq l} 128^{l-i} \text{ord}(s_i)$

6.1.1 Divisions-Rest Methode

- Die Hashfunktion der Divisions-Rest Methode ist gegeben durch: $h(k) = k \bmod m$

Eigenschaften:

- sehr schnelle Berechnung der Hashfunktion
- gute Wahl von m (Tabellengröße) ist hier sehr wichtig! Vermeide z.B.
 - $m=2^i$: ignoriert alle bis auf letzten Binärziffern
 - $m=10^i$: analog bei Dezimalzahlen
 - $m=r^i$: analog zu r -adischen Zahlen
 - $m=r^i \mp j$ für kleines j : Problem bei Vertauschung, denn sei $c := \text{Code-Diff}$, dann: $c \cdot 128 - c = c \cdot 127$

6.1.1 Divisions-Rest Methode

Z.B.: $m=2^7-1=127$:

$$pt = (112 \cdot 128 + 116) \bmod 127 = 14452 \bmod 127 = 101$$

$$tp = (116 \cdot 128 + 112) \bmod 127 = 14960 \bmod 127 = 101$$

Eigenschaften:

- sehr schnelle Berechnung der Hashfunktion
- gute Wahl von m (Tabellengröße) ist hier sehr wichtig! Vermeide z.B.
 - $m=2^i$: ignoriert alle bis auf letzten Binärziffern
 - $m=10^i$: analog bei Dezimalzahlen
 - $m=r^i$: analog zu r -adischen Zahlen
 - $m=r^i - j$ für kleines j : Problem bei Vertauschung, denn sei $c := \text{Code-Diff}$, dann: $c \cdot 128 - c = c \cdot 127$

Gute Wahl von m

- Primzahl, die
- kein $r^i \nmid j$ teilt und
- weit weg von einer Zweierpotenz ist.

Beispiel:

- Hashtabelle für ca. 700 Einträge für character strings (interpretiert als 28-adische Zahlen).
- Gute Wahl z.B. $m=701$, da $2^9=512$ und $2^{10}=1024$.

6.1.2 Multiplikationsmethode

- Die Hashfunktion der Multiplikationsmethode ist gegeben durch:

$$h(k) = \lfloor m (k \cdot A \bmod 1) \rfloor = \lfloor m(k \cdot A - \lfloor k \cdot A \rfloor) \rfloor$$

mit $0 < A < 1$.

Term $(k \cdot A - \lfloor k \cdot A \rfloor)$ heißt der „gebrochene Teil von kA “.

Eigenschaften:

- Wahl von m ist hierbei unkritisch
- Gleichmäßige Verteilung für $U = \{1, 2, \dots, n\}$ bei guter Wahl von A .

Gute Wahl für A

Irrationale Zahlen sind eine gute Wahl, denn:

Sei ξ eine irrationale Zahl. Platziert man die Punkte $\xi - \lfloor \xi \rfloor, 2\xi - \lfloor 2\xi \rfloor, \dots, n\xi - \lfloor n\xi \rfloor$ in das Intervall $[0,1)$, dann haben die $n+1$ Intervallteile höchstens drei verschiedene Längen. Außerdem fällt der nächste Punkt $(n+1)\xi - \lfloor (n+1)\xi \rfloor$ in eines der größeren Intervallteile.

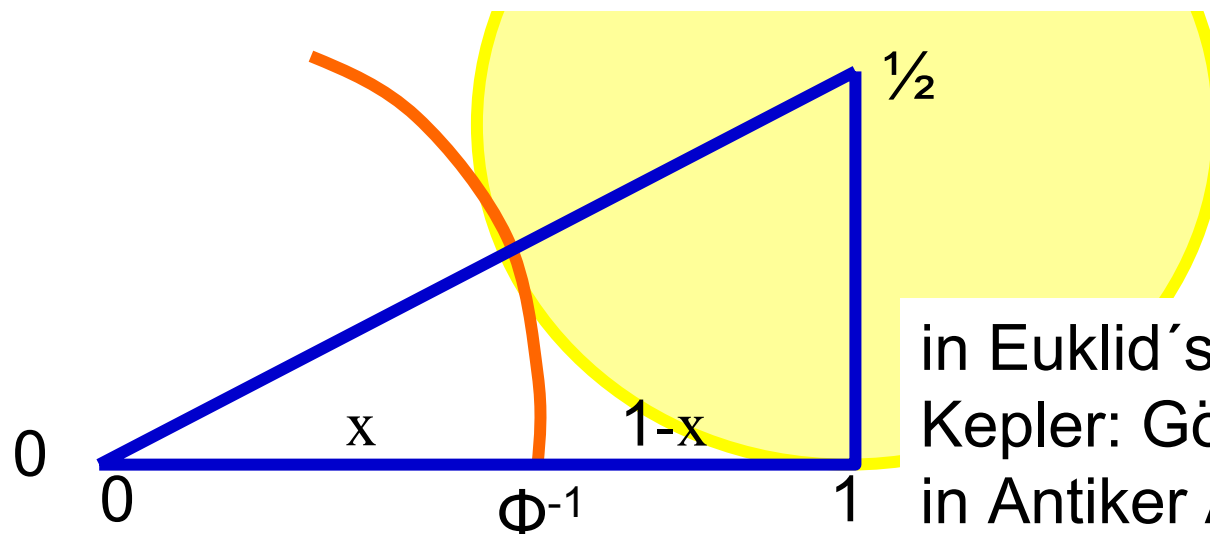
Beweis: Vera Turan Sos 1957

Beste Wahl für A nach Knuth: Der Goldene Schnitt

Der Goldene Schnitt

$\Phi^{-1} = 2/(1+\sqrt{5}) = (\sqrt{5}-1)/2 = 0,6180339887\dots$ ist bekannt als der goldene Schnitt.

Er ergibt sich durch die Zerlegung einer Strecke a in zwei positive Summanden x und $a-x$, so dass x geometrisches Mittel von a und $a-x$ ist, d.h. $x^2 = a(a-x)$.
Es gilt: $x/a = (a-x)/x$.



in Euklid's Elementen &
Kepler: Göttliche Teilung
in Antiker Architektur

Multiplikationsmethode

Beispiel: Dezimalrechnung, $k=123456$, $m=10000$, $A=\Phi^{-1}$:

$$\begin{aligned}h(k) &= \lfloor 10000 (12346 \cdot 0,61803\dots \bmod 1) \rfloor = \\ &= \lfloor 10.000 (76300,0041151\dots \bmod 1) \rfloor = \\ &= \lfloor 10.000 \cdot 0,0041151\dots \rfloor = \\ &= \lfloor 41,151\dots \rfloor = 41\end{aligned}$$

Diskussion:

- Gute Wahl für m wäre hier $m=2^i$. Dann kann $h(k)$ effizient berechnet werden (eine einfache Multiplikation und ein bis zwei Shifts).
- Empirische Untersuchungen zeigen: Divisions-Rest-Methode ist besser.

