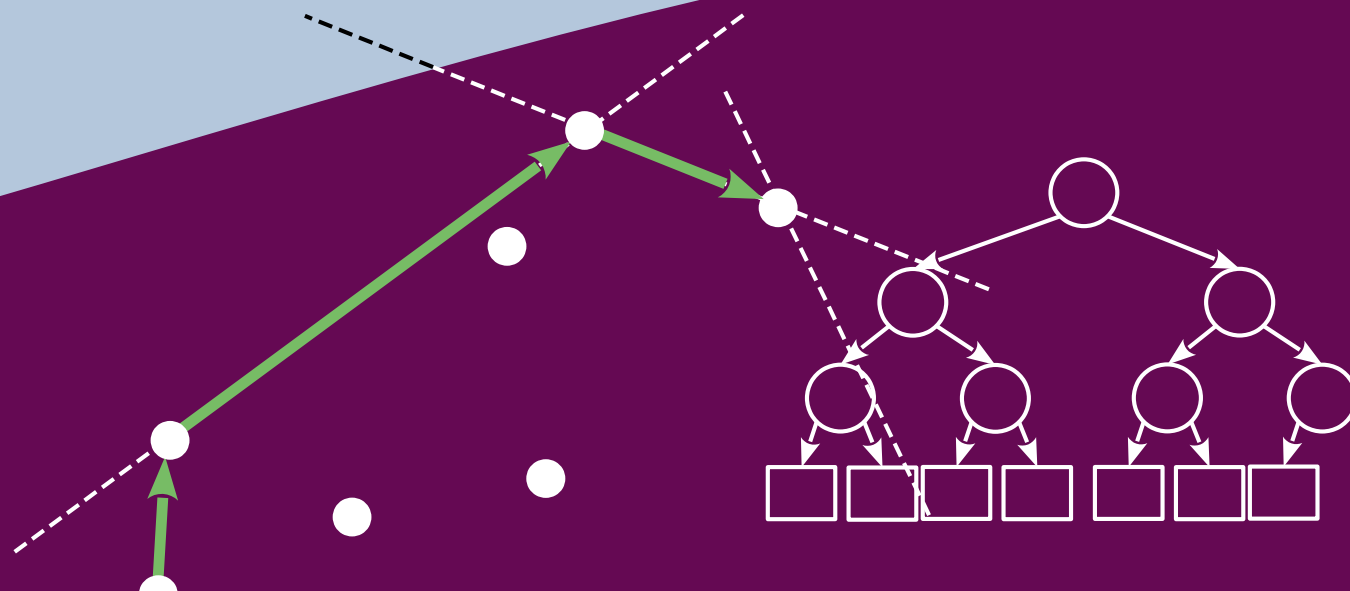
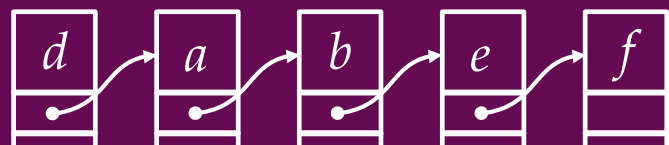
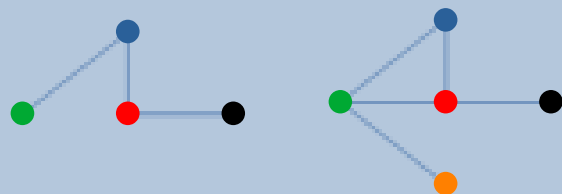


Napredni algoritmi i strukture podataka

Tjedan 7: Nasumični algoritmi

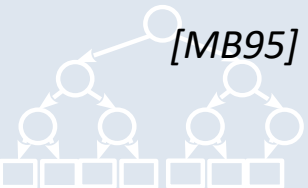


Nasumični algoritmi

- Osnove
- PRNG
- Nasumični quicksort
- Skip-liste

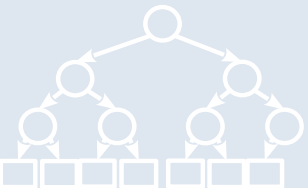
Predavanje bazirano na :

[MB95] R. Motwani, P.Raghavan „Randomized algorithms“, 1995; **predgovor i poglavlje 1**



Slučajnost?

- Potreba?
 - Otvoreno pitanje!
 - Suparničke igre (Nash,...)
- Resurs?
 - Može sačuvati druge resurse (vrijeme, memorija)



Motivacija

- Suparničke igre
 - Denial-of-service napadi
 - Napadi temeljeni na računalnoj složenosti
 - Suparničko strojno učenje
 - Krađa povjerljivih podataka
 - Kriptografija
- Lakše do boljih algoritama!
 - Jednostavnost i/ili brzina



Motivacija – suparničke igre

- Napadi temeljeni na računalnoj složenosti
 - Npr. quicksort
- Suparničko strojno učenje
 - [Randomization matters: How to defend against strong adversarial attacks](#), ICML 2020.
 - [On the robustness of randomized classifiers to adversarial examples](#), Arxiv 2021.



Motivacija – bolji algoritmi

- Testiranje primalnosti
 - Najbrži deterministički algoritam – $O(n^5)$
 - Najbrži probabilistički algoritam – $O(n^3)$

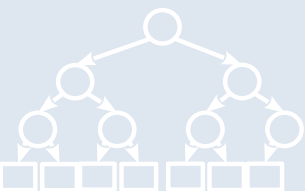


Motivacija – paradigma dizajna algoritama

1. Izrada efikasnog probabilističkog algoritma
2. Derandomizacija (složeno!)
3. Dobiven efikasni deterministički algoritam

- Primjeri

- Test primalnosti u polinomijalnom vremenu
 - [probabilistički algoritam](#) (2003) -> [deterministički algoritam](#) (2004)
- Test povezanosti neusmjerenog grafa u logaritamskom prostoru
 - [probabilistički algoritam](#) (1979) -> [deterministički algoritam](#) (2008)



Nasumični algoritmi

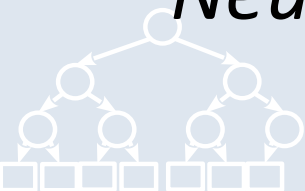
- Algoritmi sa pristupom izvoru nezavisnih, nepristranih slučajnih bitova
 - Slučajni bitovi utječu na izračune
- Algoritmi sa teorijskim jamstvima!
- Stohastički algoritmi – nisu tema ovog predavanja
 - Nemaju čvrstih teorijskih jamstava, „samo” empirijski rezultati
 - Optimizacijski algoritmi: poput evolucijskih, tabu-pretraživanja, simuliranog kaljenja...



Nasumični algoritmi

- Algoritmi sa pristupom izvoru nezavisnih, nepristranih slučajnih bitova
 - Slučajni bitovi utječu na izračune
- Pseudo-slučajni brojevi

"Anyone who considers arithmetical methods of producing random digits is, of course, in a state of sin., *John von Neumann*



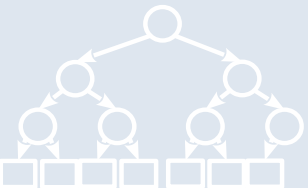
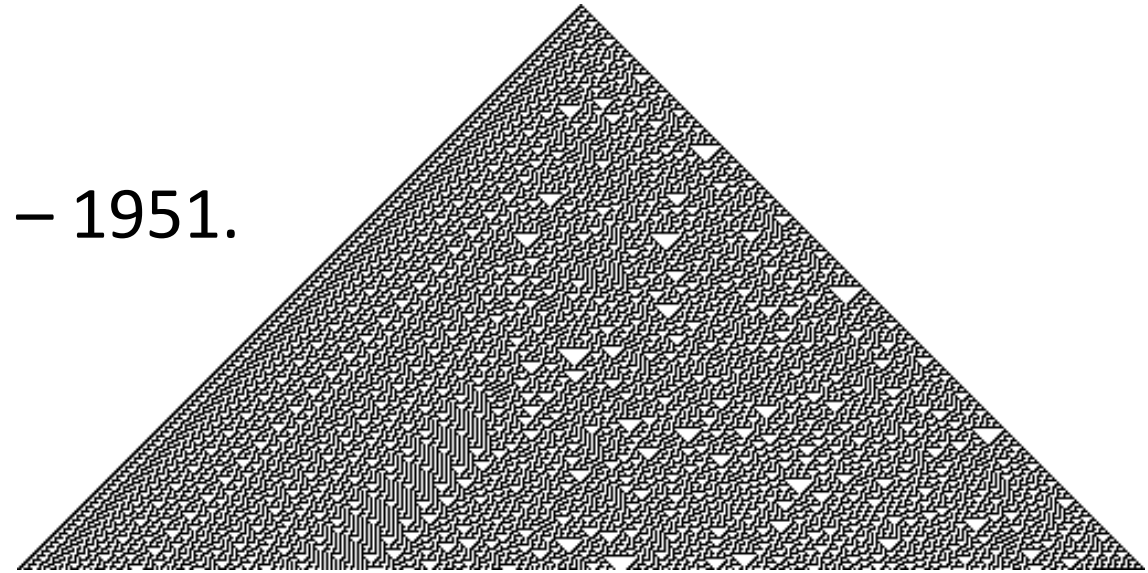
Nasumični algoritmi - paradigme

1. Prevara protivnika
2. Slučajno uzorkovanje -> npr. iz populacije
3. Pronalazak „svjedoka” -> npr. testovi za dokaze
4. Fingerprinting i hashing
5. Slučajno preraspoređivanje
6. Balansiranje opterećenja -> raspodijeljeno računarstvo
7. Brzo-miješajući Markovljevi lanci -> približna brojanja
8. Izolacija i razbijanje simetrija -> koordinacija
9. Probabilističke metode i dokazi postojanja -> $p > 0$



Generatori pseudo-slučajnih brojeva

- Bogat razvoj
- Linearni kongruentni generatori (LCG) – 1951.
- [Pravilo 30](#) – 1983.
- [Generator srednjeg kvadrata i Weylovog slijeda](#) – 2017.

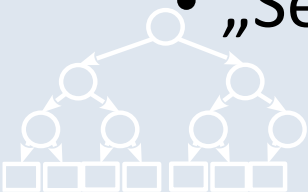


Linearni kongruentni generator

- Jedan od najjednostavnijih
- Široko korišten (C, Java)

$$X_{n+1} = (aX_n + c) \bmod m$$

- Konstante: a, c, m
- „Seed”: X_0



Nasumični algoritmi - vrste

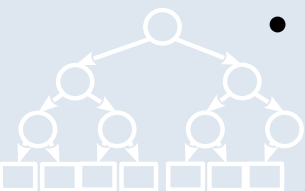
- Las Vegas

- Uvijek daje točan odgovor ili info o neuspjehu
- Vrijeme izvođenja varira

- Monte Carlo

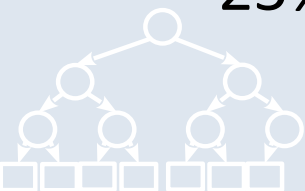
- Vrijeme izvođenja ograničeno
- Nekad ne vraća točan odgovor
 - Neuspjeh
 - Netočan odgovor

- Nezavisna uzastopna pokretanja -> proizvoljno smanjenje šanse neuspjeha



Klase složenosti za probleme odlučivanja

- **P** – svi koji mogu biti riješeni u polinomijalnom vremenu
- **NP** – svi za koje verifikacija rješenja može biti obavljena u polinomijalnom vremenu
- **ZPP** (zero-error probabilistic polynomial) – svi koji imaju Las Vegas algoritam sa očekivanim polinomijalnim trajanjem
- **RP** (randomized polynomial) – svi koji imaju Monte Carlo algoritam sa jednostranom greškom i polinomijalno trajanje u najgorem slučaju
- **PP** (probabilistic polynomial) – svi koji imaju Monte Carlo algoritam sa dvostranom greškom (ne gore od 50%) i polinomijalno trajanje u najgorem slučaju
- **BPP** (bounded-error probabilistic polynomial) – PP, ali greška ne gora od 25%



Nasumični quicksort

- Quicksort

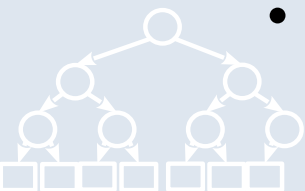
- najbolji slučaj $O(n \cdot \log n)$
- najgori slučaj $O(n^2)$

Quicksort(lo,hi,A):

```
1. if lo >= 0 && hi >= 0 && lo < hi then  
2.     p := partition(A, lo, hi) // pivot selection, splitting  
3.     quicksort(A, lo, p) // Note: the pivot is now included  
4.     quicksort(A, p + 1, hi)
```

- deterministička fja partition

- najgori slučaj $O(n^2)$ – otvoreno za napade bazirane na složenosti



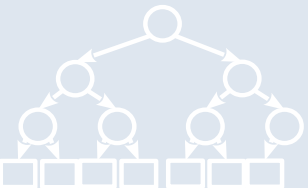
Nasumični quicksort

- nasumična fja `partition`
 - Uniformno slučajan odabir pivota iz danog raspona
 - najgori slučaj $O(n^2)$ – iščezavajuća vjerojatnost
 - U očekivanju $O(n \cdot \log n)$ *
 - za svaki ulaz
 - čak i protiv „neprijatelja”
 - Vrijeme izvođenja slučajno, čak i za ponovljeni ulaz
- Odvajanje (decoupling) strukture ulaza od funkcioniranja algoritma



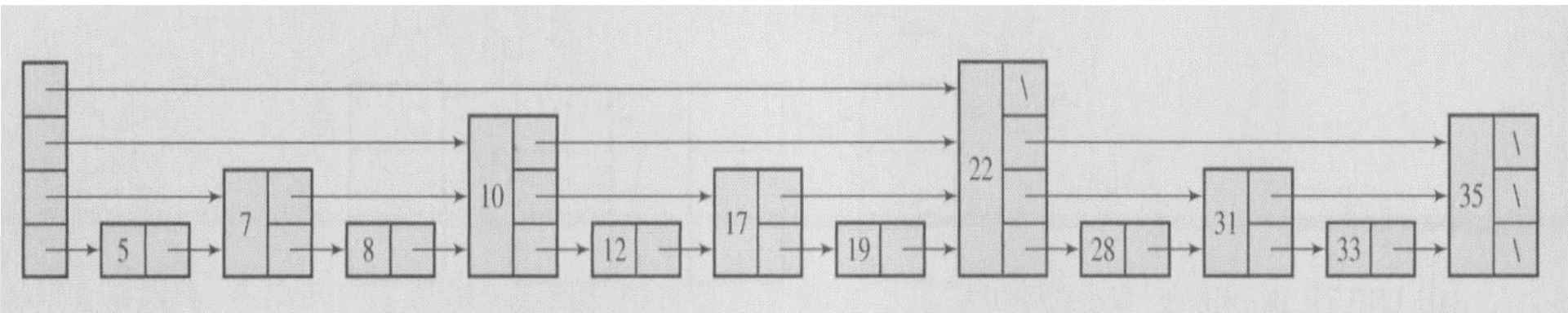
Preskočne liste

- [Pugh, William: "Skip lists: a probabilistic alternative to balanced trees", Communications of the ACM 33, June 1990, pp. 668-676](#)
- **Osnovni nedostatak lista:** $O(n)$ pretraživanje
- **Ograničavajuće svojstvo stabala:** po prirodi hijerarhijske strukture, logički neprikladne za sve primjene.
- Skip liste:
 - ključne operacije $O(\log_2 n) \dots O(n)$
 - **nema hijerarhije**
 - relativno jednostavno programiranje
 - [Paralelni pristup!!!](#)



Preskočne liste – izvod ideje iz stabala

- Stupanj (level) čvora = broj pokazivača
 - U primjeru niže, stupanj glave = 4
- Održavanje ovakve savršene strukture
 - Komplicirano i neučinkovito – „glumi” stablo
 - Restrukturiranje svih čvorova iza promjene

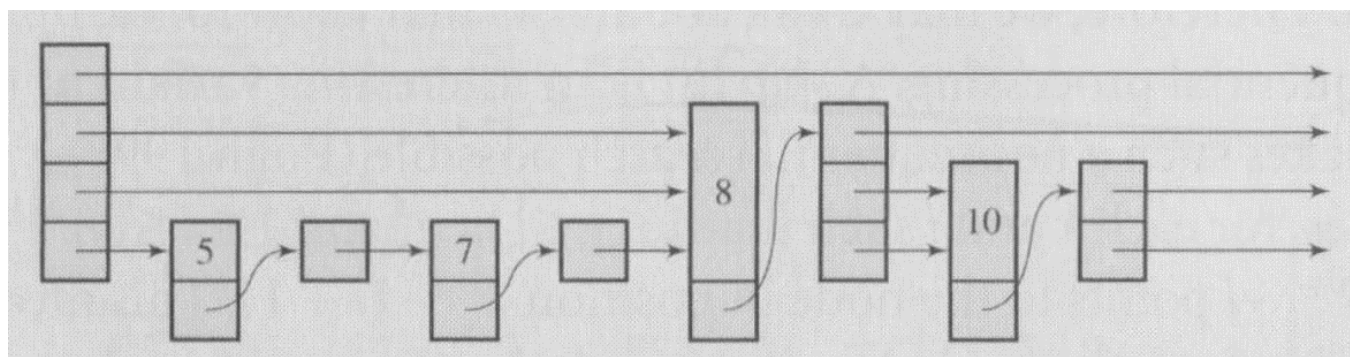
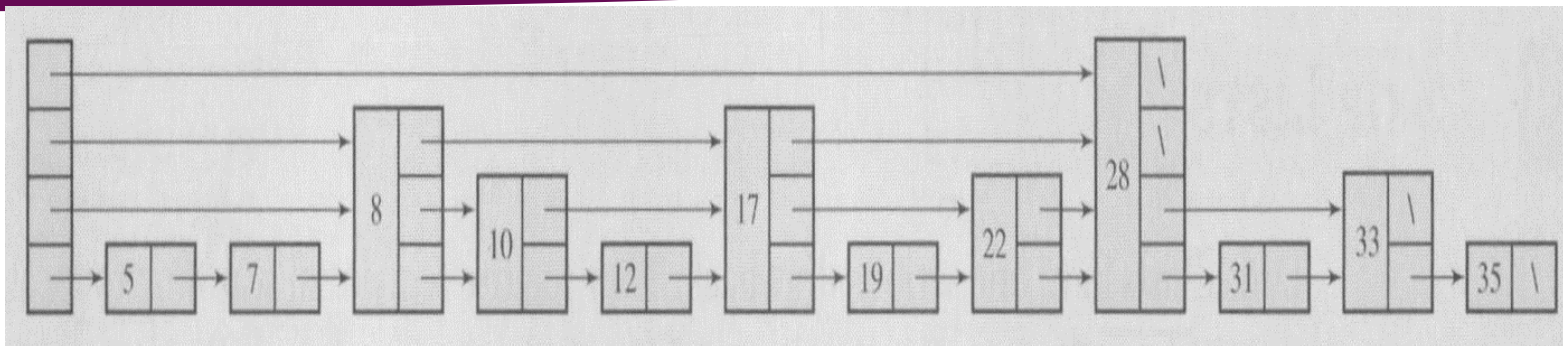


Preskočne liste

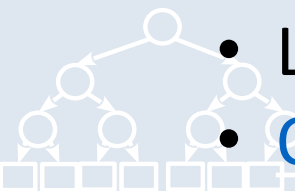
- **Odustajanje** od zahtjeva za pravilnim hijerarhijskim rasporedom čvorova
- **nastoji** se vjerojatnosno postići pravilna *razdioba* njihovih stupnjeva.
 - Histogram broja čvorova po stupnjevima bi trebao težiti histogramu idealnog slučaja



Preskočne liste

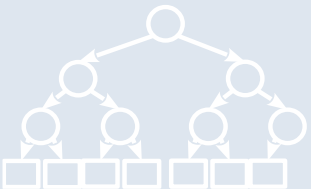


- Brzina pristupa podacima je u **prosjeku** sumjerljiva brzini u AVL ili RB stablu
- Lošija jamstva za najgori slučaj
- Odnos između stabala i preskočne liste




Preskočne liste

- Uporabna struktura skip liste ovisi o dva čimbenika:
 1. predviđenom kapacitetu n
 - pretpostavljeni najveći broj elemenata u listi
 2. vjerojatnosti pojedinih stupnjeva čvora
 - određuje razdiobu
 - najčešće se odabire samo vjerojatnost p prijelaska čvora u višu razinu
 - definira **geometrijsku razdiobu**
 - prijelasci u višu razinu ponavljaju se sve do prvog neuspjeha



Preskočne liste

- **Kapacitet** n i **vjerojatnost** p određuju sve teorijske značajke preskočne liste
- Vjerojatnost $P(k)$ da novi čvor u konačnici postigne k -ti stupanj
- $P(k) = [P(\text{prijelaz})]^{k-1} \cdot P(\text{ostanak}) = p^{k-1} \cdot (1 - p)$



Geometrijska razdioba
- $k-1$ uspješnih preskoka u višu razinu i jedan, konačni, neuspjeh



Preskočne liste

- Očekivani (“srednji”) stupanj čvorova u listi (“srednja visina” liste) predviđenom kapacitetu n

$$E(k) = \sum_{k=1}^{\infty} k \cdot P(k) = (1-p) \sum_{k=1}^{\infty} k \cdot p^{k-1}$$

$$\sum_{k=1}^{\infty} k \cdot p^{k-1} = \frac{1}{(1-p)^2}$$

$$\Rightarrow E(k) = \sum_{k=1}^{\infty} k \cdot P(k) = \frac{1}{1-p}$$



Preskočne liste

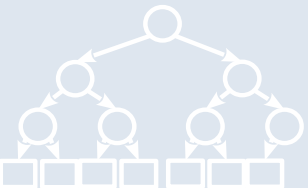
- Točan broj n_k čvorova k -tog stupnja je slučajna varijabla pa se može izračunati **očekivanje** $E(n_k)$

- n_k s n ukupno ubačenih brojeva u skip listu ima binomnu razdiobu

$$n_k \sim B(n_k; n, P(k))$$

- Prema tome, očekivanje $E(n_k)$ je

$$E(n_k) = n \cdot P(k) = n \cdot p^{k-1} \cdot (1 - p)$$



Preskočne liste – konstruiranje – broj razina

- U savršeno konstruiranoj skip-listi bit će samo jedan čvor najvišeg stupnja h

$$\begin{aligned} n \cdot p^{h-1} &\geq 1 \\ h &\leq 1 + \log_p \frac{1}{n} = 1 + \log_{\frac{1}{p}} n \end{aligned}$$

- Uzimamo

$$h = \text{floor}(1 + \log_{\frac{1}{p}} n)$$

Primjer: $p=0.5$, $n=12$

$$h = \text{floor}(1 + \log_2 12) = \text{floor}(4.6) = 4$$



Preskočne liste – konstruiranje – uzorkovanje

1. Uzastopno uzorkovanje uspona sa prekidom na h
 - „Višak” se rasporedi na najvišoj razini
- Direktno uzorkovanje – koristi samo jedan sluč.broj po umetanju
 2. Iz odrezane kumulativne distribucije $F(k)$ -> prekid na h
 - „Višak” se rasporedi na najvišoj razini
 3. Iz kvantizirane kumulativne distribucije $H(k)$
 - Kvantizacija - zaokruživanje
 - „višak” se rasporedi po histogramu



Preskočne liste – konstruiranje – uzorkovanje 1

- Uzastopno uzorkovanje uspona sa prekidom na h
 - „Višak” se rasporedi na najvišoj razini

```
randomLevel()
```

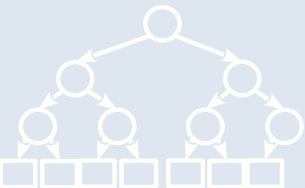
```
  lvl := 1
```

```
  -- random() that returns a random value in [0...1)
```

```
  while random() < p and lvl < MaxLevel do
```

```
    lvl := lvl + 1
```

```
  return lvl
```



Preskočne liste – konstruiranje – uzorkovanje 2

- Direktno uzorkovanje iz „odrezane“ kumulativne distribucije $F(k)$ (prekid na h)
 - „Višak“ se rasporedi na najvišoj razini
 - $F(k) = P(x \leq k); F(h) = 1$
- randomLevelDirect()
1. $|v| = 1$
 2. $r = \text{random}()$
 3. while $r > F(|v|)$:
 1. $|v| := |v| + 1$
 4. return $|v|$



Preskočne liste – konstruiranje – histogram

- Složimo kumulativni histogram prema očekivanju (prethodni slideovi)

$$\mathbb{E}(n_x \leq k) = n \cdot (1 - P(x > k)) = n \cdot (1 - p^k)$$

Broj čvorova razine manje ili jednake k u kumulativnom histogramu:

$$H(k) = \text{ceil}(\mathbb{E}(n_{x \leq k})) = \text{ceil}(n \cdot (1 - p^k))$$

*trivijalno, $H(0) = 0$



Preskočne liste – konstruiranje – histogram

- $p=0.5$, $n=12$, $h=4$

$$H(k) = \text{ceil}(n \cdot (1 - p^k))$$

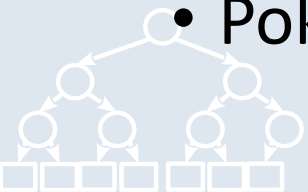
$$H(1) = \text{ceil}(12 \cdot (1 - 0.5^1)) = 6$$

$$H(2) = \text{ceil}(12 \cdot (1 - 0.5^2)) = 9$$

$$H(3) = \text{ceil}(12 \cdot (1 - 0.5^3)) = \text{ceil}(10.5) = 11$$

$$H(4) = \text{ceil}(12 \cdot (1 - 0.5^4)) = \text{ceil}(11.25) = 12$$

- Pokušaj izračuna za sve više razine daje $H(k) = 12$



Preskočne liste – konstruiranje – uzorkovanje 3

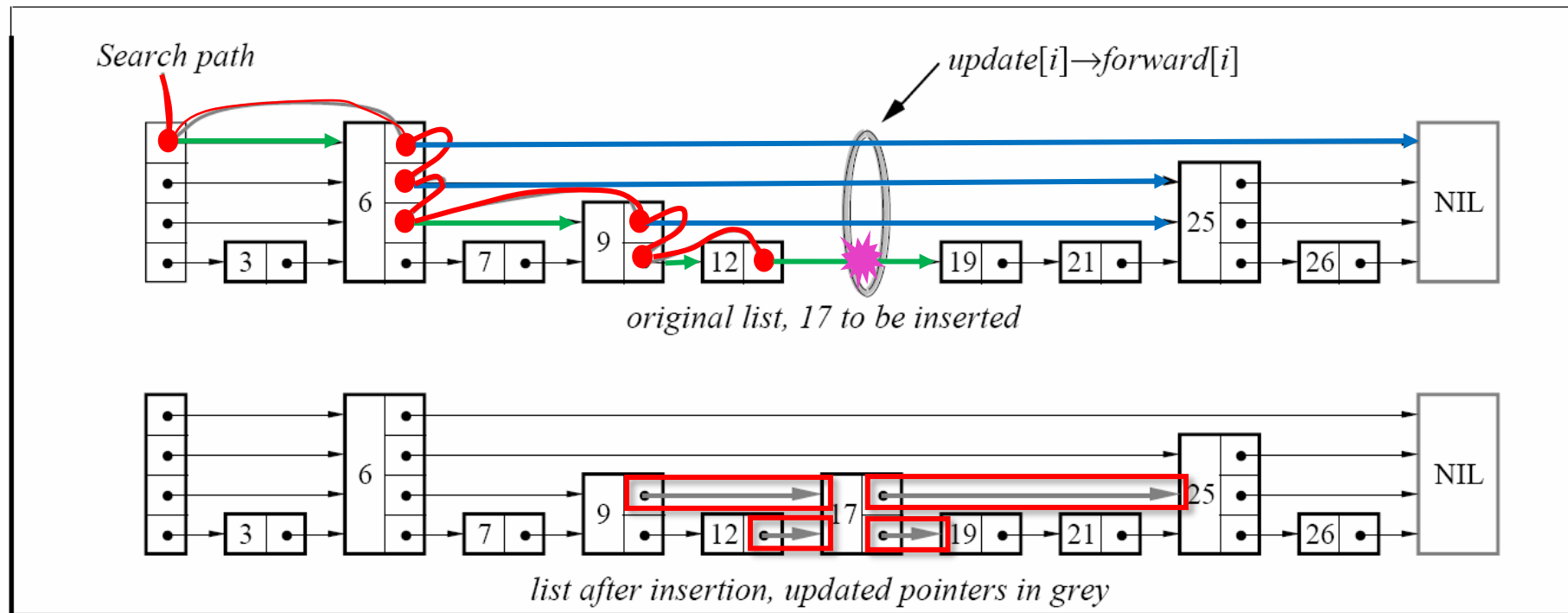
- Napravimo polje H duljine $h+1$ za kum.histogram

randomLevelDirectHist()

1. $|v| = 1$
2. $r = \text{randint}(1, n)$ // cijeli broj iz $[1, n]$
3. while $r > H[|v|]$:
 1. $|v| := |v| + 1$
4. return $|v|$



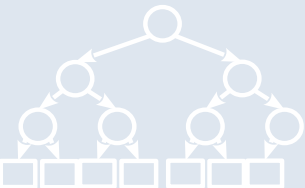
Napredne teme



Brisanje?

Preskočne liste - primjena

- Alternativa stablima za primjene sa velikim paralelizmom
 - Jednostavnije implementacije za lock-free operacije
 - Veće zauzeće memorije za brži pristup
 - Jednostavnije operacije umetanja i brisanja
- Inspiracija za algoritme
 - [približni najbliži susjedi](#)(2016)
 - [najkraći putevi u cestovnoj mreži](#)(2015)
 - [dinamička polja u kvantnim algoritmima](#)(2021)



Nasumični algoritmi – napredne teme

- Otvoreno pitanje $BPP=P$?
 - Nasumičnost pomaže, ALI...
 - PRNG
 - Derandomizacija
- Kriptografija, teorija računalnog učenja, raspodijeljeno računarstvo
 - blockchain
- Proširenje i definiranje pojmova
 - Znanje, tajnost, učenje, dokaz, slučajnost
- Interaktivni sustavi dokaza, vjerojatnosno provjerljivi dokazi

