Diskrete Mathematik

Univ.-Prof. Dr. Goulnara ARZHANTSEVA

SS 2018

Überblick: Vorlesung

Einführung in die Grundbegriffe der Diskreten Mathematik

1. Einfache und abzählende Kombinatorik: Stichproben, Permutationen, Partitionen
2. Erzeugende Funktionen, Lösen von Rekursionen
3. Das Prinzip der Inklusion und Exklusion, Suchen und Sortieren
4. Graphen und Netzwerke

Average–Case Analyse

Für praktische Anwendungen ist es meist von größerer Bedeutung, die durchschnittliche Dauer eines Algorithmus zu bestimmen: Das Average–Case Analyse eines Algorithmus.

Gegeben ist ein q–Baum $\mathbf{W}$ mit n Blättern, die mit $1, 2, \ldots, n$ numeriert sind. Jedem Blatt i ist eine bestimmte Wahrscheinlichkeit $\mathbf{P}(i)$ zugeordnet; mit $0 \leq \mathbf{P}(i) \leq 1$ und $\sum_{i=1}^{n} \mathbf{P}(i) = 1$. Sei $\ell(i)$ die Länge des Blattes i, dann interessiert uns die erwartete Länge der Blätter des Baumes $\mathbf{W}$, die wir mit $\overline{L}(\mathbf{W})$ bezeichnen:

$$\overline{L}(\mathbf{W}) = \sum_{i=1}^{n} \mathbf{P}(i)\, \ell(i)\,.$$

Average–Case Analyse

Für praktische Anwendungen ist es meist von größerer Bedeutung, die durchschnittliche Dauer eines Algorithmus zu bestimmen: Das Average–Case Analyse eines Algorithmus.

Gegeben ist ein q–Baum $\mathbf{W}$ mit n Blättern, die mit $1, 2, \ldots, n$ numeriert sind. Jedem Blatt i ist eine bestimmte Wahrscheinlichkeit $\mathbf{P}(i)$ zugeordnet; mit $0 \leq \mathbf{P}(i) \leq 1$ und $\sum_{i=1}^{n} \mathbf{P}(i) = 1$. Sei $\ell(i)$ die Länge des Blattes i, dann interessiert uns die erwartete Länge der Blätter des Baumes $\mathbf{W}$, die wir mit $\overline{L}(\mathbf{W})$ bezeichnen:

$$\overline{L}(\mathbf{W}) = \sum_{i=1}^{n} \mathbf{P}(i)\,\ell(i)\,.$$

Wie klein kann $\overline{L}(\mathbf{W})$ werden, wenn $\mathbf{W}$ alle möglichen q–Bäume mit n Blättern durchläuft?

Kraftsche Ungleichung

Satz: Kraftsche Ungleichung

(1) Sei **W** ein q–Baum mit n Blättern $1, 2, \ldots, n$, deren Längen durch $\ell(1), \ell(2), \ldots, \ell(n)$ gegeben sind. Dann ist

$$\sum_{i=1}^{n} q^{-\ell(i)} \leq 1,$$

und Gleichheit gilt genau dann, wenn **W** ein vollständiger q–Baum ist.

(2) Seien $\ell(1), \ell(2), \ldots, \ell(n)$ in $\mathbb{Z}^{+}$ gegeben mit $\sum_{i=1}^{n} q^{-\ell(i)} \leq 1$. Dann existiert ein q–Baum **W** mit n Blättern, deren Längen $\ell(1), \ldots, \ell(n)$ sind.

Kraftsche Ungleichung: Beweis

Beweis:

Für $q = 1$ sind alle diese Aussagen trivial. Sei also $q \geq 2$.

ad (1): Wenn wir einen q–Baum $\mathbf{W}$ gegeben haben, der nicht vollständig sein sollte, dann können wir ihn durch Anhängen von weiteren Blättern an ungesättigte innere Knoten zu einem vollständigen Baum $\mathbf{W}'$ machen. Dieser hat dann n' Blätter mit $n' \geq n$. Da die Summe $\sum_{i\geq 1} q^{-\ell(i)}$ dabei sicher zunimmt, genügt es zu zeigen, daß für einen vollständigen q–Baum die Gleichheit gilt.

Kraftsche Ungleichung: Beweis

Beweis:

Für $q = 1$ sind alle diese Aussagen trivial. Sei also $q \geq 2$.

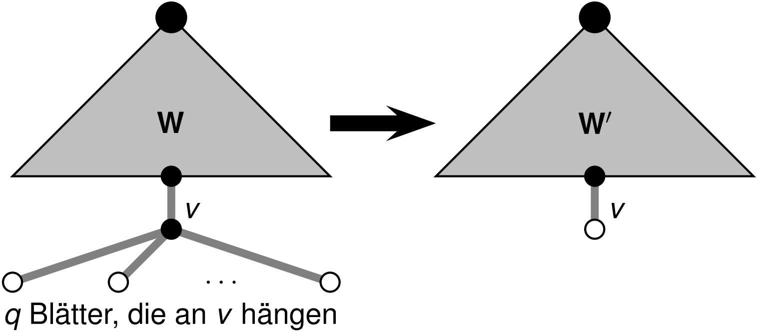
ad (1): Wenn wir einen q–Baum **W** gegeben haben, der nicht vollständig sein sollte, dann können wir ihn durch Anhängen von weiteren Blättern an ungesättigte innere Knoten zu einem vollständigen Baum $\mathbf{W}'$ machen. Dieser hat dann n' Blätter mit $n' \geq n$. Da die Summe $\sum_{i \geq 1} q^{-\ell(i)}$ dabei sicher zunimmt, genügt es zu zeigen, daß für einen vollständigen q–Baum die Gleichheit gilt.

Das beweisen wir mit Induktion nach der Anzahl der Blätter n.

Falls $n = 1$ ist, dann besteht der q–Baum nur aus der Wurzel, die gleichzeitig ein Blatt ist (denn $q \geq 2$); für diesen Baum gilt $\sum_{i=1}^{1} q^{-\ell(i)} = q^0 = 1$.

Kraftsche Ungleichung: Beweis (aus Skriptum)

Für den Induktionsschritt sei $\mathbf{W}$ ein vollständiger q–Baum mit n Blättern. Wir betrachten einen Knoten in $v \in \mathbf{W}$, an dem nur Blätter hängen (einen solchen Knoten muß es geben: Wähle z.B. einen inneren Knoten maximaler Länge). Da $\mathbf{W}$ vollständig ist, hängen an v q Blätter $v_1, \ldots, v_q$, die wir alle entfernen; d.h., wir betrachten den Teilgraphen $\mathbf{W}'$, der durch $V(\mathbf{W}) \setminus \{v_1, \ldots, v_q\}$ induziert wird.



Kraftsche Ungleichung: Beweis

$\mathbf{W}'$ ist wieder ein vollständiger q–Baum, in dem der Knoten v nun ein Blatt ist; er hat also $n - q + 1$ Blätter. Sei $\ell = \ell(v_1) = \cdots = \ell(v_q)$ die Länge der entfernten Blätter in $\mathbf{W}$. Dann gilt in $\mathbf{W}$:

$$\sum_{i=1}^{n} q^{-\ell(i)} = \sum_{i \in [n] \setminus \{v_1, \ldots, v_q\}} q^{-\ell(i)} + q \cdot q^{-\ell}. \tag{1}$$

Die rechte Summe in (1) ist aber einfach die Summe über die $n - q + 1$ Blätter von $\mathbf{W}'$, also gleich 1 nach Induktion.

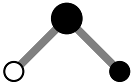
Kraftsche Ungleichung: Beweis

ad (2): Der gesuchte q–Baum kann sukzessive konstruiert werden: Wir illustrieren die Vorgangsweise anhand eines Beispiels. Sei $q = 2$, $n = 6$, $\ell(1) = 1$, $\ell(2) = 2$, $\ell(3) = 3$, $\ell(4) = \ell(5) = 5$, $\ell(6) = 6$. Es gilt in der Tat

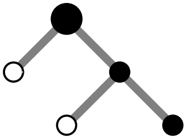
$$2^{-1} + 2^{-2} + 2^{-3} + 2 \cdot 2^{-5} + 2^{-6} = \frac{61}{64} \leq 1.$$

Kraftsche Ungleichung: Beweis (aus Skriptum)

Wir beginnen mit einer Wurzel und zeichnen $q = 2$ Kanten nach unten, an denen jeweils ein Knoten hängt. Da eines der Blätter die Länge 1 haben soll, machen wir einen dieser Knoten zu einem Blatt:

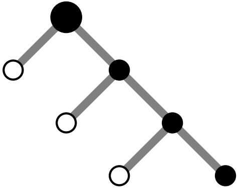


Beim anderen Knoten können wir weiter verzweigen. Da ein Blatt die Länge zwei haben soll, machen wir einen der neuen Knoten zu einem Blatt:

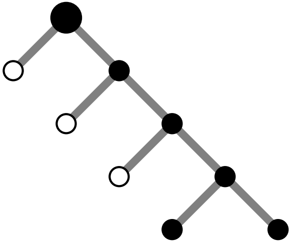


Kraftsche Ungleichung: Beweis (aus Skriptum)

Beim anderen Knoten können wir wieder verzweigen. Ein Blatt soll die Länge 3 haben, also machen wir einen der neuen Knoten zu einem Blatt:

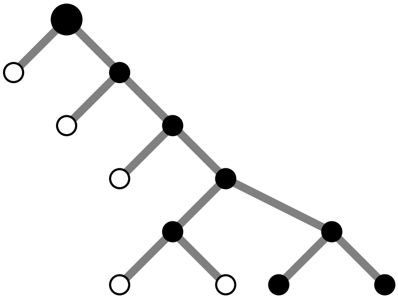


Beim anderen Knoten können wir wieder verzweigen. Da es kein Blatt der Länge 4 geben soll, belassen wir die beiden neuen Knoten als innere Knoten:



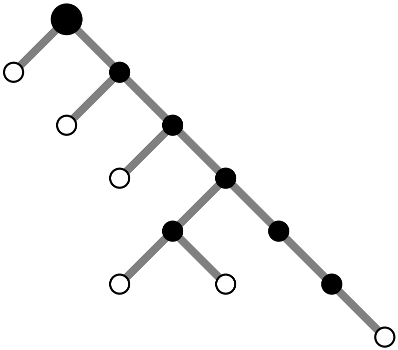
Kraftsche Ungleichung: Beweis (aus Skriptum)

Wir können nun bei beiden noch Knoten verzweigen. Es soll zwei Blätter der Länge 5 geben, daher machen wir die ersten beiden durch die Verzweigungen entstandenen Knoten zu Blättern:



Kraftsche Ungleichung: Beweis (aus Skriptum)

Wir müssen jetzt nur noch ein Blatt der Länge 6 realisieren: Wir hängen es an einen der freien Knoten an; eine Verzweigung ist nun überflüssig und kann wieder entfernt werden:



Kraftsche Ungleichung: Beweis

Die Vorgangsweise ist an sich klar.

Könnte es nicht passieren, daß an irgendeiner Stelle mehr Blätter einer bestimmten Länge erzeugt werden müssen als “freie” Knoten zur Verfügung stehen?

Kraftsche Ungleichung: Beweis

Die Vorgangsweise ist an sich klar.

Könnte es nicht passieren, daß an irgendeiner Stelle mehr Blätter einer bestimmten Länge erzeugt werden müssen als “freie” Knoten zur Verfügung stehen?

Wenn wir die Anzahl der Blätter mit Länge k mit $w(k)$ bezeichnen (d.h., wir wollen einen q–Baum mit $w(k)$ Blättern der Länge k konstruieren, mit $k = 0, 1, \ldots, L$, wobei L die maximale Länge eines Blatts ist), dann können wir die Voraussetzung $\sum_{i=1}^{n} q^{-\ell(i)} \leq 1$ so schreiben:

$$\sum_{k=0}^{L} w(k)\, q^{-k} \leq 1,$$

oder äquivalent

$$w(0)\, q^{L} + w(1)\, q^{L-1} + \cdots + w(L-1)\, q + w(L) \leq q^{L}. \tag{2}$$

Kraftsche Ungleichung: Beweis

Angenommen, wir sind mit der oben illustrierten Vorgangsweise soweit gekommen, daß alle Blätter der Längen $0, 1, \ldots, k$ bereits erzeugt wurden; nun müßten wir also $w(k+1)$ Blätter der Länge $k+1$ anhängen.

Durch “vollständiges Verzweigen” von der Wurzel aus bis zum “Niveau” $k+1$ wären an sich q^{k+1} “freie” Knoten vorhanden.

Einige dieser Äste stehen aber nicht mehr zur Verfügung, denn wir haben bereits Blätter kürzerer Länge eingetragen; genauer gesagt: Wenn ein Blatt der Länge m eingetragen wurde, dann verringert dies die Anzahl der “freien” Knoten auf Niveau $k+1$ um q^{k+1-m}.

Kraftsche Ungleichung: Beweis

Insgesamt stehen uns also nur mehr

$$q^{k+1} - w(0)\, q^{k+1} - w(1)\, q^{k} - \cdots - w(k)\, q$$

freie Knoten auf Niveau $k + 1$ zur Verfügung, und wenn wir mit unserer Prozedur fortfahren wollen, müßte

$$w(k+1) \leq q^{k+1} - w(0)\, q^{k+1} - w(1)\, q^{k} - \cdots - w(k)\, q$$

gelten. Dies ist aber äquivalent mit

$$w(0)\, q^{L} + w(1)\, q^{L-1} + \cdots + w(k)\, q^{L-k} + w(k+1)\, q^{L-k-1} \leq q^{L},$$

und diese Ungleichung ist nach Voraussetzung (2) richtig. ■

Hilfssatz aus der Analysis

Lemma: Hilfssatz aus der Analysis — cf. Log Sum Ungleichung

Es seien $s_1, s_2, \ldots, s_n$ und $y_1, y_2, \ldots, y_n$ positive reelle Zahlen mit $\sum_{i=1}^n s_i \leq \sum_{i=1}^n y_i$. Für $q > 1$ gilt dann

$$\sum_{i=1}^{n} y_i \log_q \frac{y_i}{s_i} \geq 0,$$

und Gleichheit gilt genau dann, wenn $s_i = y_i$ für alle i gilt.

Damit können wir nun den Hauptsatz der Informationstheorie beweisen.

Hauptsatz der Informationstheorie

Satz: Hauptsatz der Informationstheorie

Sei $n \geq 1$, $q \geq 2$, und sei $(p_1, p_2, \ldots, p_n)$ mit $p_i < 1$ für alle i eine Wahrscheinlichkeitsverteilung auf den Blättern von q–Bäumen **W** mit n Blättern. Dann gilt

$$-\sum_{i=1}^{n} p_i \log_q p_i \leq \min_{\mathbf{W}} \overline{L}(\mathbf{W}) < -\sum_{i=1}^{n} p_i \log_q p_i + 1,$$

wobei $0 \log_q 0$ als 0 zu interpretieren ist.

Hauptsatz der Informationstheorie: Beweis

Beweis: Wir nehmen zunächst $1 > p_i > 0$ für alle i an. Sei $\mathbf{W}$ irgendein q–Baum mit n Blättern, und seien die Blattlängen $\ell(1), \ell(2), \ldots, \ell(n)$. Aus der Kraftschen Ungleichung wissen wir, daß

$$\sum_{i=1}^{n} q^{-\ell(i)} \leq 1 = p_1 + p_2 + \cdots + p_n$$

gilt. Wir wählen nun in Lemma $s_i = q^{-\ell(i)}$ und $y_i = p_i$. Das liefert

$$\sum_{i=1}^{n} p_i \log_q \frac{p_i}{q^{-\ell(i)}} \geq 0$$

oder nach Umformung

$$\sum_{i=1}^{n} p_i(\log_q p_i + \ell(i)) \geq 0.$$

Das impliziert $\overline{L}(\mathbf{W}) = \sum_{i=1}^{n} p_i \ell(i) \geq -\sum_{i=1}^{n} p_i \log_q p_i$. Damit ist die linke Ungleichung bewiesen.

Hauptsatz der Informationstheorie: Beweis

Um die rechte Ungleichung zu beweisen, müssen wir einen q–Baum **W** mit n Blättern finden, der

$$\overline{L}(\mathbf{W}) < -\sum_{i=1}^{n} p_i \log_q p_i + 1$$

erfüllt.

Dazu definieren wir natürliche Zahlen $\ell(i)$ durch

$$-\log_q p_i \leq \ell(i) < -\log_q p_i + 1.$$

(Diese sind wegen $0 < p_i \leq 1$ tatsächlich wohldefiniert.)

Es gilt also $q^{-\ell(i)} \leq p_i$ für alle i und somit $\sum_{i=1}^{n} q^{-\ell(i)} \leq \sum_{i=1}^{n} p_i = 1$.

Hauptsatz der Informationstheorie: Beweis

Nach dem zweiten Teil der Kraftschen Ungleichung wissen wir, daß es dann einen q–Baum $\mathbf{W}$ mit n Blättern geben muß, dessen Blätter die Längen $\ell(1), \ell(2), \ldots, \ell(n)$ haben. Für diesen Baum gilt dann:

$$\begin{aligned}\overline{L}(\mathbf{W}) = \sum_{i=1}^{n} p_i \ell(i) &< \sum_{i=1}^{n} p_i(-\log_q p_i + 1) \\ &= -\sum_{i=1}^{n} p_i \log_q p_i + \sum_{i=1}^{n} p_i \\ &= -\sum_{i=1}^{n} p_i \log_q p_i + 1.\end{aligned}$$

Schließlich kann man überlegen, daß mit der Konvention $0 \log_q 0 = 0$ die obigen Argumente auch für den Fall modifiziert werden können, daß eine oder mehrere Wahrscheinlichkeiten gleich 0 sind. ■

Hauptsatz der Informationstheorie: Abschluss

Der Satz besagt, daß die minimal mögliche durchschnittliche Laufzeit eines Suchalgorithmus, der durch einen q–Baum beschrieben werden kann, ziemlich genau gleich $-\sum_{i=1}^{n} p_i \log_q p_i$ ist:

Diese Größe ist daher sehr wichtig für die Analyse von Suchalgorithmen. Im Fall der Gleichverteilung (also für $p_1 = p_2 = \cdots = p_n = 1/n$) erhalten wir

$$-\sum_{i=1}^{n} \frac{1}{n} \log_q \frac{1}{n} = -\log_q \frac{1}{n} = \log_q n.$$

Das ist “im wesentlichen” derselbe Wert wie in Satz: Die Länge des Wurzelbaumes:

Im Fall der Gleichverteilung gibt es also “praktisch keinen” Unterschied zwischen der worst case analysis und der average case analysis.

Entropie

Definition: Entropie

Für $q = 2$ nennt man die Größe

$$-\sum_{i=1}^{n} p_i \log_2 p_i,$$

die Entropie der Wahrscheinlichkeitsverteilung $(p_1, p_2, \ldots, p_n)$:

Sie gibt die durchschnittliche Anzahl an Ja/Nein–Fragen an, die man stellen muß, um die volle Information zu erhalten.

Hauptsatz der Informationstheorie: Optimale Beispiel

Der Hauptsatz der Informationstheorie gibt eine untere Schranke für die erwartete Laufzeit eines Suchalgorithmus.

Der Huffman–Algorithmus ist ein Beispiel für einen Algorithmus, mit dem ein q–Baum konstruiert wird, der einem Suchalgorithmus mit der minimalen erwarteten Laufzeit entspricht.

Hauptsatz der Informationstheorie: Optimale Beispiel

Der Hauptsatz der Informationstheorie gibt eine untere Schranke für die erwartete Laufzeit eines Suchalgorithmus.

Der Huffman–Algorithmus ist ein Beispiel für einen Algorithmus, mit dem ein q–Baum konstruiert wird, der einem Suchalgorithmus mit der minimalen erwarteten Laufzeit entspricht.

Wir werden der später präsentieren.

Sortieralgorithmen: Allgemeine Aufgabe

Gegeben sei eine Liste von n verschiedene reellen Zahlen

$$(a_1, a_2, \ldots, a_n)\,.$$

Man ordne diese Zahlen der Größe nach auf möglichst effiziente Weise, und zwar unter Verwendung der Tests "$a_i > a_j$?".

Sortieralgorithmen: Allgemeine Aufgabe

Gegeben sei eine Liste von n verschiedene reellen Zahlen

$$(a_1, a_2, \ldots, a_n)\,.$$

Man ordne diese Zahlen der Größe nach auf möglichst effiziente Weise, und zwar unter Verwendung der Tests “$a_i > a_j$?”.

Wenn wir die der Größe nach geordnete Liste mit $(a_{\pi(1)}, a_{\pi(2)}, \ldots, a_{\pi(n)})$ bezeichnen, dann wird klar, daß das Sortieren der Liste gleichbedeutend damit ist, die Permutation π^{-1} (also die Anordnung der ursprünglichen Liste $(a_1, a_2, \ldots, a_n)$) zu bestimmen.

Wir nehmen der Einfachheit halber an, daß alle möglichen Anordnungen der Elemente $a_1, a_2, \ldots, a_n$ gleich wahrscheinlich sind.

Sortieralgorithmen: Theoretische Schranke

Der Suchraum umfaßt hier alle $n!$ Permutationen von $a_1, a_2, \ldots, a_n$, und q ist hier 2. Die Anzahl der notwendigen Tests nach unten begrenzt ist durch die informationstheoretische Schranke

$$\lceil \log_2 n! \rceil$$

Wir verwenden die aus der Analysis bekannte Stirlingsche Formel

$$n! \sim \left(\frac{n}{e}\right)^n \sqrt{2\pi n}$$

und erhalten damit

$$\lceil \log_2 n! \rceil \sim \log_2 \left(\left(\frac{n}{e}\right)^n \sqrt{2\pi n} \right) = n \log_2 n - n \log_2 e + \frac{1}{2} \log_2(2\pi n).$$

Für unsere Zwecke ignorieren wir den letzten Term und schreiben also

$$\lceil \log_2 n! \rceil \sim n \log_2 n - n \log_2 e. \tag{3}$$

Sortieralgorithmen: Beispiel

Sei $n = 4$. Die informationstheoretische Schranke liefert $\lceil \log_2 24 \rceil = 5$: Theoretisch ist es also möglich, einen Algorithmus zu finden, der immer nach 5 Fragen die Reihenfolge der vier Elemente a_1, a_2, a_3, a_4 herausgefunden hat.

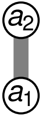
Sortieralgorithmen: Beispiel

Sei $n = 4$. Die informationstheoretische Schranke liefert $\lceil \log_2 24 \rceil = 5$: Theoretisch ist es also möglich, einen Algorithmus zu finden, der immer nach 5 Fragen die Reihenfolge der vier Elemente a_1, a_2, a_3, a_4 herausgefunden hat.

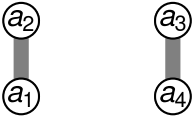
Am Anfang ist es sicherlich egal, welche zwei Elemente wir vergleichen; beginnen wir also mit a_1 und a_2: O.B.d.A. können wir $a_1 < a_2$ annehmen (der andere Fall ist symmetrisch). Wir notieren diese Information in einem sogenannten Hasse–Diagramm, das die bisher bekannten Ordnungsrelationen in einem (von oben nach unten orientierten) Wald zeigt:

Sortieralgorithmen: Beispiel (aus Skriptum)

Ein Element x im Hasse–Diagramm ist größer als ein anderes Element y, wenn x mit y durch einen Weg “von oben nach unten” verbunden ist.

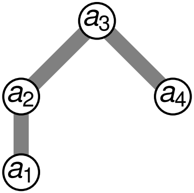


Nun vergleichen wir a_3 und a_4, wieder nehmen wir o.B.d.A. $a_3 > a_4$ an (der andere Fall ist symmetrisch):



Sortieralgorithmen: Beispiel (aus Skriptum)

Nun vergleichen wir a_2 und a_3, sei o.B.d.A. $a_2 < a_3$ (der andere Fall ist symmetrisch):

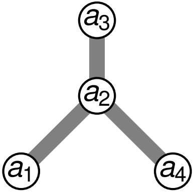


Sortieralgorithmen: Beispiel (aus Skriptum)

Wenn wir jetzt a_2 und a_4 vergleichen, dann gibt es zwei (nicht–symmetrische) Möglichkeiten: Falls $a_2 < a_4$ ist, dann kennen wir bereits die vollständige Ordnung

$$a_1 < a_2 < a_4 < a_3.$$

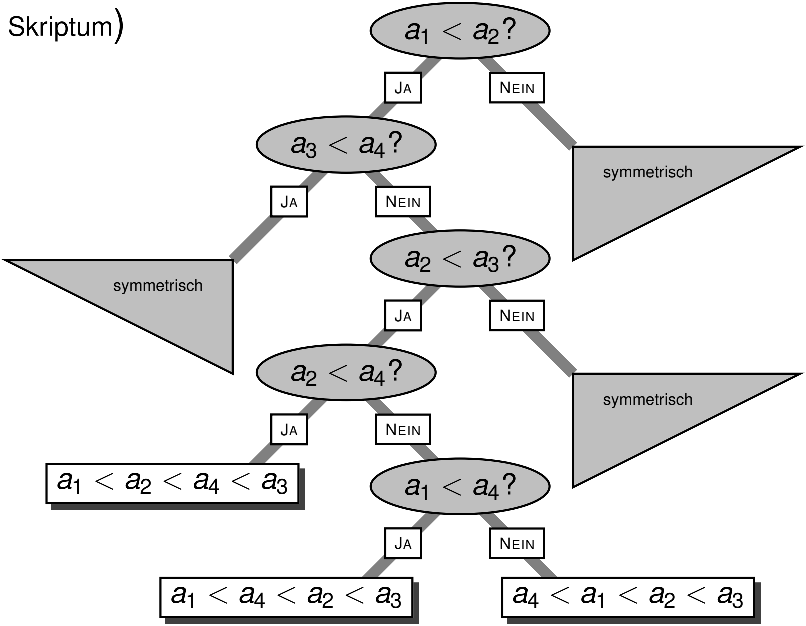
Andernfalls sieht unser Informationsstand im Hasse–Diagramm so aus:



Wenn wir jetzt noch a_1 und a_4 vergleichen, sind wir fertig: Tatsächlich haben wir höchstens 5 Tests gebraucht.

Der entsprechende Entscheidungsbaum sieht so aus:

(aus Skriptum)



Sortieralgorithmen: Drei gängige Algorithmen

1. Sortieren durch Einfügen

2. Mergesort

3. Quicksort