DATENSTRUKTUREN UND ZAHLENSYSTEME

RALF HINZE

Institute of Information and Computing Sciences
Utrecht University
Email: ralf@cs.uu.nl
Homepage: http://www.cs.uu.nl/~ralf/

March, 2001

(Die Folien finden Sie unter .../~ralf/talks.html#T21.)

Eine Analogie: natürliche Zahlen und Listen

$$
\begin{array}{lcl}
\textbf{data}\ \mathit{Nat} & = & \mathit{Zero} \mid \mathit{Succ}\ \mathit{Nat} \\
\mathit{plus} & :: & \mathit{Nat} \to \mathit{Nat} \to \mathit{Nat} \\
\mathit{plus}\ \mathit{Zero}\ n_2 & = & n_2 \\
\mathit{plus}\ (\mathit{Succ}\ n_1)\ n_2 & = & \mathit{Succ}\ (\mathit{plus}\ n_1\ n_2)
\end{array}
$$

$$
\begin{array}{lcl}
\textbf{data}\ \mathit{List}\ a & = & \mathit{Nil} \mid \mathit{Cons}\ a\ (\mathit{List}\ a) \\
\mathit{append} & :: & \forall a\,.\,\mathit{List}\ a \to \mathit{List}\ a \to \mathit{List}\ a \\
\mathit{append}\ \mathit{Nil}\ l_2 & = & l_2 \\
\mathit{append}\ (\mathit{Cons}\ a\ l_1)\ l_2 & = & \mathit{Cons}\ a\ (\mathit{append}\ l_1\ l_2)
\end{array}
$$

Numerische Darstellungen

Datenstrukturen, die in Analogie zu Zahlensystemen entworfen werden, heißen *numerische Darstellungen* (engl. numerical representations).

Inkrement	Einfügen eines Elements in einen Behälter
Dekrement	Löschen eines Elements in einen Behälter
Addition	Verschmelzen zweier Behälter
Division durch 2	Halbieren eines Behälters

Diese Designtechnik eignet sich für beliebige Behältertypen: Sequenzen, Prioritätswarteschlangen etc.

Literatur: (Clancy & Knuth, *A programming and problem-solving seminar*, 1977), (Okasaki, *Purely Functional Data Structures*, 1998).

Überblick

Design von Datenstrukturen (persistente Sequenzen):

☞ binäre Listen (engl. random-access lists),

☞ schräge binäre Listen (engl. skew-binay random-access lists).

Analyse von Datenstrukturen (Suchbäume):

☞ Rot-schwarz Bäume.

Stellenwertsysteme

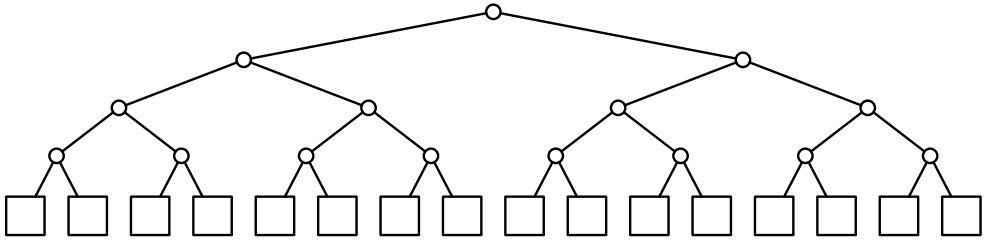
Die gebräuchlichsten Zahlensysteme sind *Stellenwertsysteme*:

$$(b_0 \dots b_{n-1}) = \sum_{i=0}^{n-1} b_i \cdot w_i \text{ mit } b_i \in B_i.$$

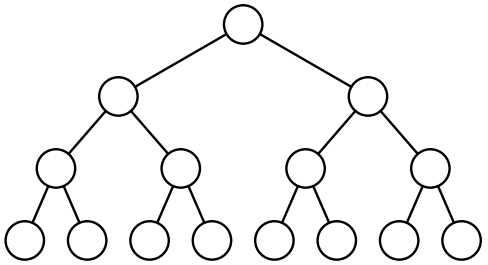
Unäres Zahlensystem: $w_i = 1$ und $B_i = \{1\}$ für alle i.

Binäres Zahlensystem: $w_i = 2^i$ und $B_i = \{0, 1\}$ für alle i.

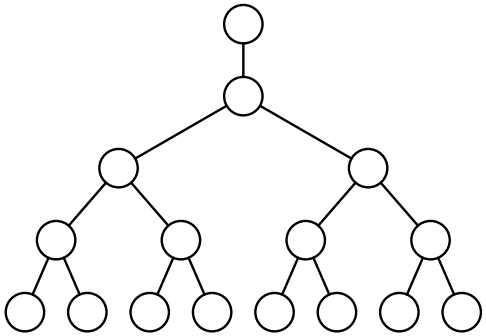
Beliebte Bausteine



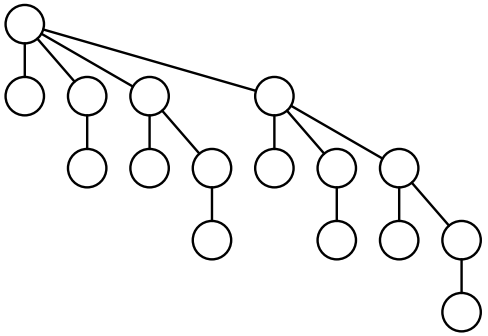
vollständiger Blattbaum



vollständiger Baum



Wimpel



Binomialbaum

Binäre Listen

Listen sind nach dem unären Zahlensystem modelliert, binäre Listen (engl. random-access lists) nach dem binären Zahlensystem.

$$\begin{array}{lcl} \textbf{data}\ \mathit{Digit} & = & \mathit{Zero} \mid \mathit{One} \\ \textbf{type}\ \mathit{Nat} & = & [\mathit{Digit}] \end{array}$$

$$\begin{array}{lcl} \textbf{data}\ \mathit{Digit}\ a & = & \mathit{Zero} \mid \mathit{One}\ (\mathit{Tree}\ a) \\ \textbf{data}\ \mathit{Tree}\ a & = & \mathit{Leaf}\ a \mid \mathit{Node}\ (\mathit{Tree}\ a)\ (\mathit{Tree}\ a) \\ \textbf{type}\ \mathit{List}\ a & = & [\mathit{Digit}\ a] \end{array}$$

Eine Ziffer mit dem Gewicht 2^i entspricht einem vollständigen Blattbaum der Höhe i.

$$
\begin{array}{lcl}
incr & :: & Nat \to Nat \\
incr\ [\,] & = & [\,One\,] \\
incr\ (Zero : ds) & = & One : ds \\
incr\ (One : ds) & = & Zero : incr\ ds
\end{array}
$$

$$
\begin{array}{lcl}
incr & :: & \forall a\,.\,Tree\ a \to List\ a \to List\ a \\
incr\ t\ [\,] & = & [\,One\ t\,] \\
incr\ t\ (Zero : ds) & = & One\ t : ds \\
incr\ t\ (One\ t' : ds) & = & Zero : incr\ (Node\ t\ t')\ ds \\
cons & :: & \forall a\,.\,a \to List\ a \to List\ a \\
cons\ a\ l & = & incr\ (Leaf\ a)\ l
\end{array}
$$

Binäre Listen — eine alternative Darstellung

Die folgende Definition erfaßt die strukturellen Invarianten von binären Listen. **NB.** '$List$' is ein *nicht-uniformer Datentyp* (engl. nested datatype).

$$\mathbf{data}\ List\ a\ =\ Nil \mid Zero\ (List\ (a,a)) \mid One\ a\ (List\ (a,a))$$

$$\begin{array}{lcl}
lookup & :: & \forall a\,.\,Int \to List\ a \to a \\
lookup\ 0\ (One\ a\ l) & = & a \\
lookup\ (n+1)\ (One\ a\ l) & = & lookup\ n\ (Zero\ l) \\
lookup\ (2*n+0)\ (Zero\ l) & = & fst\ (lookup\ n\ l) \\
lookup\ (2*n+1)\ (Zero\ l) & = & snd\ (lookup\ n\ l)
\end{array}$$

NB. '$lookup$' verwendet *polymorphe Rekursion*.

Schräge Binärzahlen

Schräge Binärzahlen: $w_i = 2^{i+1} - 1$ und $B_i = \{0, 1, 2\}$ für alle i.

Zusätzliche Einschränkung: nur die kleinste Ziffer ($\neq 0$) darf eine 2 sein.

Jede natürliche Zahl hat eine eindeutige Darstellung in diesem Zahlensystem.

$$(0), (1), (2), (01), (11), (21), (02), (001), (101), (201), (011), (111), (211), (021), (002), (0001) \ldots$$

Vorteil: Inkrement in konstanter Zeit.

Schräge binäre Listen

Schräge binäre Listen (engl. skew-binary random-access lists) basieren auf den schrägen Binärzahlen.

Wir verwenden eine dünne Darstellung (Nullen werden nicht aufgeführt).

$$\begin{array}{lcl} \textbf{type}\ \mathit{Weight} & = & \mathit{Int} \\ \textbf{type}\ \mathit{Nat} & = & [\mathit{Weight}] \end{array}$$

$$\begin{array}{lcl} \textbf{data}\ \mathit{Tree}\ a & = & \mathit{Leaf}\ a \mid \mathit{Node}\ a\ (\mathit{Tree}\ a)\ (\mathit{Tree}\ a) \\ \textbf{type}\ \mathit{List}\ a & = & [(\mathit{Weight}, \mathit{Tree}\ a)] \end{array}$$

Eine Ziffer mit dem Gewicht $2^{i+1} - 1$ entspricht einem vollständigen (Such-) Baum der Höhe i.

$$
\begin{array}{lcl}
incr & :: & Nat \to Nat \\
incr\ (w_1 : w_2 : ws) & & \\
\quad \mid w_1 == w_2 & = & (1 + w_1 + w_2) : ws \\
incr\ ws & = & 1 : ws
\end{array}
$$

$$
\begin{array}{lcl}
cons & :: & \forall a\,.\,a \to List\ a \to List\ a \\
cons\ a\ ((w_1, t_1) : (w_2, t_2) : ws) & & \\
\quad \mid w_1 == w_2 & = & (1 + w_1 + w_2, Node\ a\ t_1\ t_2) : ws \\
cons\ a\ ws & = & (1, Leaf\ a) : ws
\end{array}
$$

Analye von Rot-schwarz-Bäumen

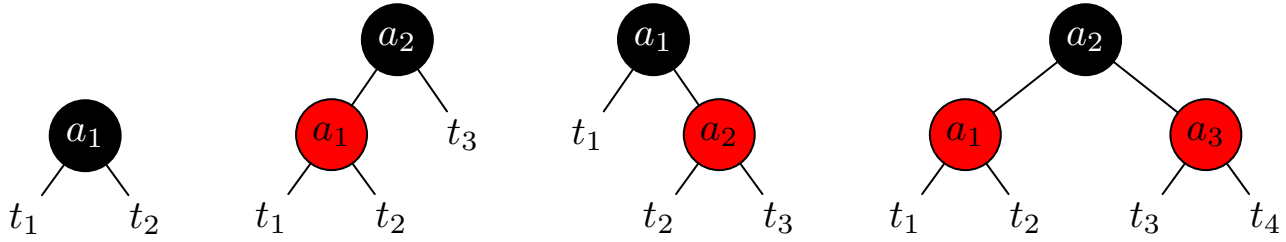
Problem: Wir fügen eine absteigend geordnete Folge von Schlüsseln in einen anfangs leeren Rot-schwarz-Baum ein.

Welche Form hat der so konstruierte Baum?

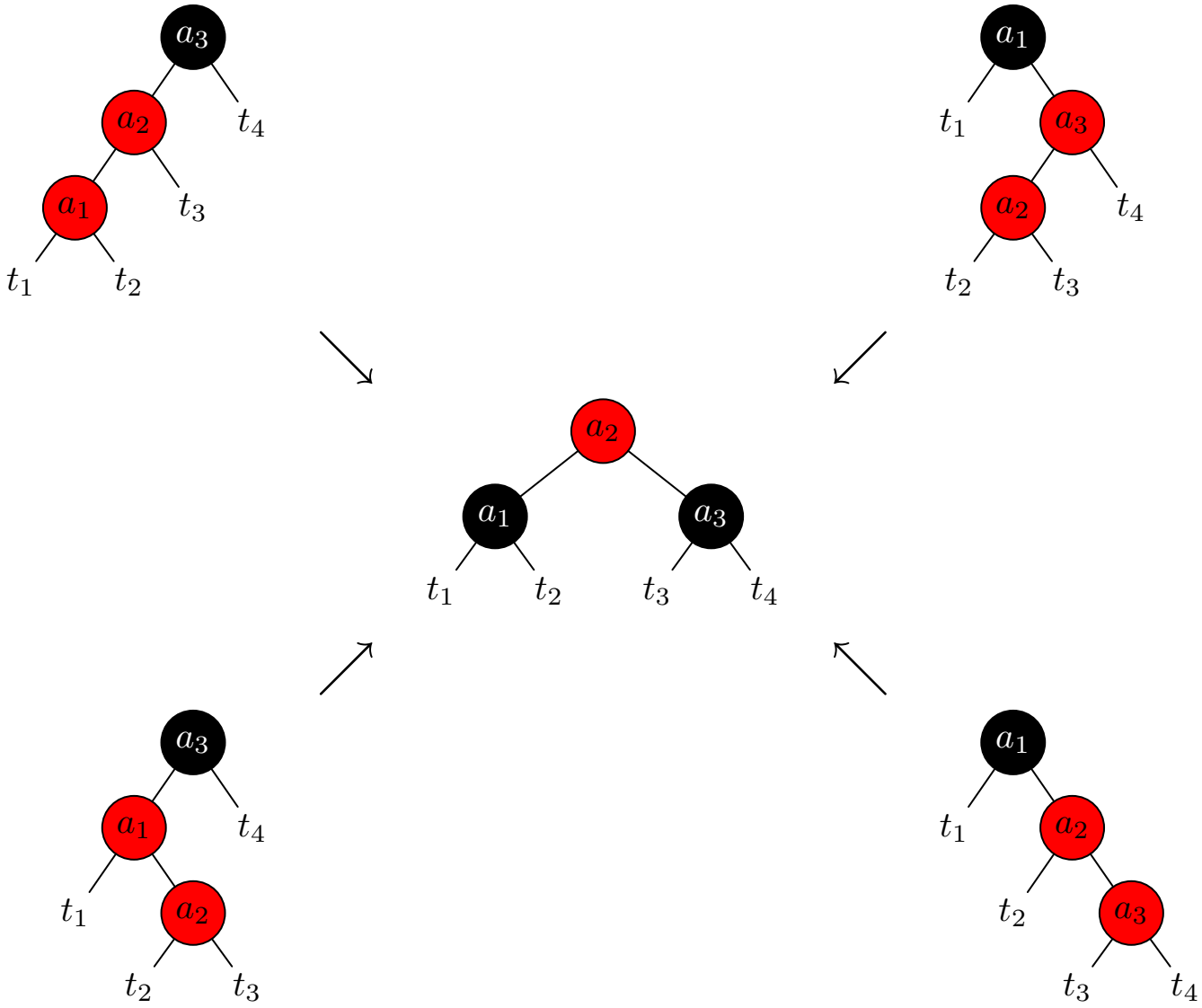
Rot-schwarz-Bäume

Rot-schwarz-Bäume wurden von Bayer (1972) unter dem Namen *symmetric binary B-trees* als binäre Darstellung von 2-3-4-Bäumen entwickelt.

In einem Rot-schwarz-Baum werden die 3- und die 4-Knoten durch kleine Binärbäume repräsentiert, die aus einer schwarzen Wurzel und ein oder zwei roten Hilfsknoten bestehen.

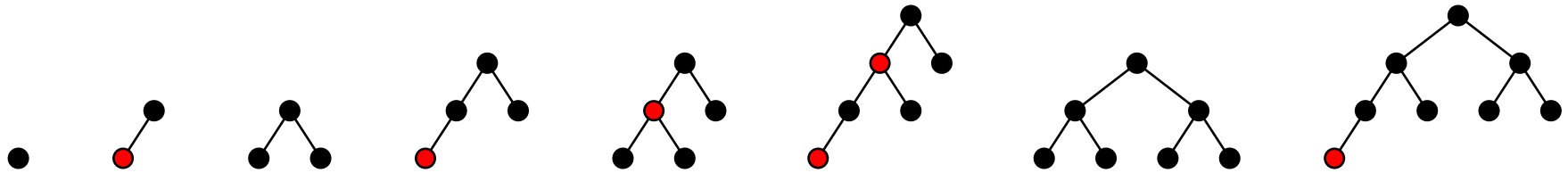


Rot-Bedingung: Jeder rote Knoten hat einen schwarzen Vorgänger.
Schwarz-Bedingung: Jeder Pfad von der Wurzel zu einem Blatt enthält die gleiche Anzahl schwarzer Knoten (Schwarzhöhe).


a_3
a_2
t_4
a_1
t_3
t_1
t_2
a_1
t_1
a_3
a_2
t_4
t_2
t_3
a_2
a_1
a_3
t_1
t_2
t_3
t_4
a_3
a_1
t_4
t_1
a_2
t_2
t_3
a_1
t_1
a_2
t_2
a_3
t_3
t_4

Analyse des wiederholten Einfügens

Die folgenden Bäume entstehen, wenn i absteigend geordnete Schlüssel in einen anfangs leeren Baum eingefügt werden $(1 \leqslant i \leqslant 8)$.

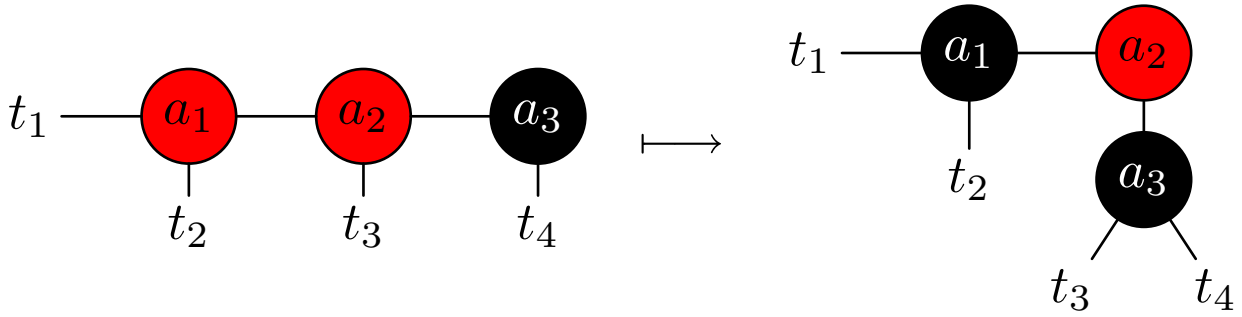


Beim Einfügen wird immer das *linke Rückgrat* des Baumes bis zum linkesten Blatt durchlaufen.

Beobachtungen

Alle Knoten unterhalb des Rückgrats sind schwarz.

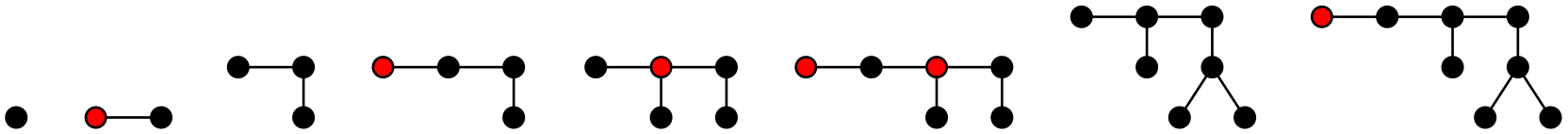
Betrachten wir noch einmal die Balancierungsoperation – zur Verdeutlichung zeichnen wir das linke Rückgrat waagerecht.



☞ Aufgrund der Schwarz-Bedingung müssen die Bäume unterhalb des Rückgrats vollständig ausgeglichen sein. Damit entsprechen die generierten Rot-schwarz-Bäume Listen von *Wimpeln*.

Beobachtungen

Zeichnen wir die obigen Beispiele noch einmal neu.



Sei r die Höhe des rechtesten Wimpels. Aufgrund der Schwarz-Bedingung müssen Wimpel der Höhe i entweder ein- oder zweimal auftreten für alle i mit $0 \leqslant i \leqslant r$.

☞ Da ein Wimpel der Höhe r genau 2^r Schlüssel enthält, korrespondieren die generierten Rot-schwarz-Bäume zu Binärzahlen mit den Ziffern 1 und 2.

Das 1-2-Binärsystem

1-2-Binärsystem: $w_i = 2^i$ und $B_i = \{1, 2\}$ für alle i.

Jede natürliche Zahl hat eine eindeutige Darstellung in diesem Zahlensystem.

$$(), (1), (2), (11), (21), (12), (22), (111), (211), (121), (221), (112),$$
$$(212), (122), (222), (1111) \ldots$$

Konstruktion von Rot-schwarz-Bäumen

Die Analogie zum 1-2-Zahlensystem kann ausgenutzt werden, um einen Rot-schwarz-Baum aus einer geordneten Folge von Elementen in linearer Zeit zu konstruieren.

$$\begin{array}{lcl} \textbf{data}\ \mathit{Digit} & = & \mathit{One} \mid \mathit{Two} \\ \textbf{type}\ \mathit{Nat} & = & [\mathit{Digit}] \end{array}$$

$$\begin{array}{lcl} \textbf{data}\ \mathit{Digit}\ a & = & \mathit{One}\ a\ (\mathit{Tree}\ a) \mid \mathit{Two}\ a\ (\mathit{Tree}\ a)\ a\ (\mathit{Tree}\ a) \\ \textbf{data}\ \mathit{Tree}\ a & = & \mathit{Empty} \mid \mathit{Node}\ (\mathit{Tree}\ a)\ a\ (\mathit{Tree}\ a) \\ \textbf{type}\ \mathit{Set}\ a & = & [\mathit{Digit}\ a] \end{array}$$

Konstruktion von Rot-schwarz-Bäumen

$$
\begin{array}{lcl}
incr & :: & Nat \to Nat \\
incr\ [\,] & = & [\,One\,] \\
incr\ (One : ds) & = & Two : ds \\
incr\ (Two : ds) & = & One : incr\ ds
\end{array}
$$

$$
\begin{array}{lcl}
incr & :: & \forall a\,.\,a \to Tree\ a \to Set\ a \to Set\ a \\
incr\ a\ t\ [\,] & = & [\,One\ a\ t\,] \\
incr\ a_1\ t_1\ (One\ a_2\ t_2 : ps) & = & Two\ a_1\ t_1\ a_2\ t_2 : ps \\
incr\ a_1\ t_1\ (Two\ a_2\ t_2\ a_3\ t_3 : ps) & & \\
\quad = & & One\ a_1\ t_1 : incr\ a_2\ (Node\ t_2\ a_3\ t_3)\ ps \\
insert & :: & \forall a\,.\,a \to Set\ a \to Set\ a \\
insert\ a\ ps & = & incr\ a\ Empty\ ps
\end{array}
$$

Zusammenfassung

Zahlensysteme sind ein ideales Hilfsmittel, um systematisch Datenstrukturen zu konstruieren und zu analysieren.

Weitere Beispiele:

- Binomialwarteschlangen (Vuillemin, 1978; Hinze, 1999): Binärsystem,
- Optimale Prioritätswarteschlangen (Okasaki, 1996): schräges Binärsystem,
- Katenierbare Sequenzen (Okasaki, 1997): redundantes Binärsystem,
- Linksvollständige Bäume (Sack & Strothotte 1990, Hinze 1999): 1-2 Binärsystem,
- Zufallspermutationen (Knuth, 1997): 'factorial number system', $w_i = i!$ und $B_i = \{0, \ldots, i\}$ für alle i.