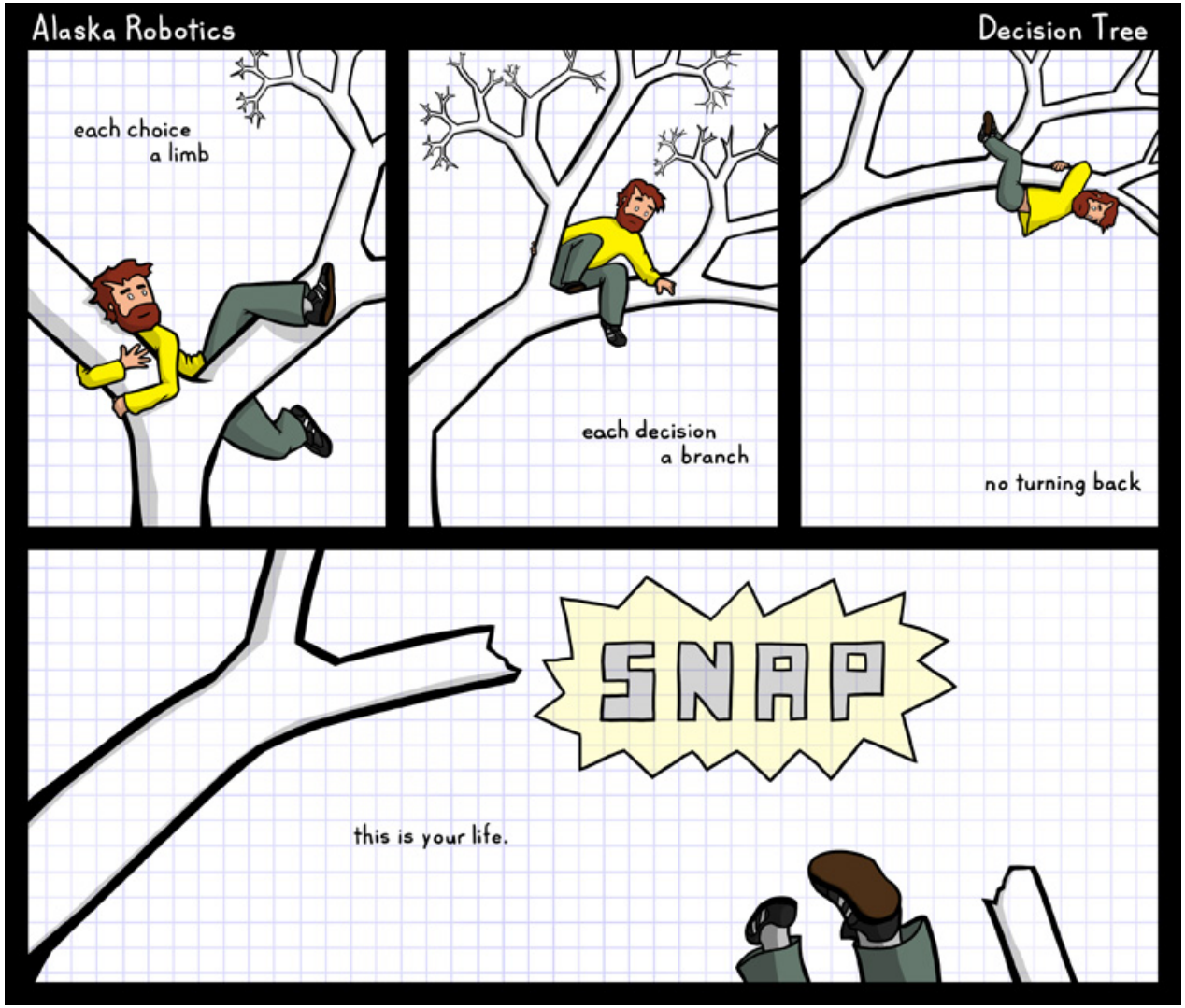
Karlsruher Institut für Technologie

Untere Schranken, Entscheidungsbäume, Yao's Min-Max-Lemma

Proseminar: Die P-ungleich-NP-Vermutung (SS 2012)

Yassine Marrakchi, Yuanfan Wang | 15. Juni 2012

INSTITUT FÜR THEORETISCHE INFORMATIK


Alaska Robotics
Decision Tree
each choice
a limb
each decision
a branch
no turning back
SNAP
this is your life.

Gliederung

Motivation

Abbildung: Franz Grillparzer (1791-1872)

“Let no one say that taking action is hard... the hardest thing in the world is making a decision”

Beipiel

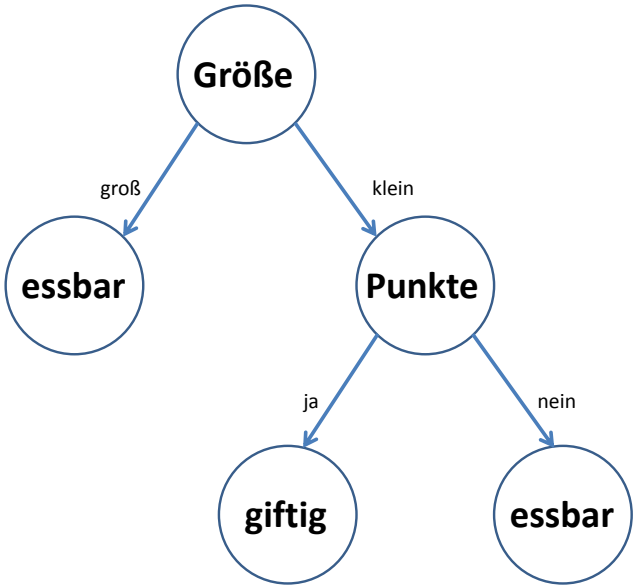
- Es sei die folgende Beispielsammlung für Pilze gegeben.

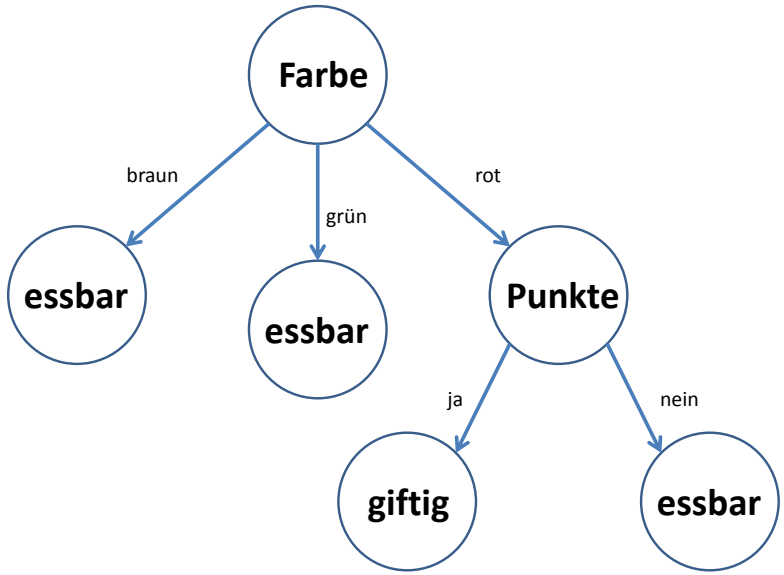
Farbe	Größe	Punkte	Klasse
rot	klein	ja	giftig
braun	klein	nein	essbar
braun	groß	ja	essbar
grün	klein	nein	essbar
rot	groß	nein	essbar

- Ziel: möglichst schnell entscheiden, ob ein Pilz giftig oder essbar ist.

Wie löst man das Problem?

zwei Bäume zum Lösen des Problems





Wie kann man die Qualität eines Baumes beurteilen?

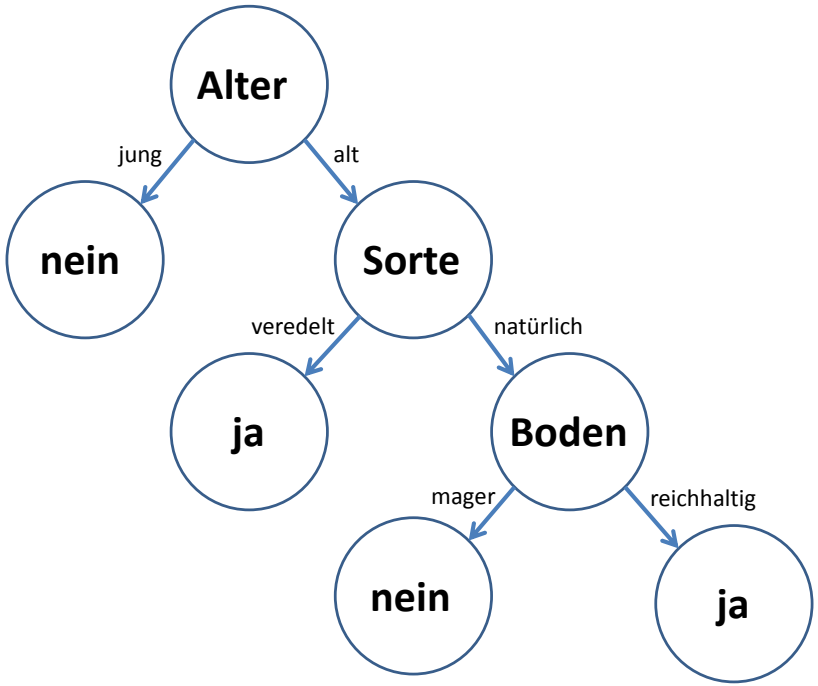
- Die Anzahl der Knoten des Baumes
- Die Tiefe des Baumes
- ...

Wie kann man die Qualität eines Baumes beurteilen?

- Die Anzahl der Knoten des Baumes
- Die Tiefe des Baumes
- ...

Definition von Entscheidungsbäumen

- Entscheidungsbäume sind geordnete, gerichtete Bäume, die der Darstellung von Entscheidungsregeln dienen.

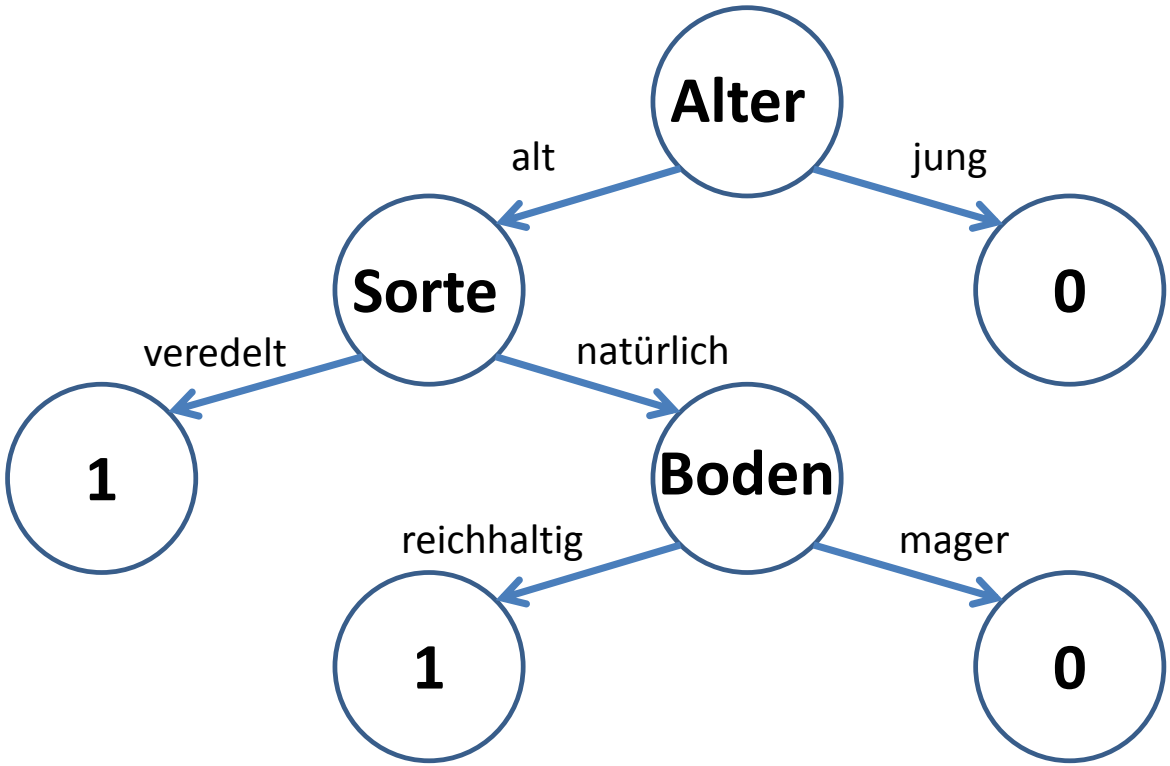


- Einfacher Entscheidungsbaum zur Vorhersage, ob ein Apfelbaum Früchte tragen wird. (Quelle: Wikipedia)

Deterministische Entscheidungsbäume

Definition

- Ein deterministischer Entscheidungsbaum ist ein binärer Baum T.
- Jeder innere Knoten von T ist mit einer Variable x_i markiert und jedes Blatt mit einem Wert von 0 oder 1 bezeichnet.



Berechnung mit deterministischen Entscheidungsbäumen

Gegeben

- Eine Funktion $f : \{0,1\}^n \to \{0,1\}$ und ein deterministischer Entscheidungsbaum T, sodass $\forall x \in \{0,1\}^n$ gilt $f(x) = T(x)$. Der Baum T berechnet die Funktion f.

Gesucht

- Eine Ausgabe der Funktion f

Berechnung mit deterministischen Entscheidungsbäumen

Gegeben

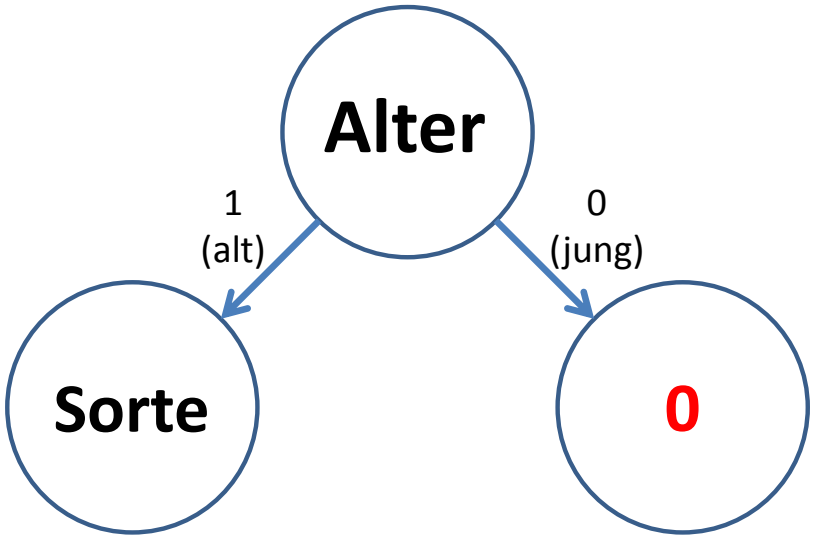
- Eine Funktion $f : \{0,1\}^n \to \{0,1\}$ und ein deterministischer Entscheidungsbaum T, sodass $\forall x \in \{0,1\}^n$ gilt $f(x) = T(x)$. Der Baum T berechnet die Funktion f.

Gesucht

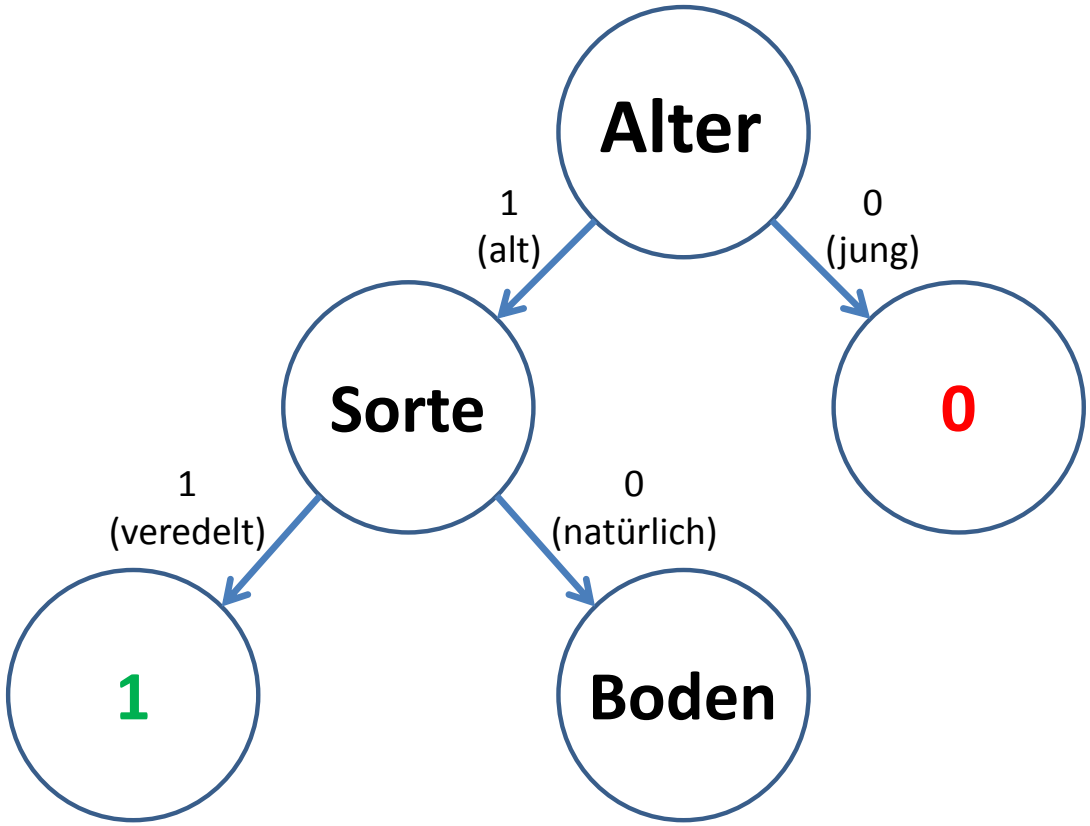
- Eine Ausgabe der Funktion f

Beispiel

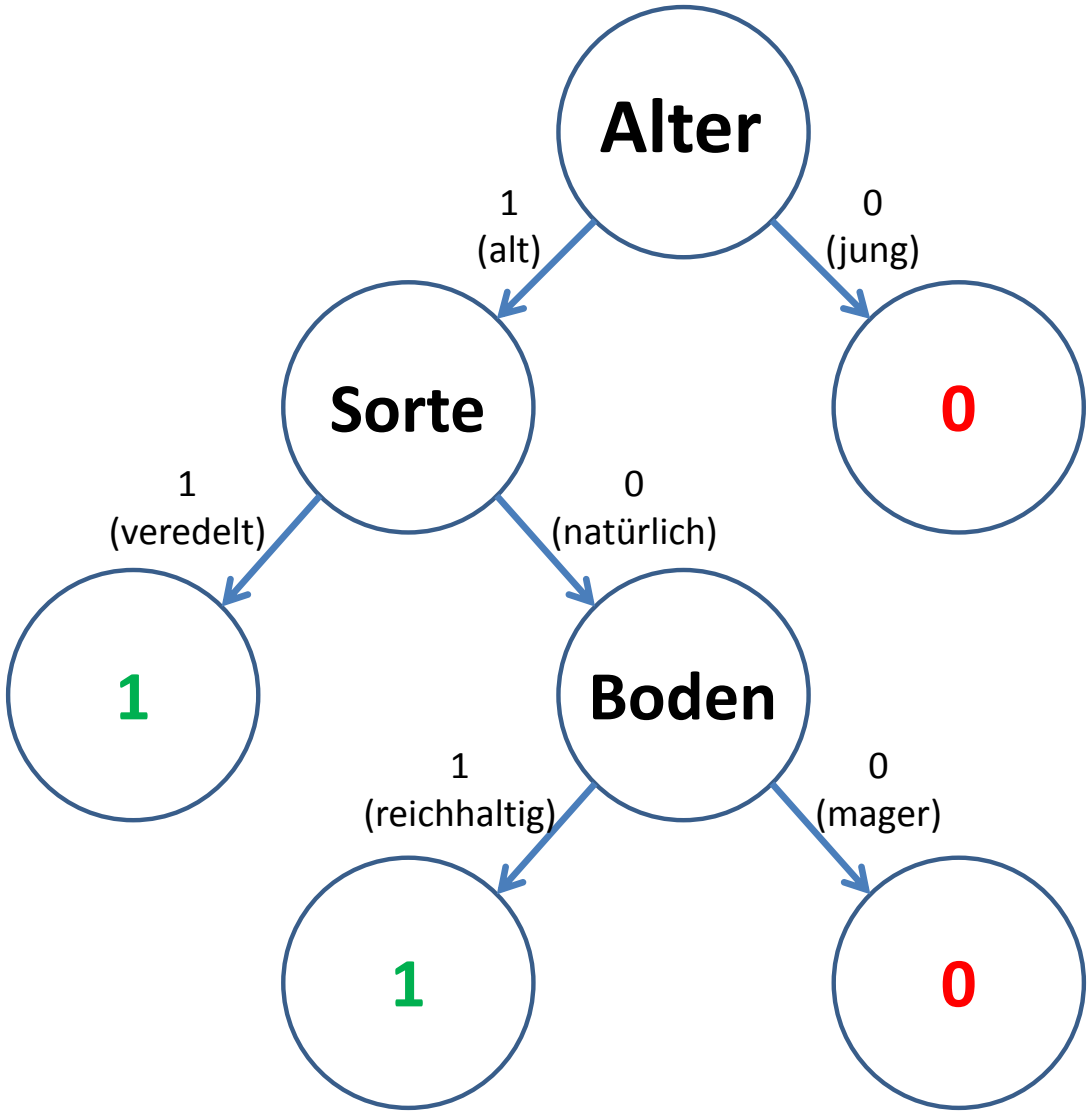
Beispiel



Beispiel



Beispiel


Alter
1
(alt)
0
(jung)
Sorte
0
1
(veredelt)
0
(natürlich)
1
Boden
1
(reichhaltig)
0
(mager)
1
0

Eine erste Metrik

- Kosten $cost(T, x)$ eines Baumes T: maximale Anzahl von Bits, die untersucht werden müssen, um Eingabe x der Länge n zu berechnen.
- Eine Funktion kann anhand von mehreren Entscheidungsbäumen berechnet werden.

$\Longrightarrow$ Keine gute Metrik für die Funktion

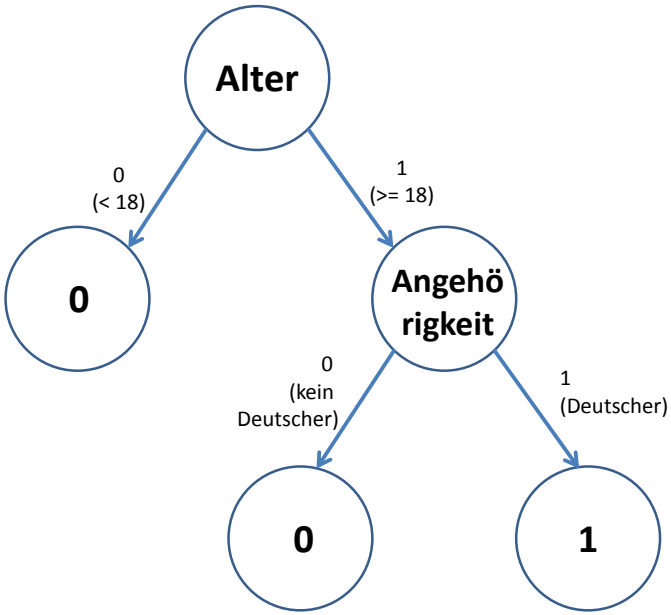
Beispiel

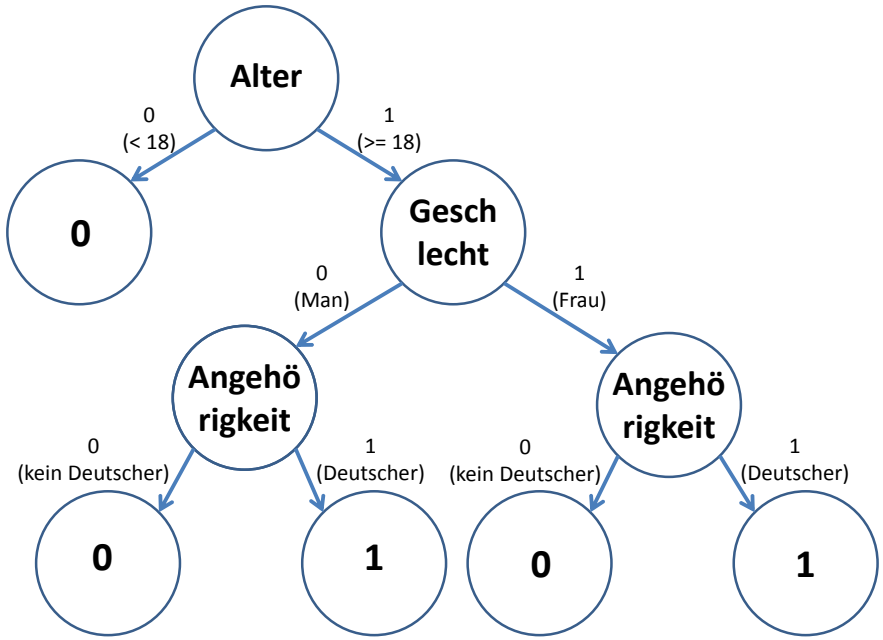
Beispiel: Ein Bürger darf abstimmen genau dann wenn er Deutscher und über 18 ist.

Alter	Geschlecht	Staatsangehörigkeit	Stimmrecht
16	M	Deutsch	Nein
18	M	Französisch	Nein
20	M	Deutsch	Ja
16	F	Französisch	Nein
18	F	Deutsch	Ja
20	F	Französisch	Nein

Beispiel(Fortsetzung)

- Während für den ersten Baum gilt $cost(T_1, x) = 2$, ist $cost(T_2, x) = 3$





Entscheidungsbaumkomplexität

- Sei T(f) die Menge aller Bäume, die eine boolesche Funktion f berechnen. Wir definieren die Entscheidungsbaumkomplexität $D(f)$ als: $D(f) = \min_{t \in T(f)} \max_{x \in \{0,1\}^n} cost(t, x)$.
- Damit gilt für das vorige Beispiel: $D(f) = 2$.
- Offensichtlich gilt $D(f) \leq n$. Eine untere Schranke ist aber interessanter.

Entscheidungsbaumkomplexität

- Sei T(f) die Menge aller Bäume, die eine boolesche Funktion f berechnen. Wir definieren die Entscheidungsbaumkomplexität $D(f)$ als: $D(f) = \min_{t \in T(f)} \max_{x \in \{0,1\}^n} cost(t, x)$.
- Damit gilt für das vorige Beispiel: $D(f) = 2$.
- Offensichtlich gilt $D(f) \leq n$. Eine untere Schranke ist aber interessanter.

Entscheidungsbaumkomplexität

- Sei T(f) die Menge aller Bäume, die eine boolesche Funktion f berechnen. Wir definieren die Entscheidungsbaumkomplexität $D(f)$ als: $D(f) = \min_{t \in T(f)} \max_{x \in \{0,1\}^n} cost(t, x)$.
- Damit gilt für das vorige Beispiel: $D(f) = 2$.
- Offensichtlich gilt $D(f) \leq n$. Eine untere Schranke ist aber interessanter.

Beispiel: OR-Funktion

Gegeben

- Eine Funktion f definiert durch: $f(x_1, x_2, x_3, ..., x_n) = \bigvee_{i=1}^{n} x_i$.

Gesucht

- Eine untere Schranke für $D(f)$.

Behauptung

- Es gibt keinen Baum der alle Eingaben x in weniger als n Schritten berechnet. D.h. $D(f) = n$.

Beispiel: OR-Funktion

Gegeben

- Eine Funktion f definiert durch: $f(x_1, x_2, x_3, ..., x_n) = \bigvee_{i=1}^{n} x_i$.

Gesucht

- Eine untere Schranke für $D(f)$.

Behauptung

- Es gibt keinen Baum der alle Eingaben x in weniger als n Schritten berechnet. D.h. $D(f) = n$.

Beispiel: OR-Funktion

Gegeben

- Eine Funktion f definiert durch: $f(x_1, x_2, x_3, ..., x_n) = \bigvee_{i=1}^{n} x_i$.

Gesucht

- Eine untere Schranke für $D(f)$.

Behauptung

- Es gibt keinen Baum der alle Eingaben x in weniger als n Schritten berechnet. D.h. $D(f) = n$.

Beispiel: OR- Funktion

Behauptung

- Es gibt keinen Baum der alle Eingaben x in weniger als n Schritten berechnet, d.h $D(f) = n$.

Beweis

- Die Funktion genau dann 1 falls ein Eintrag gleich 1 ist.
- Wir benutzen das "Adversary Argument".

$\Longrightarrow$ Der Gegner antwortet immer mit 0.

$\Longrightarrow$ Die Entscheidung ist bis zum vorletzten Schritt nicht sicher.

$\Longrightarrow$ Alle Einträge müssen getestet werden.

$\Longrightarrow$ $D(f) \geq n$.

$\Longrightarrow$ $D(f) = n$. (wegen: $D(f) \leq n$)

Beispiel: OR- Funktion

Behauptung

- Es gibt keinen Baum der alle Eingaben x in weniger als n Schritten berechnet, d.h $D(f) = n$.

Beweis

- Die Funktion genau dann 1 falls ein Eintrag gleich 1 ist.
- Wir benutzen das "Adversary Argument".

$\Longrightarrow$ Der Gegner antwortet immer mit 0.

$\Longrightarrow$ Die Entscheidung ist bis zum vorletzten Schritt nicht sicher.

$\Longrightarrow$ Alle Einträge müssen getestet werden.

$\Longrightarrow$ $D(f) \geq n$.

$\Longrightarrow$ $D(f) = n$. (wegen: $D(f) \leq n$)

Beispiel: OR- Funktion

Behauptung

- Es gibt keinen Baum der alle Eingaben x in weniger als n Schritten berechnet, d.h $D(f) = n$.

Beweis

- Die Funktion genau dann 1 falls ein Eintrag gleich 1 ist.
- Wir benutzen das "Adversary Argument".

$\Longrightarrow$ Der Gegner antwortet immer mit 0.

$\Longrightarrow$ Die Entscheidung ist bis zum vorletzten Schritt nicht sicher.

$\Longrightarrow$ Alle Einträge müssen getestet werden.

$\Longrightarrow$ $D(f) \geq n$.

$\Longrightarrow$ $D(f) = n$. (wegen: $D(f) \leq n$)

Beispiel: OR- Funktion

Behauptung

- Es gibt keinen Baum der alle Eingaben x in weniger als n Schritten berechnet, d.h $D(f) = n$.

Beweis

- Die Funktion genau dann 1 falls ein Eintrag gleich 1 ist.
- Wir benutzen das "Adversary Argument".

$\Longrightarrow$ Der Gegner antwortet immer mit 0.

$\Longrightarrow$ Die Entscheidung ist bis zum vorletzten Schritt nicht sicher.

$\Longrightarrow$ Alle Einträge müssen getestet werden.

$\Longrightarrow$ $D(f) \geq n$.

$\Longrightarrow$ $D(f) = n$. (wegen: $D(f) \leq n$)

Beispiel: OR- Funktion

Behauptung

- Es gibt keinen Baum der alle Eingaben x in weniger als n Schritten berechnet, d.h $D(f) = n$.

Beweis

- Die Funktion genau dann 1 falls ein Eintrag gleich 1 ist.
- Wir benutzen das "Adversary Argument".

$\Longrightarrow$ Der Gegner antwortet immer mit 0.

$\Longrightarrow$ Die Entscheidung ist bis zum vorletzten Schritt nicht sicher.

$\Longrightarrow$ Alle Einträge müssen getestet werden.

$\Longrightarrow$ $D(f) \geq n$.

$\Longrightarrow$ $D(f) = n$. (wegen: $D(f) \leq n$)

Beispiel: OR- Funktion

Behauptung

- Es gibt keinen Baum der alle Eingaben x in weniger als n Schritten berechnet, d.h $D(f) = n$.

Beweis

- Die Funktion genau dann 1 falls ein Eintrag gleich 1 ist.
- Wir benutzen das "Adversary Argument".

$\Longrightarrow$ Der Gegner antwortet immer mit 0.

$\Longrightarrow$ Die Entscheidung ist bis zum vorletzten Schritt nicht sicher.

$\Longrightarrow$ Alle Einträge müssen getestet werden.

$\Longrightarrow$ $D(f) \geq n$.

$\Longrightarrow$ $D(f) = n$. (wegen: $D(f) \leq n$)

Beispiel: Zusammenhang eines Graphen

Gegeben

- Ein ungerichteter Graph mit m Knoten.

Gesucht

- Die minimale Anzahl von Kanten, die getestet werden müssen um den Zusammenhang des Graphen zu überprüfen.

Behauptung

- Das Problem lässt sich in genau $\frac{m \times (m-1)}{2}$ Schritten berechnen.

Beispiel: Zusammenhang eines Graphen

Gegeben

- Ein ungerichteter Graph mit m Knoten.

Gesucht

- Die minimale Anzahl von Kanten, die getestet werden müssen um den Zusammenhang des Graphen zu überprüfen.

Behauptung

- Das Problem lässt sich in genau $\frac{m \times (m-1)}{2}$ Schritten berechnen.

Beispiel: Zusammenhang eines Graphen

Gegeben

- Ein ungerichteter Graph mit m Knoten.

Gesucht

- Die minimale Anzahl von Kanten, die getestet werden müssen um den Zusammenhang des Graphen zu überprüfen.

Behauptung

- Das Problem lässt sich in genau $\frac{m \times (m-1)}{2}$ Schritten berechnen.

Weg zur Lösung

- Nochmal wird das “Adversary Argument” benutzt:
- Betrachte die Adjazenzmatrix:
  - Der Graph ist ungerichtet

$\Longrightarrow$ Die Matrix ist symmetrisch

$\Longrightarrow$ o.B.d.A. ist die obere Dreicksmatrix irrelevant.

$$\mathbf{X} = \begin{pmatrix} * & * & * & * & * & * & * \\ * & * & * & * & * & * & * \\ * & * & * & * & * & * & * \\ * & * & * & * & * & * & * \\ * & * & * & * & * & * & * \\ * & * & * & * & * & * & * \\ * & * & * & * & * & * & * \end{pmatrix}$$

Weg zur Lösung

- Die Schlingen haben keinen Einfluss auf den Zusammenhang eines Graphen

$\Longrightarrow$ Die Hauptdiagonale spielt keine Rolle

$$\mathbf{X} = \begin{pmatrix} * & * & * & * & * & * & * \\ * & * & * & * & * & * & * \\ * & * & * & * & * & * & * \\ * & * & * & * & * & * & * \\ * & * & * & * & * & * & * \\ * & * & * & * & * & * & * \\ * & * & * & * & * & * & * \end{pmatrix}$$

Weg zur Lösung

- Es gilt bis jetzt: $D(f) \leq m \times \frac{m-1}{2}$
- Ungleichung ist scharf: Widerspruchsbeweis.

Nichtdet. Entscheidungsbäume

Definition von Zertifikaten

- Ein 0-Zertifikat für x ist eine Teilmenge $S \subseteq \{0,1\}^n$ sodass $f(x') = 0$ für alle x' für die $x' \mid_S = x \mid_S$. Analog definieren wir auch ein 1-Zertifikat.
- Die Zertifikatskomplexität $C(f)$ ist das minimale k, so dass jede Zeichenkette x ein f(x)-Zertifikat von maximaler Länge k hat.

110- - -01

Nichtdet. Entscheidungsbäume

Definition von Zertifikaten

- Ein 0-Zertifikat für x ist eine Teilmenge $S \subseteq \{0,1\}^n$ sodass $f(x') = 0$ für alle x' für die $x' \mid_S = x \mid_S$. Analog definieren wir auch ein 1-Zertifikat.
- Die Zertifikatskomplexität $C(f)$ ist das minimale k, so dass jede Zeichenkette x ein f(x)-Zertifikat von maximaler Länge k hat.

110- - -01

Nichtdet. Entscheidungsbäume

Definition von Zertifikaten

- Ein 0-Zertifikat für x ist eine Teilmenge $S \subseteq \{0,1\}^n$ sodass $f(x') = 0$ für alle x' für die $x' \mid_S = x \mid_S$. Analog definieren wir auch ein 1-Zertifikat.
- Die Zertifikatskomplexität $C(f)$ ist das minimale k, so dass jede Zeichenkette x ein f(x)-Zertifikat von maximaler Länge k hat.

110- - -01

Nichtdet. Entscheidungsbäume (Fortsetzung)

Bemerkung

- Eine Zeichenkette x kann nicht gleichzeitig ein 0-Zertifikat und ein 1-Zertifikat haben.
- Offensichtlich ist $f(x)$ entweder 0 oder 1

Folgerung

- Jedes 0- und 1-Zertifikat müssen sich mindestens an einer Stelle überlappen.

Begründung

- Gäbe es zwei Zertifikate, die disjunkt sind, so könnte man eine Zeichenkette basteln für die die beide Zertifikate erfüllt sind (Widerspruch).

Nichtdet. Entscheidungsbäume (Fortsetzung)

Bemerkung

- Eine Zeichenkette x kann nicht gleichzeitig ein 0-Zertifikat und ein 1-Zertifikat haben.
- Offensichtlich ist $f(x)$ entweder 0 oder 1

Folgerung

- Jedes 0- und 1-Zertifikat müssen sich mindestens an einer Stelle überlappen.

Begründung

- Gäbe es zwei Zertifikate, die disjunkt sind, so könnte man eine Zeichenkette basteln für die die beide Zertifikate erfüllt sind (Widerspruch).

Nichtdet. Entscheidungsbäume (Fortsetzung)

Bemerkung

- Eine Zeichenkette x kann nicht gleichzeitig ein 0-Zertifikat und ein 1-Zertifikat haben.
- Offensichtlich ist $f(x)$ entweder 0 oder 1

Folgerung

- Jedes 0- und 1-Zertifikat müssen sich mindestens an einer Stelle überlappen.

Begründung

- Gäbe es zwei Zertifikate, die disjunkt sind, so könnte man eine Zeichenkette basteln für die die beide Zertifikate erfüllt sind (Widerspruch).

Beispiel: Zusammenhang eines Graphen

1-Zertifikat von G

- Eine Menge von Kanten, deren Existenz den Zusammenhang des Graphen G impliziert.
- Ein Graph ist zusammenhängend, falls er einen Spannbaum enthält. Ein Spannbaum hat immer die Größe $n - 1$.

$\Longrightarrow C_1(f) = n - 1$

Beispiel: Zusammenhang eines Graphen

0-Zertifikat von G

- Eine Menge von Kanten, die falls nicht vorhanden einen nicht zusammenhangenden Graphen erzeugen.

- Der Graph ist nicht zusammenhängend $\Longrightarrow$ Es gibt 2 Zusammenhangskomponenten.
- Gesucht ist der maximale Schnitt zwischen diese Zusammenhangskomponenten:
  - Seien p und q die Kardinalitäten der beiden Komponenten:
  - Maximiere $p * q$ mit $p + q = n$
  - Beweis:
    - $p * q$ ist maximal wenn:
      1) $p = q = \frac{n}{2}$ für n gerade
      2) $p = q - 1 = \frac{n-1}{2}$ für n ungerade

    $\Longrightarrow$ $C_0(f) = (\frac{n}{2})^2$ für n gerade und $C_0(f) = \frac{n^2-1}{4}$ für n ungerade.

Zusammenhang zwischen D(f) und C(f)

- $C(f) \leq D(f)$
  - Beweis: Es ist klar, dass jede Zeichenkette mit ausreichender Länge um $f(x)$ zu berechnen ein f(x)-Zertifikat von f liefert.
- $D(f) \leq C(f)^2$.
  - Hier ohne Beweis.

Randomisierte Entscheidungsbäume

Definition

- Für jede Funktion f sei P die Menge der Wahrscheinlichkeitsverteilungen über Entscheidungsbäume, die f zu berechnen. Die randomisierte Baumkomplexität von f ist definiert als
- $R(f) = \underset{P \in P_f}{Min} \; \underset{x \in \{0,1\}^n}{Max} \; \underset{t \in_R P}{E} [cost(t, x)]$

Eigenschaften von randomisierten Entscheidungsbäumen

- $C(f)$: Komplexität von Zertifikaten
- $D(f)$: Komplexität von deterministischen Entscheidungsbäumen
- $R(f)$: Komplexität von randomisierten Entscheidungsbäumen

Eigenschaft

- Behauptung 1: Es gilt $R(f) \leq D(f)$
  Diese Beziehung ist offensichtlich weil ein deterministischer Baum ein Sonderfall einer Verteilung ist.
- Behauptung 2: Es gilt $C(f) \leq R(f)$
  Begründung: jeder Entscheidungsbaum enthält ein f(x)-Zertifikat für jede Eingabe x.

Eigenschaften von randomisierten Entscheidungsbäumen

- $C(f)$: Komplexität von Zertifikaten
- $D(f)$: Komplexität von deterministischen Entscheidungsbäumen
- $R(f)$: Komplexität von randomisierten Entscheidungsbäumen

Eigenschaft

- Behauptung 1: Es gilt $R(f) \leq D(f)$
  Diese Beziehung ist offensichtlich weil ein deterministischer Baum ein Sonderfall einer Verteilung ist.
- Behauptung 2: Es gilt $C(f) \leq R(f)$
  Begründung: jeder Entscheidungsbaum enthält ein f(x)-Zertifikat für jede Eingabe x.

Beispiel für randomisierte Entscheidungsbäume

- Wir betrachten die MAJ3 Funktion.
- Behauptung: $R(f) \leq \frac{8}{3}$
- Beweis:
  - Sei t ein randomisierter Entscheidungsbaum der eine zufällige Permutation von x_1, x_2 und x_3 auswählt.
  - Bei 000 und 111 gibt es kein Problem. Auf jeden Fall und unabhängig von der Reihenfolge ist die Entscheidung direkt nach dem 2. Test festgelegt
  - Bei den anderen 6 Möglichkeiten kann die Entscheidung nach 2 Tests getroffen werden mit Wahrscheinlichkeit von $\frac{1}{3}$. Ansonsten ist ein 3. Test notwendig.
  - Insgesamt ist der erwartete Wert von $R(f) \leq 2 \times (\frac{1}{3}) + 3 \times (\frac{2}{3}) = \frac{8}{3}$.

Motivation

- Bis jetzt haben wir nur mit dem Adversary Argument die untere Schranke von Funktionen berechnet.
- Diese Methode ist aber:
  - bei vielen Problemen unpraktisch und viel zu kompliziert
  - für nichtdeterministische und randomisierte Fälle sehr beschränkt.

$\rightarrow$ Alternative?

Wofür Yao's Min-Max Lemma

Problem

- Bei randomisierten Entscheidungsbäumen ist es schwieriger, die untere Schranke zu bestimmen, als bei deterministischen Entscheidungsbäumen.

Lösung

- Zum Glück hat Yao bewiesen, dass ein äquivalentes Problem existiert, das mittels deterministischer Bäume lösbar ist.

Wofür Yao's Min-Max Lemma

Problem

- Bei randomisierten Entscheidungsbäumen ist es schwieriger, die untere Schranke zu bestimmen, als bei deterministischen Entscheidungsbäumen.

Lösung

- Zum Glück hat Yao bewiesen, dass ein äquivalentes Problem existiert, das mittels deterministischer Bäume lösbar ist.

Yao's Min-Max Lemma

- Sei X eine endliche Menge von Eingaben und A die Menge aller Algorithmen, die zum Lösen von f dienen. Die randomisierte Komplexität des Problems lautet:
  - $\min_{R} \max_{x \in X} cost(R, x)$ $(*)$
- Sei D die Verteilung von Eingaben. Für jeden deterministischen Algorithmus $a \in A$ sind die Kosten $cost(a, D) = E_{x \in D}[cost(a, x)]$. Die Komplexität des Problems lautet:
  - $\max_{D} \min_{a \in A} cost(a, D)$ $(**)$
- Yao's Lemma besagt, dass die beiden Entitäten $(*)$ und $(**)$ gleich sind.

Interpretation

- Anstatt die Verteilung der Bäume für spezifische Eingaben zu betrachten, kümmert man sich um die Verteilung von Eingaben für bestimmte Bäume.
- Dafür geht man wie folgt vor:
  1. Wähle eine günstige Verteilung der Eingaben
  2. Bestimme $C = cost(a, D) = \underset{x \in D}{E}[cost(a, x)]$
  3. Mit Yao's Lemma kann man daraus folgern dass $R(f) \geq C$

Interpretation

- Anstatt die Verteilung der Bäume für spezifische Eingaben zu betrachten, kümmert man sich um die Verteilung von Eingaben für bestimmte Bäume.
- Dafür geht man wie folgt vor:
  1. Wähle eine günstige Verteilung der Eingaben
  2. Bestimme $C = cost(a, D) = \underset{x \in D}{E}[cost(a, x)]$
  3. Mit Yao's Lemma kann man daraus folgern dass $R(f) \geq C$

Beispiel : MAJ3 Funktion

- Gesucht ist eine untere Schranke von $R(f)$
- Weg zur Lösung:
  1. Setze die folgende Verteilung: 000 und 111 tauchen mit der Wahrscheinlichkeit 0 auf. Für die anderen Eingaben gilt die Gleichverteilung.
  2. Die Entscheidung kann nach 2 Operationen mit Wahrscheinlichkeit $\frac{1}{3}$ getroffen werden. Die Kosten sind 3 in $\frac{2}{3}$ der Fälle. Damit beträgt der Erwartungswert genau $\frac{8}{3}$.
  3. Es gilt $R(f) \geq \frac{8}{3}$

$\longrightarrow$ Wir haben aber schon gezeigt dass R (f) $\leq \frac{8}{3}$. Es gilt dann: R (f) = $\frac{8}{3}$.

Beispiel : MAJ3 Funktion

- Gesucht ist eine untere Schranke von $R(f)$
- Weg zur Lösung:
  1. Setze die folgende Verteilung: 000 und 111 tauchen mit der Wahrscheinlichkeit 0 auf. Für die anderen Eingaben gilt die Gleichverteilung.
  2. Die Entscheidung kann nach 2 Operationen mit Wahrscheinlichkeit $\frac{1}{3}$ getroffen werden. Die Kosten sind 3 in $\frac{2}{3}$ der Fälle. Damit beträgt der Erwartungswert genau $\frac{8}{3}$.
  3. Es gilt $R(f) \geq \frac{8}{3}$

$\longrightarrow$ Wir haben aber schon gezeigt dass R (f) $\leq \frac{8}{3}$. Es gilt dann: R (f) = $\frac{8}{3}$.

Beispiel : MAJ3 Funktion

- Gesucht ist eine untere Schranke von $R(f)$
- Weg zur Lösung:
  1. Setze die folgende Verteilung: 000 und 111 tauchen mit der Wahrscheinlichkeit 0 auf. Für die anderen Eingaben gilt die Gleichverteilung.
  2. Die Entscheidung kann nach 2 Operationen mit Wahrscheinlichkeit $\frac{1}{3}$ getroffen werden. Die Kosten sind 3 in $\frac{2}{3}$ der Fälle. Damit beträgt der Erwartungswert genau $\frac{8}{3}$.
  3. Es gilt $R(f) \geq \frac{8}{3}$

$\longrightarrow$ Wir haben aber schon gezeigt dass R (f) $\leq \frac{8}{3}$. Es gilt dann: R (f) = $\frac{8}{3}$.

Sensitivitätsanalyse

- Wofür die Sensitivitätsanalyse?
  - eine Analyseform für komplexe Systeme
  - Wirkbeziehungen werden zwischen Systemvariablen zu einem Wirkungsnetz verbunden. Die kleine Änderungen beeinflussen das gesamte System.
  - Diese Methode bietet die Möglichkeit, eine untere Schranke für die Entscheidungsbaumkomplexität abzuschätzen.

Definition von Sensitivität

- Sei $f : \{0,1\}^n \to \{0,1\}$ eine Funktion und $x \in \{0,1\}^n$
- Sensivität von f bezüglich x ist die Anzahl von Stellen für die gilt:
  - f(x) $\neq f(x^i)$ wobei x^i genau x bis auf die i-te Stelle ist.
  - Wir bezeichnen diese Sensitivität mit $s_x(f)$.
- Die Sensitivität von f, bezeichnet mit $s(f)$, ist $\max_x \{s_x(f)\}$.

Beispiel: AND-OR funktion

- Gegeben sei eine positive Zahl k und ein String x der Länge k^2 sodass $x = x_1 x_2 \ldots x_k$. Die Blöcke x_i haben die Länge k.
  - Setze $c(i)$ = OR x_{ij} mit $0 \leq j < k$
  - $f(x)$ = AND c_i mit $0 < i \leq k$
- Die Sensitivität von f ist k.

Verallgemeinerung

- Ähnlich wie bei der Sensitivität definieren wir noch eine Metrik : die Block Sensitivität.
- Wir bezeichnen mit $bs_x(f)$ die Block Sensitivität definiert durch: b die maximale Anzahl von disjunkten Blöcken B_1, B_2,...,B_b für die gilt f(x) ungleich $f(x^{B_i})$.

Satz

- Für jede Funktion gilt: $s(f) \leq bs(f) \leq C(f)$
- Beweis:
  1. Die Sensitivität ist ein Sonderfall von Blocksensitivität
  2. Wähle x sodass $bs_x(f) = bs(f)$.
     Damit man zwischen x und x^{B_i} unterscheiden kann, muss man mindestens eine Stelle von jedem B_i betrachten und damit ist $bs(f) \leq C(f)$.
- Für jede Funktion gilt $C(f) \leq s(f) * bs(f)$
  (ohne Beweis)

Satz

- Für jede Funktion gilt: $s(f) \leq bs(f) \leq C(f)$
- Beweis:
  1. Die Sensitivität ist ein Sonderfall von Blocksensitivität
  2. Wähle x sodass $bs_x(f) = bs(f)$.
     Damit man zwischen x und x^{B_i} unterscheiden kann, muss man mindestens eine Stelle von jedem B_i betrachten und damit ist $bs(f) \leq C(f)$.
- Für jede Funktion gilt $C(f) \leq s(f) * bs(f)$
  (ohne Beweis)

Definition der Degree Methode

- Wir definieren ein multilineares Polynom als ein Polynom mit n Variablen dessen Grad maximal 1 ist.
- Der Grad von f ist der Grad des Polynoms, das f repräsentiert.
- Ein Polynom P repräsentiert eine Funktion falls für alle $x \in \{0,1\}^n$ gilt $P(x) = f(x)$.

Eindeutigkeit des Polynoms

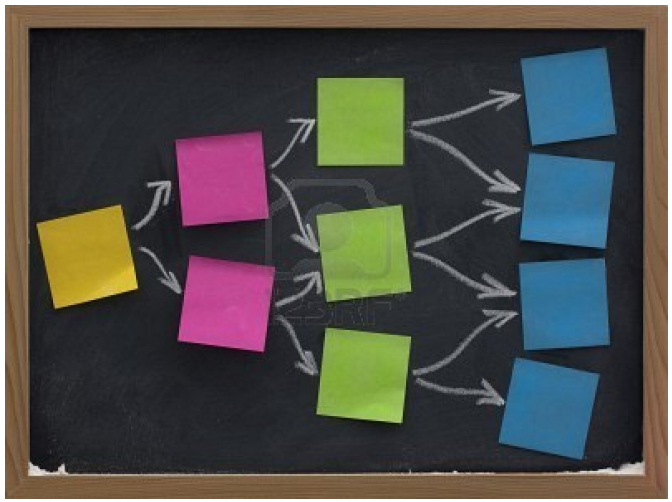
- Behauptung: Das multilineare Polynom von F ist eindeutig.
- Annahme: Es gibt zwei verschiedene Polynome p und q sodass $p(x) = q(x) = f(x)$ für alle Eingaben x der Länge n. Setze $r(x) = p(x) - q(x)$.
- Seien A das kleinste nicht 0 Monome von r und c das Koeffizient von A (c ist nicht 0).
- Definiere y sodass $y_i = 1$ genau dann wenn die i. Variable in A auftaucht, an sonst 0.
- $r(y) = c$ (Widerspruch) weil $p(y) - q(y)$ überall gleich 0 sein soll.

Eigenschaften

- Satz: $deg(f) \leq D(f)$
- Beweis:
  - Sei T ein Entscheidugnsbaum von f mit Komplexität $D(f)$. Ist die Ausgabe immer gleich 0 dann $deg(f) = 0 \leq D(f)$ ist trivial.
  - Ansonst seien ein 1-Blatt, x_1,x_2,...,x_r die Menge aller Stellen die betrachtet werden sollen und b_1,b_2,...,b_r die dazu gehörigen Werte.
  - Definiere ein Polynom PL wie folgt:
    - $PL(x) = \prod_{i:b_i=1} X_i \prod_{i:b_i=0}(1 - X_i)$
    - PL ist vom Grad $r \leq D(f)$. Für alle x gilt: $PL(x) = 1$ falls L mit der Eingabe x erreicht wird, an sonst $PL(x) = 0$.
    - Es reicht jetzt das Polynom P zu definieren als die Summe aller PL. Daraus folgt die Behauptung.

Was haben wir schon gelernt?

- Was ist ein Entscheidungsbaum
- Typen von Entscheidungsbäumen
- Techniken zur Berechnung von unteren Schranken
- Vor- und Nachteile von Entscheidungsbäumen



Zusamenfassung

- Entscheidungsbaumkomplexität einer Funktion f ist die Anzahl von Bits, die betrachtet werden müssen, um den Wert von f zu bestimmen. Davon gibt es auch nichtdeterministische und randomisierte Versionen.
- Min-Max Lemma von Yao reduziert Aufgabe bei probabilistische Algorithmen auf Berechnung von average-case bounds für deterministische Algorithmen.
- Andere Ansätze:
  - Adversary Argument
  - Komplexität von Zertifkaten
  - Sensitivität, Blocksensitivität
  - Degree Methode

Zusamenfassung

- Entscheidungsbaumkomplexität einer Funktion f ist die Anzahl von Bits, die betrachtet werden müssen, um den Wert von f zu bestimmen. Davon gibt es auch nichtdeterministische und randomisierte Versionen.
- Min-Max Lemma von Yao reduziert Aufgabe bei probabilistische Algorithmen auf Berechnung von average-case bounds für deterministische Algorithmen.
- Andere Ansätze:
  - Adversary Argument
  - Komplexität von Zertifkaten
  - Sensitivität, Blocksensitivität
  - Degree Methode

Zusamenfassung

- Entscheidungsbaumkomplexität einer Funktion f ist die Anzahl von Bits, die betrachtet werden müssen, um den Wert von f zu bestimmen. Davon gibt es auch nichtdeterministische und randomisierte Versionen.
- Min-Max Lemma von Yao reduziert Aufgabe bei probabilistische Algorithmen auf Berechnung von average-case bounds für deterministische Algorithmen.
- Andere Ansätze:
  - Adversary Argument
  - Komplexität von Zertifkaten
  - Sensitivität, Blocksensitivität
  - Degree Methode

Formelsammlung:

1. $C(f)$: Komplexität von Zertifikaten
2. $D(f)$: Komplexität von deterministischen Entscheidungsbäumen
3. $R(f)$: Komplexität von randominisierten Entscheidungsbäumen
4. $s(f)$: Sensitivität von f
5. $bs(f)$: Block Sensitivität von f
6. $deg(f)$: Degree von multilinearen Polynomen für f

Es gilt:

- $C(f) \leq R(f) \leq D(f) \leq C(f)^2$
- $s(f) \leq bs(f) \leq D(f) \leq bs(f)^3$
- $C(f) \leq s(f)bs(f)$
- $bs(f) \leq 2deg(f)$
- $D(f) \leq deg(f)^2 bs(f) \leq 2deg(f)^4$

Formelsammlung:

1. $C(f)$: Komplexität von Zertifikaten
2. $D(f)$: Komplexität von deterministischen Entscheidungsbäumen
3. $R(f)$: Komplexität von randominisierten Entscheidungsbäumen
4. $s(f)$: Sensitivität von f
5. $bs(f)$: Block Sensitivität von f
6. $deg(f)$: Degree von multilinearen Polynomen für f

Es gilt:

- $C(f) \leq R(f) \leq D(f) \leq C(f)^2$
- $s(f) \leq bs(f) \leq D(f) \leq bs(f)^3$
- $C(f) \leq s(f)bs(f)$
- $bs(f) \leq 2deg(f)$
- $D(f) \leq deg(f)^2 bs(f) \leq 2deg(f)^4$

Gibt es Fragen?

VIELEN DANK FÜR DIE AUFMERKSAMKEIT!

Quellen:

- Computational Complexity: A Modern Approach
- Wikipedia: Entscheidungsbaum
- Alaska Robotics