Samenvatting Probleem Oplossend Denken II

TIN 2 - HoGent

Lorenz Verschingel

18 april 2015

Kansrekening

1 Gebeurtenissen en hun kansen

1.1 Inleiding

De kansrekening houdt zich bezig met de studie van gebeurtenissen of toevalsveranderlijken.

1.2 Universum of uitkomstenruimte

Het universum of de utkomstenruimte van een experiment is de verzameling van alle mogelijke uitkomsten van dit experiment en wordt genoteerd met Ω.

Het is van belang dat de uitkomstenruimte volledig is: elke mogelijke uitkomst van een experiment moet tot Ω behoren. Bovendien moet elke uitkomst van een experiment overeenkomen met juist één element van Ω.

1.3 Gebeurtenis

Een gebeurtenis is een deelverzameling van de uitkomstenruimte. Een enkelvoudige of elementaire gebeurtenis is een *singleton*.

Een *samengestelde* gebeurtenis heeft cardinaliteit groter dan 1.

Gebeurtenissen die geen gemeenschappelijke uitkomsten hebben noemt men *disjunct*. Disjuncte gebeurtenissen kunnen dus nooit samen voorkomen.

1.4 Kansen en kansruimte

We wensen nu aan elke gebeurtenis A een getal te koppelen dat uitdrukt hoe waarschijnlijk het is dat deze gebeurtenis voorkomt bij het uitvoeren van een experiment. We noemen dit getal de *kans* of *waarschijnlijkheid* van A, en we noteren deze kans als $P(A)$.

Het toekennen van kansen aan gebeurtenissen dient aan de volgende drie regels te voldoen:

1. Kansen zijn steeds positief: voor elke gebeurtenis A geldt dat $P(A) \geq 0$.
2. De uitkomstenruimte heeft kans 1: $P(\Omega) = 1$.
3. Wanneer A en B disjuncte gebeurtenissen zij dan is: $P(A \cup B) = P(A) + P(B)$.

   Dit noemt men de **somregel**.

Wanneer de functie P aan de bovenstaande eigenschappen voldoet dan noemt ment het drietal $(\Omega, P(\Omega), P)$ een *kansruimte*.

Kansen voldoen aan de volgende eigenschappen:

1. Voor elke gebeurtenis A geldt dat $P(\overline{A}) = 1 - P(A)$

   $1 = P(\Omega) = P(A \cup \overline{A}) = P(A) + P(\overline{A})$
2. De onmogelijke gebeurtenis heeft een kans nul: $P(\emptyset) = 0$

   $P(\emptyset) = P(\overline{\Omega}) = 1 - P(\Omega) = 0$
3. Als $A \subseteq B$ dan is $P(A) \leq P(B)$, dan geldt$= P(A) = P(B) - P(B \setminus A)$.
4. De **uitgebreide somregel** is: $P(A \cup B) = P(A) + P(B) - P(A \cap B)$

   $P(A \cup B) = P(A \cup (B \setminus A))$
   $= P(A) + P(B \setminus A)$
   $= P(A) + P(B \setminus (A \cap B))$
   $= P(A) + -P(B) - P(A \cap B)$

1.4.1 Eindig universum

Formule van Laplace: $P(A) = \frac{\#(A)}{\#(\Omega)}$
De formule van Laplace is enkel van toepassing als alle kansen even waarschijnlijk zijn.

1.5 Voorwaardelijke kansen en (on)afhankelijkheid van gebeurtenissen

Als $P(B) > 0$, dab is de **voorwaardelijke kans** dat A voorkomt als gegeven is dat B voorkomt gedefinieerd is als: $P(A|B) = \frac{P(A \cap B)}{P(B)}$

$P(A|B)$ spreken we uit als "De kans op A gegeven B".

Twee gebeurtenissen A en B zijn **onafhankelijk** als: $P(A \cap B) = P(A)P(B)$

Bewijs: $P(A|B) = \frac{P(A \cap B)}{P(B)} = P(A) \Leftrightarrow P(A \cap B) = P(A)P(B)$

1.5.1 De regel van Bayes

$P(B_j|A) = \frac{P(B_j)P(P(A|B_j)}{\sum_{i=1}^{n} P(B_i)P(A|B_i)}$

Dit resultaat is belangrijk voor het opstellen van zogenaamde boomdiagrammen of beslissingsbomen in het kader van opeenvolgende beslissingen.

2 Kans- of toevalsvariabelen

2.1 Inleiding

Een *kansvariabele* X is een afbeelding van Ω naar R. Deze afbeelding associeert met elke mogelijke uitkomst van een kansexperiment dus een reëel getal.

2.2 Discrete kansvariabelen

Een kansvariabele X is *discreet* wanneer X slechts een eindig of aftelbaar oneindig aantal waarden aanneemt. Dit wil zeggen dat: $bld(X) = \{x_1, x_2, \dots\}$.

$P(X = x_i) = P(\{\omega \cap \Omega | X(\omega) = x_i\}) = f_X(x_i)$

2.3 Continue kansvariabelen

Een toevalsveranderlijke X is *continu* als er een functie f_X van R naar R^+ bestaat zodanig dat: $F_X(x) = \int_{-\infty}^{x} f_X(y)dy$

De functie f_X wordt de *kansdichtheid* genoemd.

2.4 Verwachtingswaarde en variantie

2.4.1 Discrete kansvariabele

$E[x] = \sum_i^n x_i P(X = x_i)$

$E[x] = \mu$ = expected value = verwachtingswaarde = gemiddelde

$Var[x] = \sum_i^n (x_i - E[x])^2 P(X = x_i)$

$Var[x] = \sigma^2$ = variantie = gemiddelde kwadratische afwijking

σ = standaardafwijking

2.4.2 Continue kansvariabele

$E[x] = \int_{-\infty}^{\infty} x f_X(x) dx$

$Var[x] = \int_{-\infty}^{\infty} (x - \sigma_x)^2 f_X(x) dx$

2.4.3 Eigenschappen van verwachtingswaarde en variantie

1. Als X constant is, i.e. $X(\omega) = k$ voor alle elementen ω van Ω, dan is $E(X) = k$.
2. Als $a \in R$ een constante is, dan geldt: $E(X + a) = E(X) + a$

   waaruit volgt dat: $E(X - \mu_X) = 0$
3. Als $a \in R$ een constante is, dan geldt: $E(aX) = aE(X)$
4. Er geldt steeds dat: $\sigma_X^2 = E(X^2) - \mu_X^2$

   Deze formule geeft de mogelijkheid om de variantie efficiënter te berekenen dan rechtstreeks via de definitie.
5. $\sigma_{X+a}^2 = \sigma_X^2$
6. $\sigma_{aX}^2 = a^2 \sigma_X^2$

3 Kansverdelingen

3.1 Discrete kansverdelingen

3.1.1 De Bernoulliverdeling

x kan maar twee waarden aannemen.

x	0	1
$f_X(x)$	1-p	p

$\mu_X = 0 \times (1 - p) + 1 \times p = p$

$\sigma_X^2 = E[x^2] - E[x]^2 = 0^2 \times (1 - p) + 1^2 \times p - p^2 = p(1 - p) = pq$

Het Bernouilliverdeling kan het best beschreven worden met het volgende voorbeeld: één maal een muntstuk opgooien en kijken of je munt hebt.

3.1.2 De binomiale verdeling

$X \approx B(n, p)$

De binimiale verdeling kan het best beschreven worden met het volgende voorbeeld: tien maal met een munt gooien en kijken hoeveel keer je munt hebt.

$f_X(k) = P(X = k) = C_n^k p^k q^{n-k} = \binom{n}{k} p^k q^{n-k}$

met $\binom{n}{k} = \frac{n!}{k!(n-k)!}$

$\mu_X = np$

$\sigma_X^2 = np(1-p)$

3.1.3 De geometrische verdeling

De geometrische verdeling kan het best beschreven worden met het volgende voorbeeld: blijven gooien met een muntstuk tot je munt hebt.

$P(X = k) = (1-p)^{k-1} p$

$\mu_X = \frac{1}{1-q}$

$\sigma_X^2 = \frac{q}{p^2}$

De markov-eigenschap

$$\begin{aligned} P(X > m+n | X > m) &= \frac{P((X>m+n) \cap (X>n))}{P(X>n)} \\ &= \frac{P(X>m+n)}{P(X>m)} \\ &= \frac{q^{m+n}}{q^m} \\ &= q^n \\ &= P(X > n) \end{aligned} \tag{1}$$

3.1.4 De Poisson verdeling

De Poisson verdeling kan gezien worden als een binomiale verdeling met n zeer groot en p $\approx$ 0.

$\Rightarrow np = \lambda$

$P(X = k) = e^{-\lambda} \frac{\lambda^k}{k!}$

$\mu_X = np = \lambda$

$\sigma_X^2 = (np(1-p)) = \lambda \times 1 = \lambda$

Het Poisson-proces

$P(\text{k voorkomens in [0,t]}) = e^{-\lambda t} \frac{(\lambda t)^k}{k!}$

3.2 Continue kansverdelingen

3.2.1 Uniforme kansverdeling

$\mu_X = \frac{a+b}{2}$

$\sigma_X^2 = \frac{(b-a)^2}{12}$

3.2.2 De exponentiële verdeling

X $\rightarrow$ Poisson-proces met parameter λ.

T = tijd tot eerste voorkomen.

$P(T > t) = P($0 voorkomens in[0,t]$) = e^{-\lambda t}\frac{(\lambda t)^k}{k!} = e^{-\lambda t}$

$\Rightarrow P(T \leq t) = 1 - e^{-\lambda t}$ als t ≥ 0

$\mu_T = \frac{1}{\lambda}$

$\sigma_T^2 = \frac{1}{\lambda^2}$

De exponentiële verdeling voldoet aan de Markov-eigenschap: ze bezit geen geheugen.

Bomen en grafen

4 Bomen

4.1 Terminologie

Een *gewortelde boom* T is een verzameling van *toppen* die aan de volgende eigenschappen voldoet:

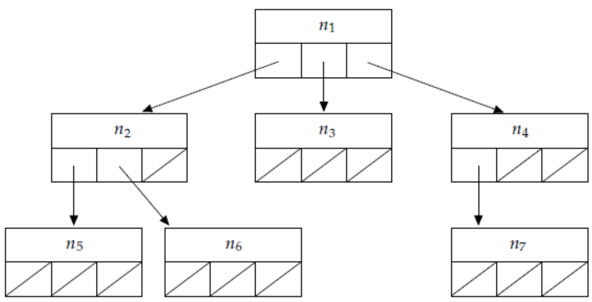
1. Er is één speciale top t die de *wortel* van de boom wordt genoemd.
2. De andere toppen zijn verdeeld in $m \geq 0$ disjuncte verzameling $T_1,\ldots,T_m$ die op hun beurt elk weer een gewortelde boom zijn.

$T_1,\ldots,T_m$ zijn *deelbomen* van T. De wortels $t_1,\ldots,t_m$ van de deelbomen $T_1,\ldots,T_m$ worden de *kinderen* van de wortel t genoemd. Omgekeerd is t de *ouder* van $t_1,\ldots,t_m$. De toppen $t_1,\ldots,t_m$ zijn *broers* van elkaar. De term *afstammeling* en *voorouder* zijn logische uitbreidingen van de kind/ouder terminologie.

Uit de recursieve definitie van een gewortelde boom volgt dat elke top in de boom uiteindelijk de wortel is van een deelboom die bevat is in de boom. Het aantal deelbomen van een top wordt de *graad* van die top genoemd. Een *blad* is een top met graad nul. Een top die geen blad is wordt *intern* genoemd. De *graad* van een boom wordt gedefinieerd als het maximum van de graden van zijn toppen.

4.2 Datastructuren voor bomen

4.2.1 Array-van-kinderen voorstelling



Figuur 1: Array-van-kinderen voorstelling

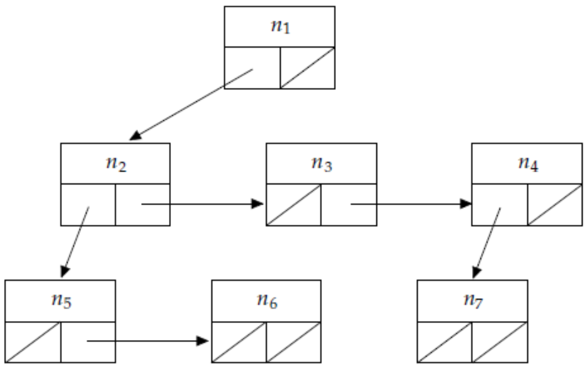
De eenvoudigste manier om een boom voor te stellen is door rechtstreeks de vaderkind relatie te implementeren. Dit betekent dat we een structuur Top definiëren die een veld heeft om de data van de top bij te houden, alsook een array van referenties naar de kinderen van die top. Wanneer een kind niet bestaat dan wordt dit voorgesteld door de referentie null. De boom zelf bestaat uit een referentie naar zijn wortel.

Aantal referenties: $n \times k$

Aantal gebruikte referenties: $n - 1$

$\frac{\text{aantal gebruikte}}{\text{totaal}} = \frac{n-1}{nk} \approx \frac{1}{k}$

4.2.2 Eerste-kind-volgende-broer voorstelling



Figuur 2: Eerste-kind-volgende-broer voorstelling

We kunnen een boom voorstellen op een manier die efficiënter met het geheugen omgaat. In plaats van in elke top referenties naar al zijn kinderen op te slaan, houden we altijd juist twee referenties bij: een referentie naar zijn eerste kind, en een referentie naar zijn volgende broer.

Aantal referenties: $2n$

Aantal gebruikte referenties: $n - 1$

$\frac{\text{aantal gebruikte}}{\text{totaal}} = \frac{n-1}{2n} \approx \frac{1}{2}$

4.3 Recursie op bomen

4.3.1 Alle toppen van een boom bezoeken

Om een boom te doorlopen in preorde gaat men als volgt tewerk:

1. Bezoek de wortel van de boom.
2. Doorloop alle deelbomen van de wortel in preorde.

Om een boom te doorlopen in postorde gaat men als volgt tewerk:

1. Doorloop alle deelbomen van de wortel in postorde.

2. Bezoek de wortel van de boom.

Dit proces zal eindigen want wanneer de boom slechts uit één top bestaat.

4.4 Binaire bomen

4.4.1 Definitie en eigenschappen

Een *binaire boom* is een verzameling toppen die

1. ofwel leeg is,
2. ofwel bestaat uit een wortel en twee disjuncte verzamelingen T_l en T_r, die op hun beurt ook een binaire boom zijn. We noemen T_l en T_r respectievelijk de linker- en rechterdeelboom van de wortel.

Eigenschappen

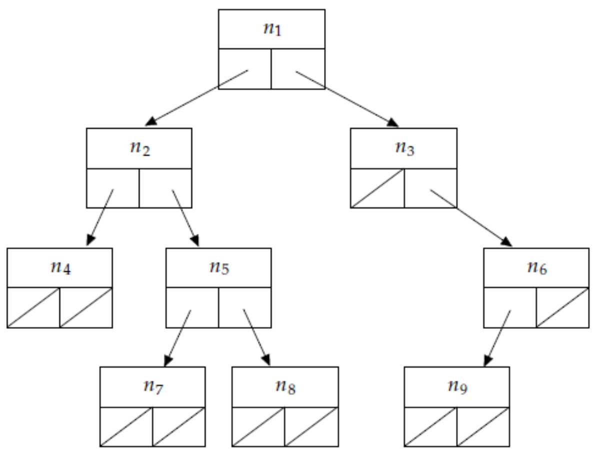
In een binaire boom is het aantal toppen met diepte k hoogstens 2^k.

Voor een (niet-lege) binaire boom T met een diepte $d \geq 0$ geldt dat: $d + 1 \leq \#(T) \leq 2^{d+1} - 1$

Bewijs:

$$\begin{aligned} \#(T) &= \textstyle\sum_{k=0}^{d} \text{aantal toppen met diepte k} \\ &\leq \textstyle\sum_{k=0}^{d} 2^k \\ &= 1 + 2 + 4 + \cdots + 2^k \\ &= 2^{d+1} - 1 \end{aligned} \tag{2}$$

4.4.2 Voorstelling binaire boom



Figuur 3: Interne voorstelling van een binaire zoekboom

4.4.3 Alle toppen van een binaire zoekboom bezoeken

Preorde:

1. Bezoek de wortel van de boom.
2. Als de linkerdeelboom niet leeg is, doorloop de linkerdeelboom dan recursief in preorde.
3. Als de rechterdeelboom niet leeg is, doorloop de rechterdeelboom dan recursief in preorde.

Postorde:

1. Als de linkerdeelboom niet leeg is, doorloop de linkerdeelboom dan recursief in postorde.
2. Als de rechterdeelboom niet leeg is, doorloop de rechterdeelboom dan recursief in postorde.
3. Bezoek de wortel van de boom.

Inorde:

1. Als de linkerdeelboom niet leeg is, doorloop de linkerdeelboom dan recursief in inorde.
2. Bezoek de wortel van de boom.
3. Als de rechterdeelboom niet leeg is, doorloop de rechterdeelboom dan recursief in inorde.

4.5 Binaire zoekbomen

Een *binaire zoekboom* is een gelabelde binaire boom die aan een bijzondere voorwaarde, de binaire zoekboomeigenschap, voldoet.

De *binaire zoekboomeigenschap* is de volgende: voor elke top x van de binaire zoekboom geldt dat alle toppen in de linkerdeelboom van x een label hebben dat kleiner is dan het label van x, terwijl voor alle toppen in de rechterdeelboom van x geldt dat hun label groter is dan het label van x.

4.5.1 Opzoeken van een sleutel in een binaire zoekboom

Om (recursief) te zoeken naar een bepaalde waarde x in een binaire zoekboom T volgt men de volgende stappen:

1. Wanneer de boom leeg is, geef dan 'niet gevonden' terug.
2. Vergelijk x met de sleutel van de wortel.
   (a) Wanneer x kleiner is dan dit label, zoek dan (recursief) in de linkerdeelboom.

(b) Wanneer x groter is dan dit label, zoek dan (recursief) in de rechterdeelboom.

(c) Geef de wortel van de boom terug (x werd gevonden).

4.5.2 Toevoegen van een sleuten aan een binaire zoekboom

We kunnen het toevoegen van een sleutel x aan een (niet-lege) binaire zoekboom als volgt recursief formuleren:

1. Vergelijk x met het label van de wortel.
   (a) Wanneer x kleiner is dan het label van de wortel, voeg dan x toe aan de linkerdeelboom wanneer die niet leeg is. Wanneer de linkerdeelboom leeg is vervang dan de (null)-referentie naar de linkerdeelboom door de referentie naar een nieuwe knoop met x als label.
   (b) Wanneer x groter is dan het label van de wortel, voeg dan x toe aan de rechterdeelboom wanneer die niet leeg is. Wanneer de rechterdeelboom leeg is vervang dan de (null)-referentie naar de rechterdeelboom door de referentie naar een nieuwe knoop met x als label.
   (c) Doe niets, want x behoort reeds tot de boom.

4.5.3 Verwijderen van een sleutel uit een binaire zoekboom

Het verwijderen van een sleutel x start met het opzoeken van deze sleutel in de boom. Er kunnen zich nu drie gevallen voordoen:

1. Sleutel is een blad:
   (a) Verwijder de referentie.
2. Sleutel is een top met één kind:
   (a) Referentie veranderen in ouder.
3. Sleutel is een top met 2 kinderen:
   (a) Zoek het minimum in de rechterboom.
   (b) Vervang het te verwijderen element door de gevonden waarde.
   (c) Verwijder het minimum in de rechterdeelboom.

4.5.4 Tijdscomplexiteit van de bewerkingen

$$\begin{aligned} n &= 2^{d+1} - 1 \\ &\approx 2^{d+1} \\ &\Rightarrow lg(n) \approx lg(2^{d-1}) = d + 1 \end{aligned} \tag{3}$$

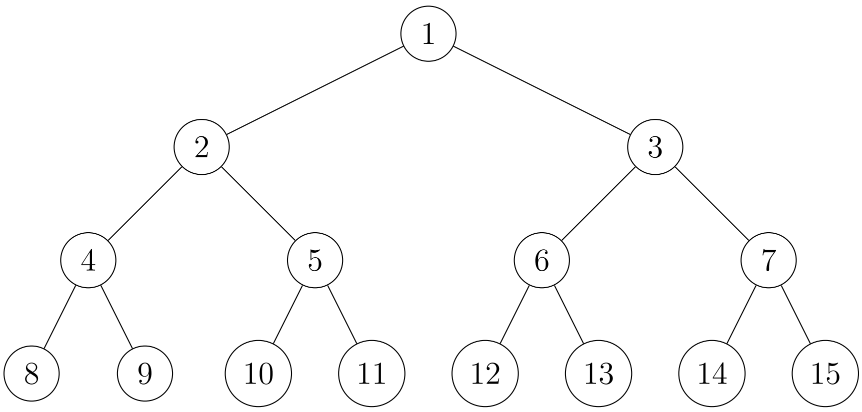
4.6 Binaire hopen

4.6.1 Prioriteitswachtrij als binaire hoop

Een *complete binaire hoop* is een binaire boom van diepte d waarbij het aantal toppen met diepte $k < d$ maximaal (dus 2^k) is. De toppen met diepte d komen voor van "links naar rechts".

De *ordeningseigenschap voor binaire hopen* zegt dat de sleutel van elke top hoogstens gelijk is aan de sleutel van zijn kinderen.

4.6.2 Implementatie van een binaire hoop



Figuur 4: Voorstelling van een binaire hoop

De boom uit figuur 4 kan opgeslagen worden als een array.

Wanneer een top rangnummer i heeft, dan hebben zijn linker- en rechterkind (als die bestaan) respectievelijk rangnummer $2i$ en $2i + 1$.

Omgekeerd geldt: wanneer een top rangnummer i heeft (en deze top is niet de wortel van de boom), dan heeft zijn ouder rangummer floor($i/2$).

4.6.3 Opzoeken van het element met de kleinste sleutel

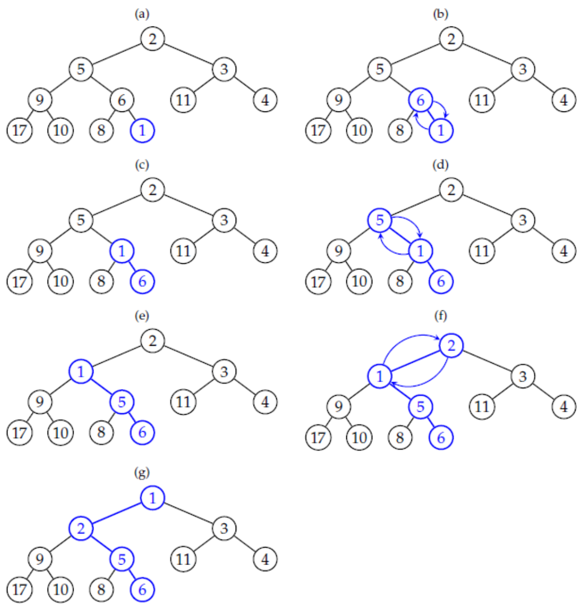
Uit de ordeningseigenschap voor binaire hopen volgt dat het element met de kleinste sleutel steeds de wortel van de boom is.

4.6.4 Toevoegen van een element

1. Creëer een nieuw element.

2. Voeg dit element toe op de eerste beschikbare plaats. Dit betekent dus als een nieuw blad, met rangnummer i, waarbij we er steeds voor zorgen dat het diepste niveau gevuld is van links naar rechts.
3. Op dit moment is het in het algemeen zo dat de ordeningseigenschap voor binaire bomen nu kan geschonden zijn tussen i en parent(i). Indien dit zo is, verwissel dan i en parent(i). Dit herstelt de ordeningseigenschap tussen i en zijn ouder. Eventueel is nu de ordeningseigenschap tussen parent(i) en parent(parent(i)) geschonden. Indien dit zo is wissel dan beide elementen. Ga zo verder tot de binaire hoop is hersteld.

Dit proces wordt geïllustreerd in figuur 5.



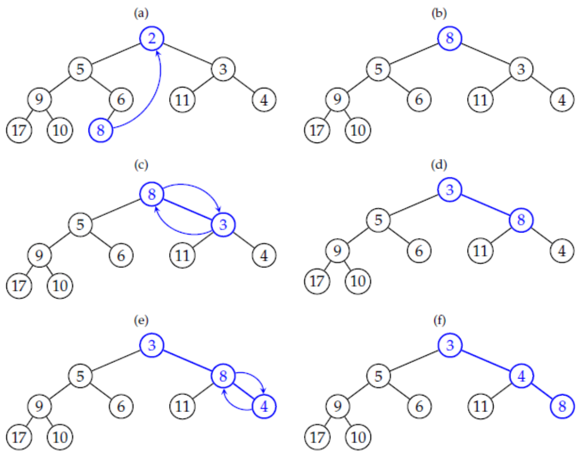
Figuur 5: Toevoegen van sleutel 1 aan een binaire hoop

Het proces van het herstellen van de binaire hoop van beneden naar boven, noemt men soms ook *omhoog bubbelen*, omdat het nieuwe element als het ware omhoog bubbelt in de hoop tot het op zijn correcte plaats staat.

4.6.5 Verwijderen van het element met de kleinste sleutel

1. Verwissel de wortel met het meest rechtse blad met de grootste diepte.
2. Verwijder het meest rechtse blad: de binaire hoop heeft nu een element minder.
3. Indien de ordeningseigenschap geschonden is in de wortel, herstel deze dan door de wortel en zijn kleinste kind i van plaats te verwisselen. Indien de ordeningseigenschap nu geschonden is in i, herstel ze dan door i te verwisselen met de kleinste van zijn kinderen. Ga zo verder tot de binaire hoop hersteld is.

Dit proces wordt geïllustreerd in figuur 6.



Figuur 6: Verwijderen van het kleinste element uit een binaire hoop

Het proces dat gebruikt wordt bij het verwijderen van een element noemt men soms ook *omlaag bubbelen*, omdat het laatste element van de hoop omlaag bubbelt doorheen de hoop tot het op zijn juiste plaats staat.