TÖL403G GREINING REIKNIRITA

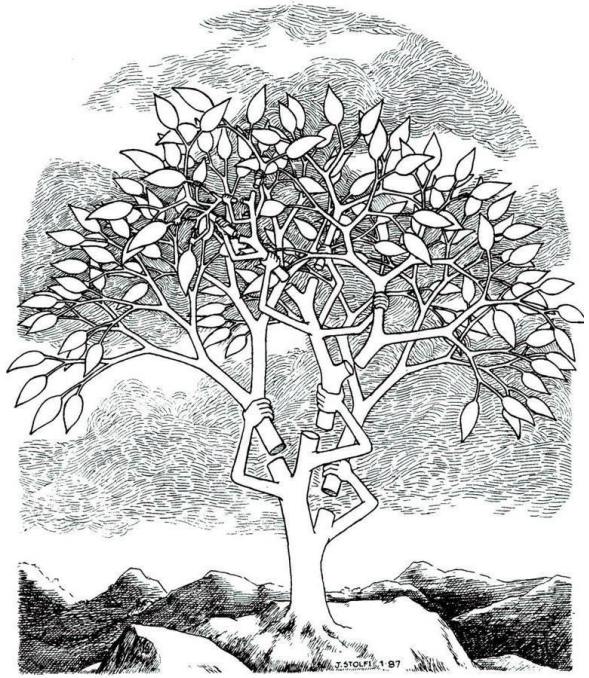
19. Jafnaðargreining 3

Hjálmtýr Hafsteinsson
Vor 2022

Í þessum fyrirlestri

- Sjálfstillandi (*self-adjusting*) gagnagrindur

- Splay tré (*splay tree*)
  - Splay aðgerðin
  - Leit, Innsetning, Eyðing
  - Jafnaðargreining



DC 10.5 − 10.6

Sjá líka Wikipedia

Sjálfstillandi listi

- Tengdur listi sem umraðar stökunum til að auka leitarhraða
  - Færa algeng stök framar í listann

Nokkrar útgáfur:

- Færa-fremst (*move-to-front*)
  - Ef leitað að staki *x* þá er það fært fremst í listann
- Teljari (*counter*)
  - Öll stök listans hafa teljara sem er hækkaður við leit að því
  - Listanum er haldið í lækkandi röð eftir teljaragildi
- Víxlun (*transpose*)
  - Ef leitað er að staki *x* þá er því víxlað við stakið fyrir framan í listanum

Lagar sig fljótt að breytingum í dreifingu gagna

En óalgeng gildi fara strax fremst og það tekur tíma að færa þau aftar í listann

Aukakostnaður við teljara og stök geta lifað of lengi á "fornri frægð"

Getur tekið langan tíma að bregðast við breytingum í gagnadreifingu

Splay tré (*splay trees*)

- Sett fram af Dan Sleator og Robert Tarjan árið 1985
  - Sjálfstillandi tvíleitartré með $O(\log(n))$ jafnaðartíma á öllum aðgerðum

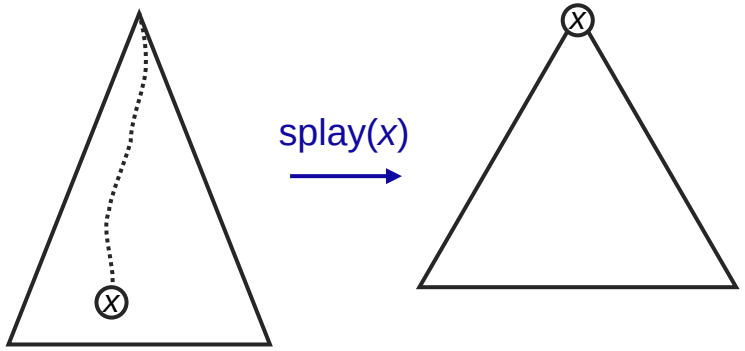
m aðgerðir á tréð taka í versta falli $O(m \log(n))$ tíma

- Inniheldur aðgerðina *splay* (ísl: glenna?)
  - Þegar leitað er að lyklinum *x* þá er tréð endurskipulagt þannig að *x* sé rót trésins
  - Notar til þess sérstakar útgáfur af trésnúningum (*tree rotations*)

Svipað og *færa-fremst* í lista

- Ávinningur af *splay*-aðgerð:
  - Færir lykilinn *x* efst í tréð
  - Færir forfeður *x* líka nær rótinni
  - Setur tréð (oftast) í meira jafnvægi

Fljótlegt að finna hann næst



Færa *x* í rótina

- Við gætum fært *x* upp í rótina með venjulegum trésnúningum

Endurtökum þetta þar
til x er orðin rót trésins

- Þetta gengur því miður ekki!

Ef við viljum $O(\log(n))$
jafnaðartíma per aðgerð

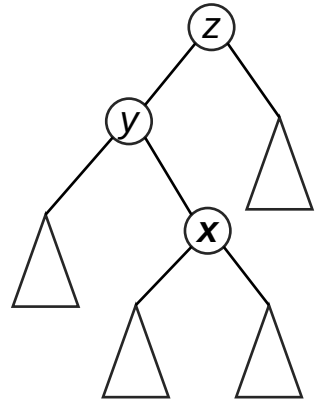
Hægt að finna runu *Finna*-aðgerða, þar sem hver aðgerð tekur $O(n)$ tíma

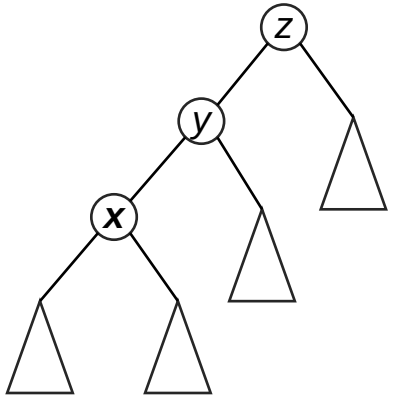
Þegar einn lykill *x* fer efst þá ýtir hann öðrum lykli *y* langt niður í tréð

Svo þegar við leitum að *y*, og færum hann efst, þá ýtir hann *x* langt niður í tréð

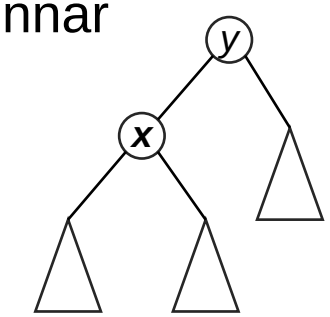
Splay snúningar

- Þurfum því aðeins flóknari trésnúninga, sem auka jafnvægi trésins
- Höfum tvær gerðir snúninga sem færa *x* upp um <u>tvö lög</u> í trénu:
  - ***Zig-Zig*** - ef *x* er vinstra barn vinstra barns, eða hægra barn hægra barns
  - ***Zig-Zag*** - ef *x* er hægra barn vinstra barns, eða öfugt

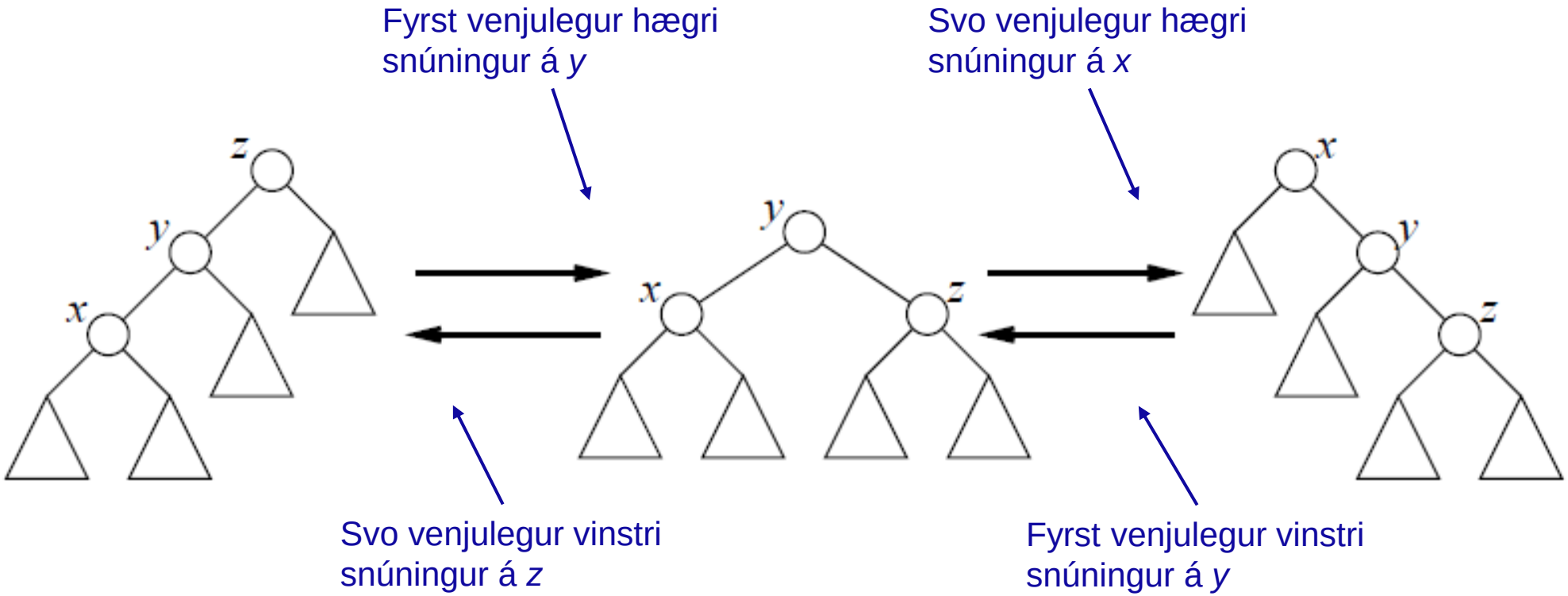




- Svo er mögulega einn ***Zig*** lokasnúningur
  - Þegar *x* er vinstra/hægra barn rótarinnar



Zig-Zig snúningur

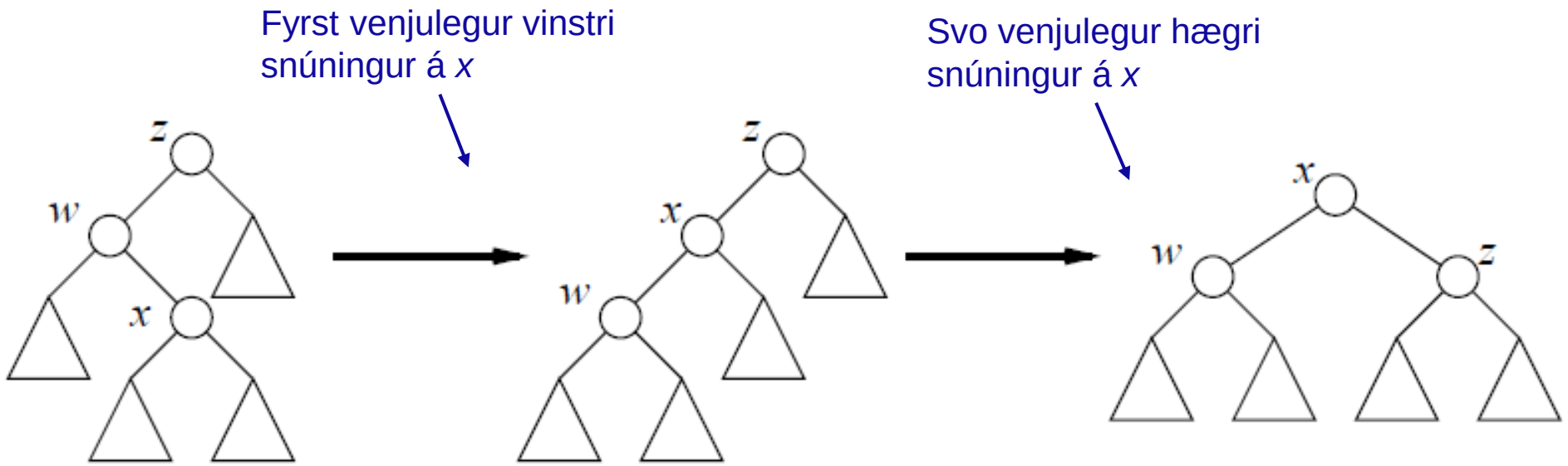


Þetta er ekki eins og tveir snúningar á *x* til að færa hann upp í rótina!

- Fyrsti snúningurinn er á foreldri *x*

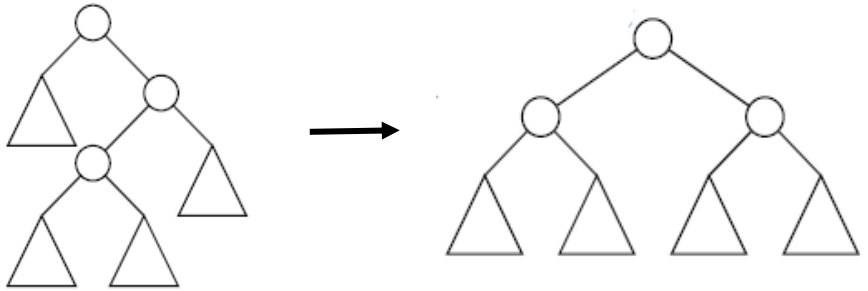
Zig-Zag snúningur

Þegar *x* er hægra barn vinstra barns:



Samhverfa tilfellið er svipað

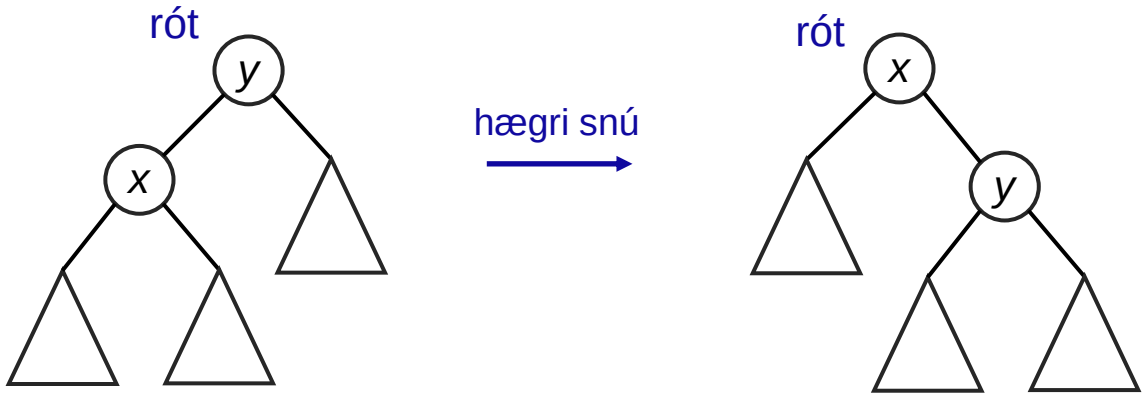
þ.e. þegar *x* er vinstra barn hægra barns



Zig snúningur − sértilvik fyrir rót

Þegar *x* er barn rótarinnar er ekki hægt að nota ***Zig-Zig*** eða ***Zig-Zag***

Þá notum við bara venjulegan trésnúning

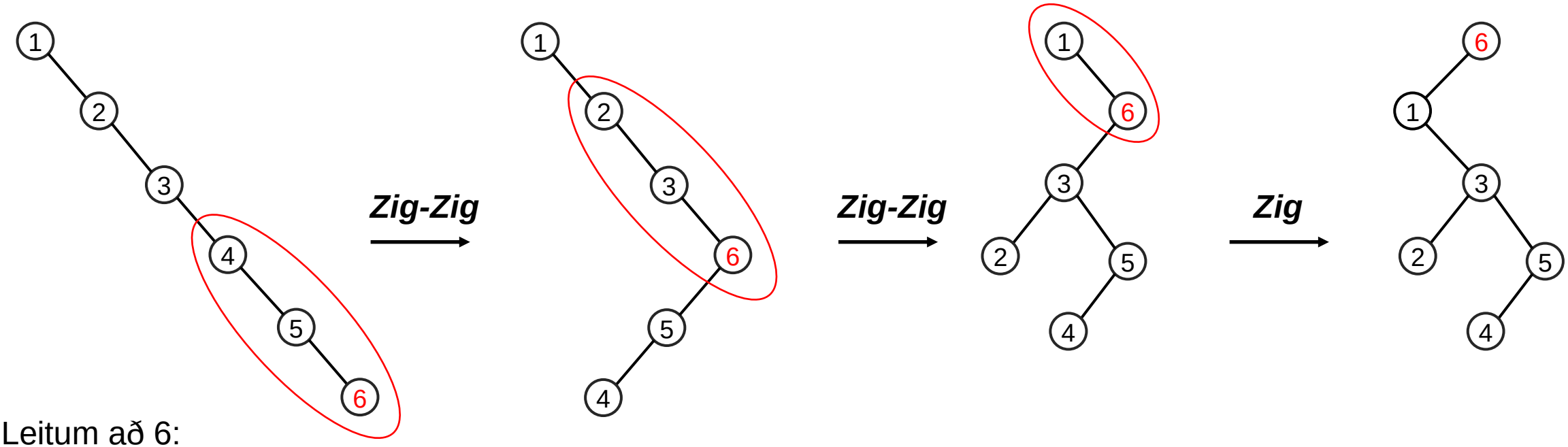


Sambærilegt þegar
x er hægra barn rótar

Þessi snúningur er gerður í mesta lagi einu sinni í hverri splay-aðgerð

Sýnidæmi

- Setjum inn gildin 6, 5, 4, 3, 2, 1 (í þessari röð):

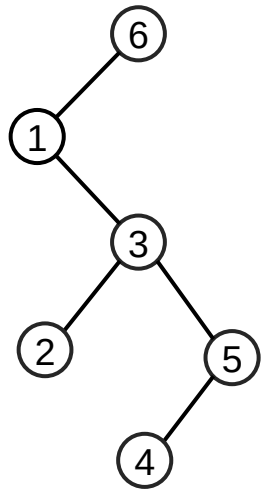


Leitum að 6:

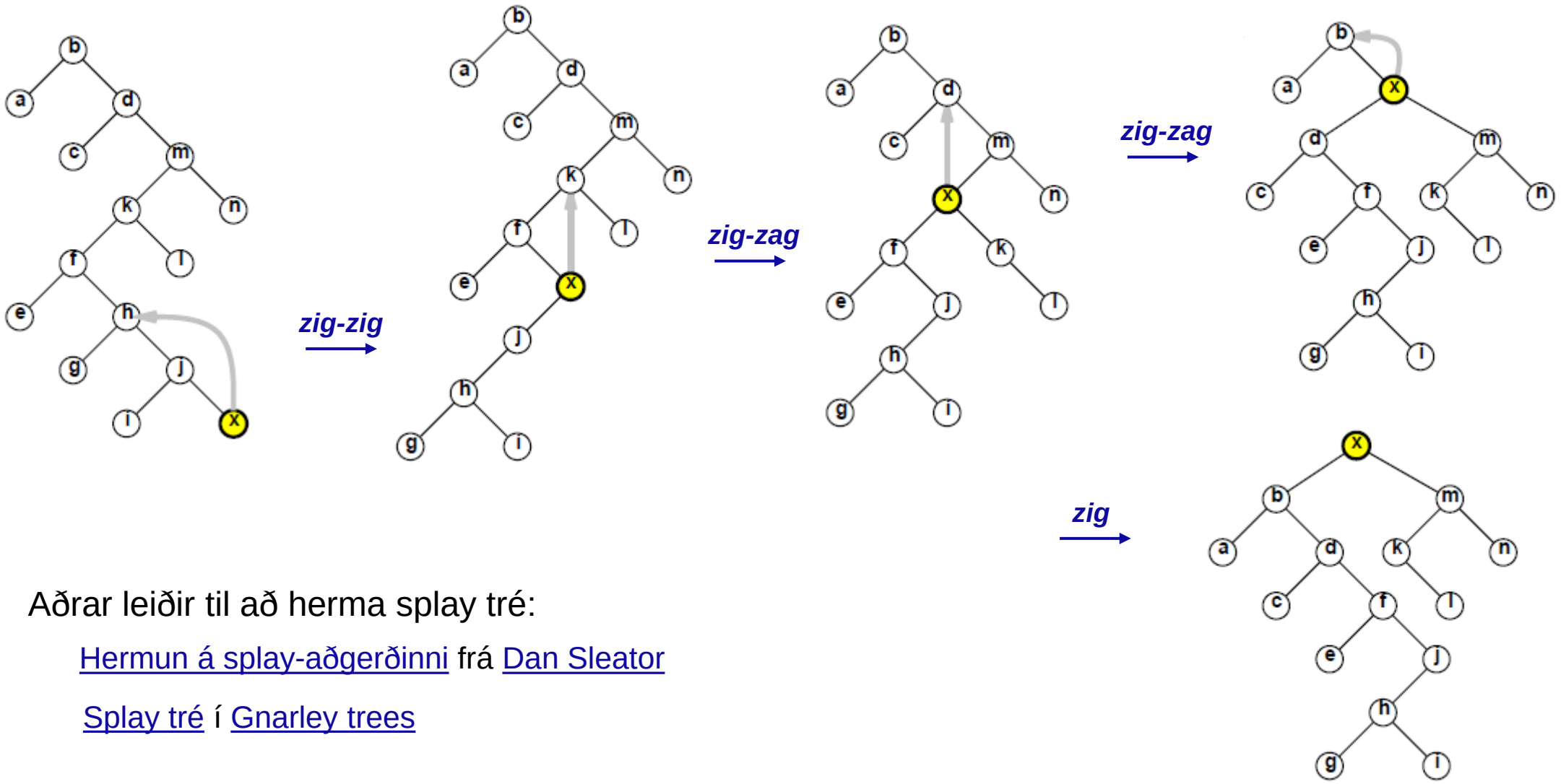
Förum niður tréð, finnum 6
og framkvæmum *splay*(6)

Æfing

- Leita að lyklinum 4 í trénu
  - Hver er fyrsti snúningurinn?
  - Hver er annar snúningurinn?



Stærra sýnidæmi



Aðrar leiðir til að herma splay tré:

Hermun á splay-aðgerðinni frá Dan Sleator

Splay tré í Gnarley trees

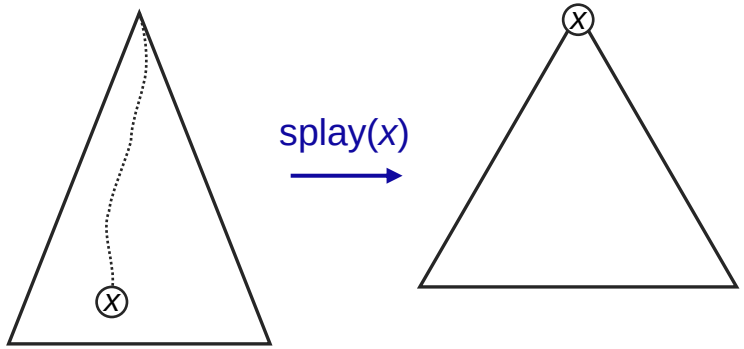
Aðrar aðgerðir á splay tré

- Allar aðrar aðgerðir nota splay-aðgerðina:

- *Finna*(*x*) (*search*)
  - Leita að lyklinum *x* í trénu og skila hnútinum
- *Innsetning*(*x*) (*insert*)
  - Setja lykilinn *x* í tréð, ef hann er þar ekki fyrir
- *Eyða*(*x*) (*delete*)
  - Eyða hnúti með lyklinum *x*, ef hann er í trénu
- *Kljúfa*(*x*) (*split*)
  - Skiptir trénu upp í tvö tré *S* (með lykla ≤*x*) og *T* (með lykla >*x*)
- *Sameina*(*S*, *T*) (*join*)
  - Sameinar trén *S* og *T* (allir lyklar í *S* < en allir lyklar í *T*)

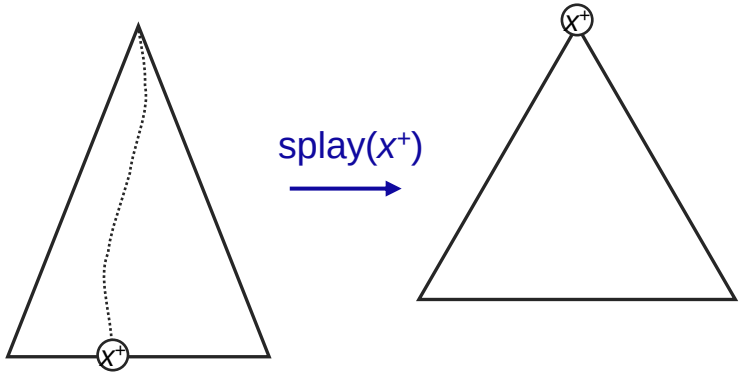
Aðgerðin *Finna*(*x*)

- Leita að *x* á venjulegan hátt í tvíleitartré



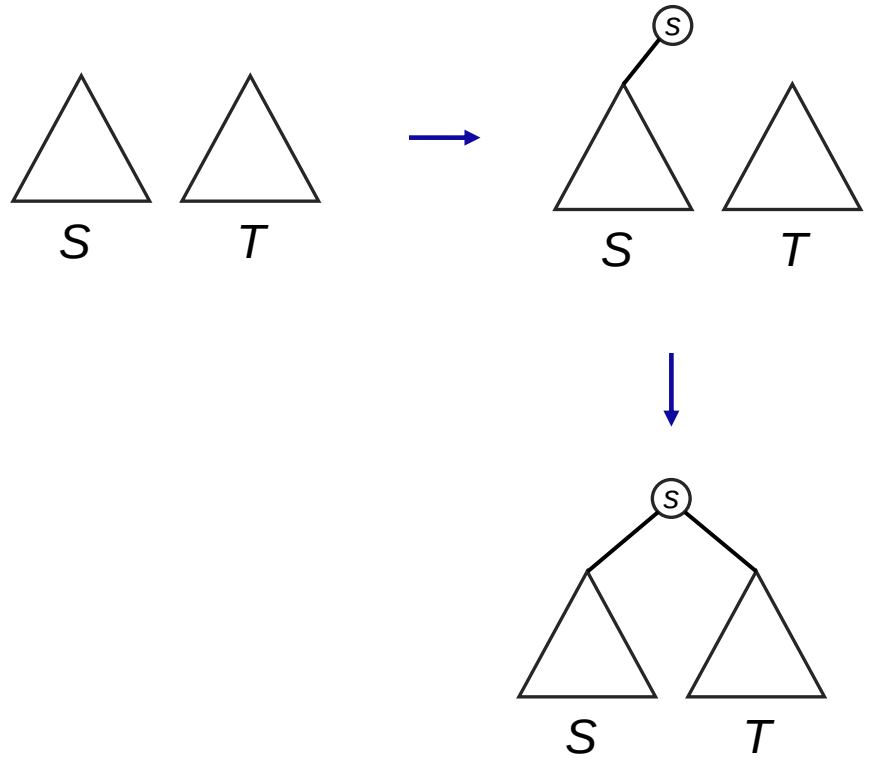
- Ef *x* finnst þá færa það upp í rót með splay(*x*)

- Ef *x* finnst ekki þá færa síðasta séða hnút upp í rót með splay-aðgerð
  - Þessi hnútur inniheldur næsta gildi fyrir neðan *x* í trénu (x^-, *predecessor*) eða næsta gildi fyrir ofan *x* í trénu (x^+, *successor*)



Aðgerðin *Sameina*(*S*, *T*)

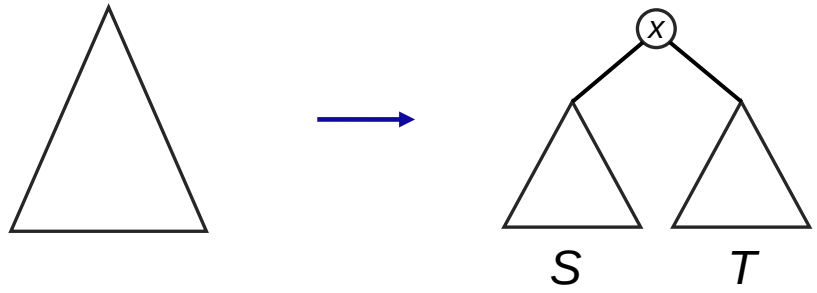
- Gefin tvö tvíleitartré *S* og *T*, með alla lykla í *S* minni en alla lykla í *T*:
- Finna stærsta lykilgildið í *S* (*Finna*(+∞) í *S*)
  - Þetta stak, *s* er nú rótin í *S*, með ekkert hægra barn
- Setja *T* sem hægra hluttré rótarinnar í *S*

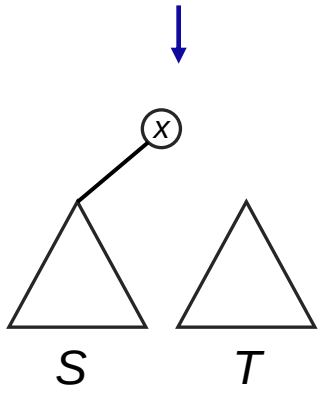


Hefðum líka geta fundið minnsta lykilinn í *T* (*Finna*(−∞) í *T*)

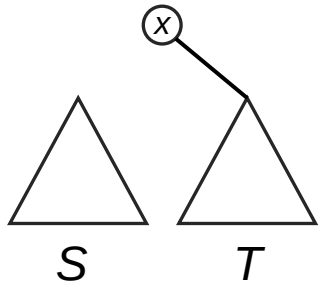
Aðgerðin *Kljúfa*(*x*)

- Gefið splay tré og lykilgildi *x*
- Framkvæma *Finna*(*x*) á tréð
  - Þá verður *x* rótin í trénu
- Láta hægra hluttré *x* verða sjálfstætt tré
  - eða láta vinstra hluttré *x* verða sjálfstætt tré



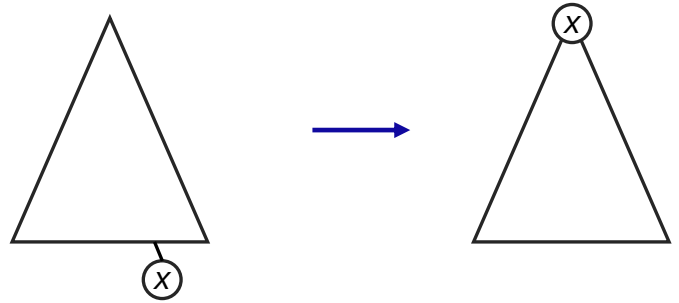


eða

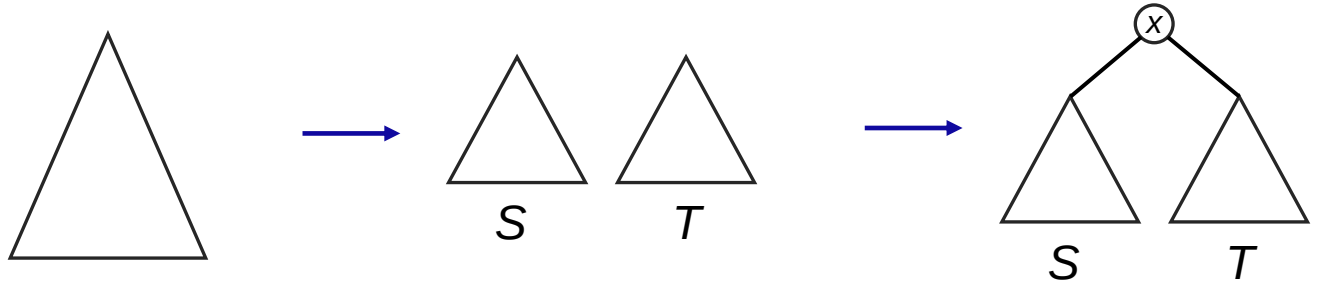


Aðgerðin *Innsetning*(*x*)

- Leita að lyklinum *x* í trénu eins og venjulega
- Ef gildið finnst ekki þá setja *x* inn sem lauf og gera splay(*x*)
  - Þá verður *x* rót trésins



- Önnur útgáfa:
  - Skipta trénu um *x* (gera *Kljúfa*(*x*)) í trén *S* og *T*
  - Setja hnút með *x* sem rót og *S* og *T* hluttré hans

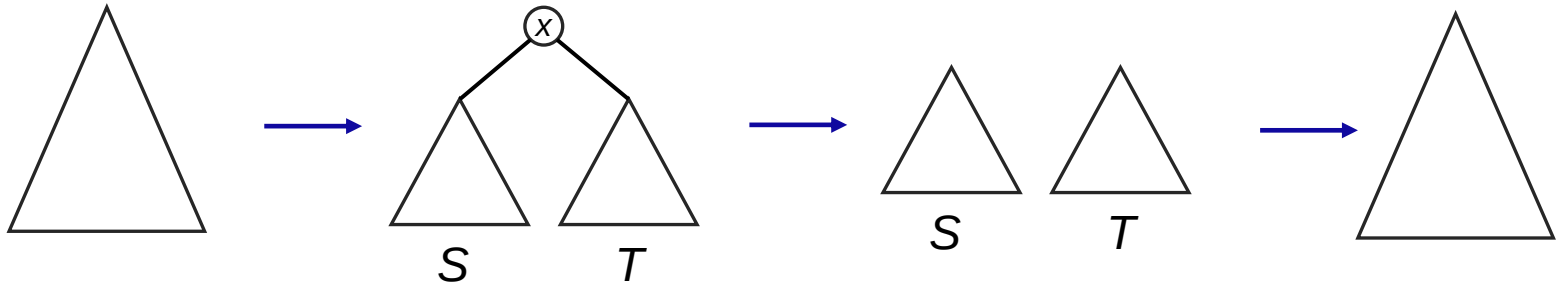


Aðgerðin *Eyðing*(*x*)

- Getum gert það á svipaðan hátt og venjulega í tvíleitartrjám:
  - Ef x hefur tvö börn þá víxla gildi við næsta gildi á eftir (*successor*) eða næsta gildi á undan (*predecessor*) í trénu
  - Eyða svo *x* út með því að tengja framhjá (ef eitt barn) eða henda hnúti (ef lauf)
- Framkvæma svo splay-aðgerð á foreldri eydda hnútsins

Viðbót við venjulegt tvíleitartré!

- Önnur útgáfa:
  - Framkvæma *Finna*(*x*), þá er *x* rótin
  - Henda hnúti *x*
  - Sameina hluttrén tvö



Jafnaðargreining

Skilgreinum:

size(*v*) → *stærð*(*v*): fjöldi hnúta í hluttré *v*

rank(*v*) → *stig*(*v*) = $\log_2$(*stærð*(*v*))

Stöðuorka trés er

$$\Phi = \sum_v stig(v) = \sum_v \log_2 stærð(v)$$

Með því að skoða öll möguleg tilvik á snúningum er hægt að sýna að

Setning: Jafnaðartími hvers einstaks splay-snúnings (*zig*, *zig-zig*, *zig-zag*) á hnúti *v* er í mesta lagi 1 + 3*stig'*(*v*) − 3*stig*(*v*)

stig(*v*) er stig *v* fyrir aðgerð
stig'(*v*) er stig *v* eftir aðgerð

Jafnaðarkostnaður við að framkvæma splay(*v*) er því 1 + 3*stig'*(*v*) − 3*stig*(*v*)

stig'(*v*) er stig *v* eftir alla splay aðgerðina

En nú er *v* orðin rót trésins, svo *stærð*(*v*) = *n* og *stig'*(v) = $\log_2$(*n*)

Þetta þýðir að jafnaðartími splay-aðgerðarinnar er *O*(log *n*)

Allar aðrar aðgerðir á splay tré nota splay-aðgerðina, svo þær hafa líka jafnaðartímann *O*(log *n*)

Aðrar útgáfur

- Framkvæma splay ofanfrá (*top-down*)
  - Þarf þá aðeins eina umferð
  - Losnum við endurkvæmni (eða foreldrabenda)

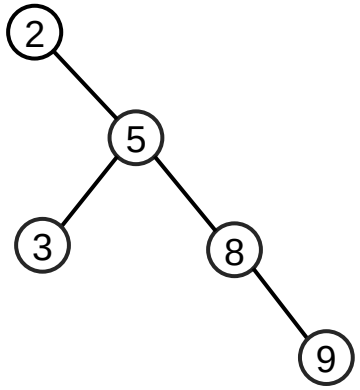
- Einföldun:
  - Oft dýrt að gera fullt splay
  - Fara bara hálfa leið
  - eða fara alla leið, en bara ef vegalengdin er meira en einhver stiki

- Slembin splay tré:
  - Við leitun þá framkvæmum við aðeins splay með líkunum p ($0 \leq p \leq 1$)
  - Getur verið dýrt að gera splay aðgerð við hverja einustu leit
  - Getum nú stýrt því hversu sjálfstillandi við viljum að tréð sé

Fyrirlestraæfingar

1. Nefnið tvö notkunardæmi þar sem sjálfstillandi gagnagrind hentar vel

2. Setjið lykilinn 7 í splay tréð hér til hliðar



3. Hvers vegna framkvæmum við *splay* á foreldri *x* þegar *x* finnst ekki í trénu? Reynið að hugsa upp einhverja góða ástæðu!