

# Dynaaminen yhtenäisyys

June 19, 2017

Tehdään kyselyitä verkkoon

- Lisää kaari verkkoon
- Poista kaari verkosta
- Laske verkon yhdistyneiden komponenttien määrä
- Ovatko solmut  $a$  ja  $b$  samassa yhdistyneessä komponentissa

Union-find rakenteella voidaan tehdä kyselyt

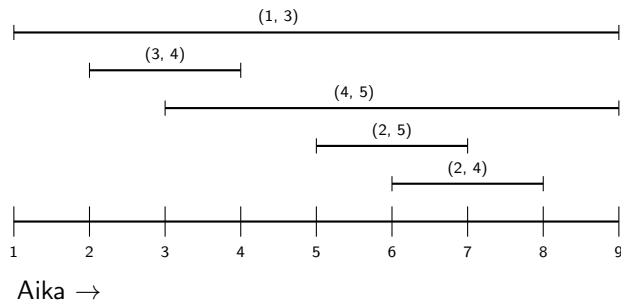
- Lisää kaari verkkoon  $O(\log n)$  ajassa
- Poista kaari verkosta
- Laske verkon yhdistyneiden komponenttien määrä  $O(1)$  ajassa
- Ovatko solmut  $a$  ja  $b$  samassa yhdistyneessä komponentissa  $O(\log n)$  ajassa

Union-find rakenteella voidaan tehdä kyselyt

- Lisää kaari verkkoon  $O(\log n)$  ajassa
- Poista kaari verkosta
- Laske verkon yhdistyneiden komponenttien määrä  $O(1)$  ajassa
- Ovatko solmut  $a$  ja  $b$  samassa yhdistyneessä komponentissa  $O(\log n)$  ajassa
- Peruuta aikaisempaan tilaan  $O(1)$  ajassa

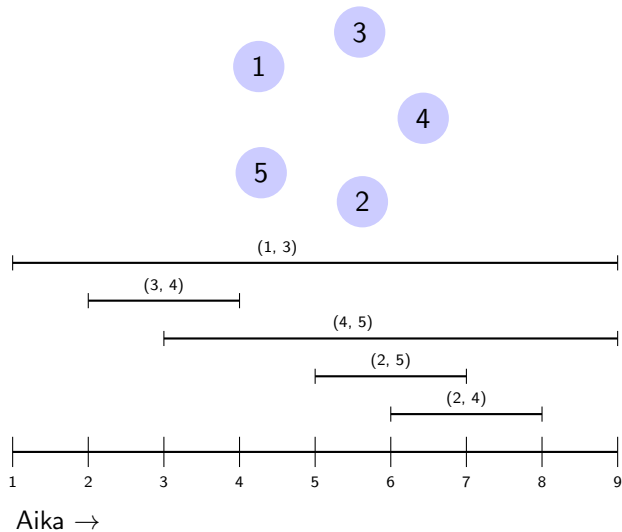
- Jokaiseen kaareen liitetään alkamisaika ja loppumisaika

```
Lisää (1, 3)
Lisää (3, 4)
Lisää (4, 5)
Poista (3, 4)
Lisää (2, 5)
Lisää (2, 4)
Poista (2, 5)
Poista (2, 4)
```

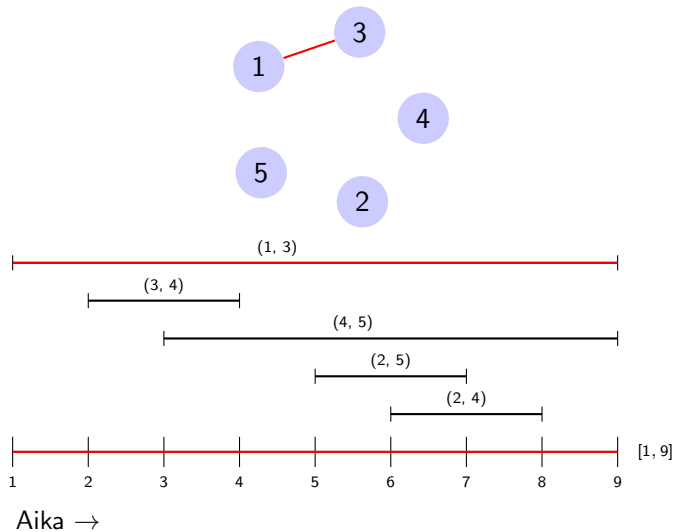


- Nyt voidaan tehdä divide and conquer tyyppinen algoritmi joka selvittää jokaiselle ajanhetkelle minkälainen verkko silloin on

# Offline algoritmi

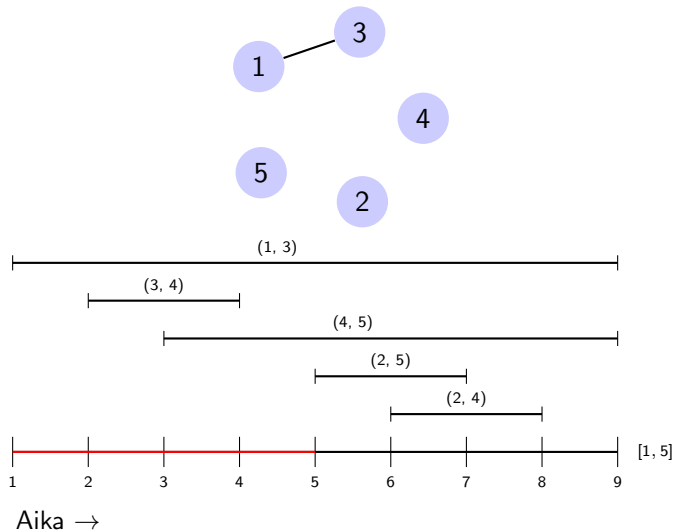


# Offline algoritmi

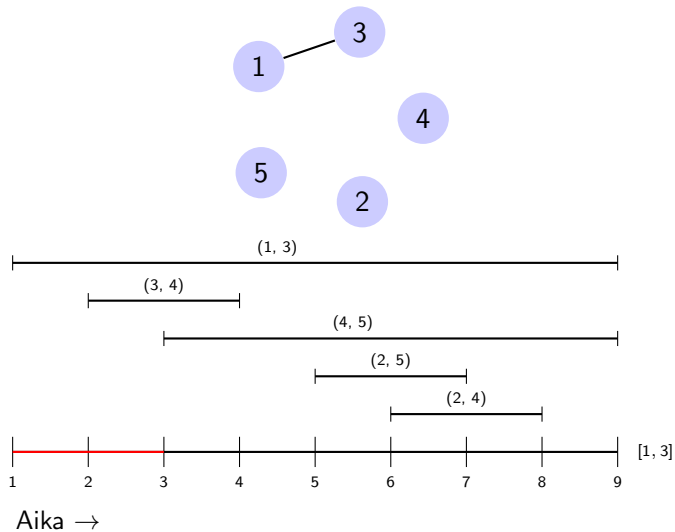




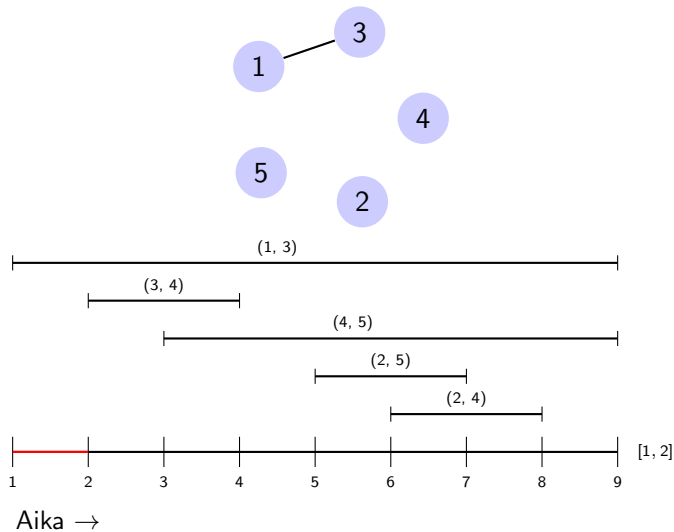
# Offline algoritmi



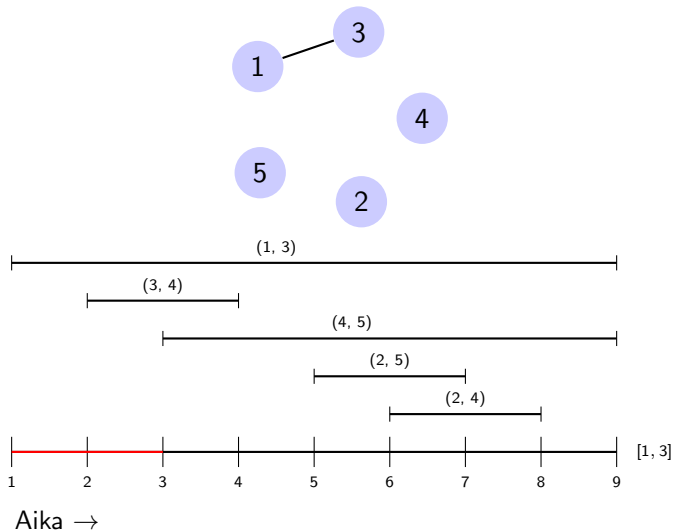
# Offline algoritmi



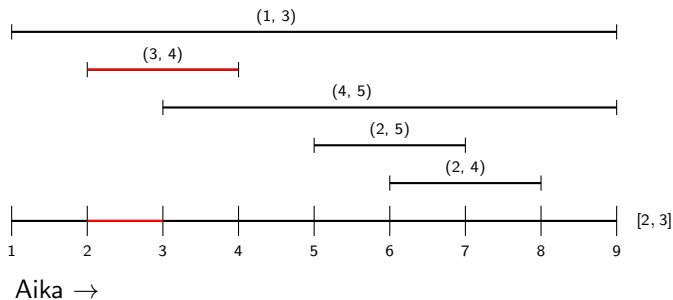
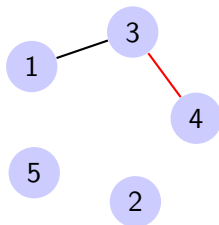
# Offline algoritmi



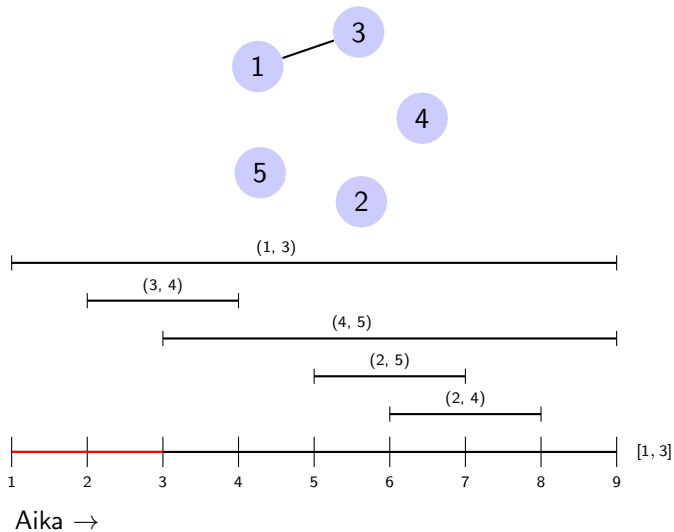
# Offline algoritmi



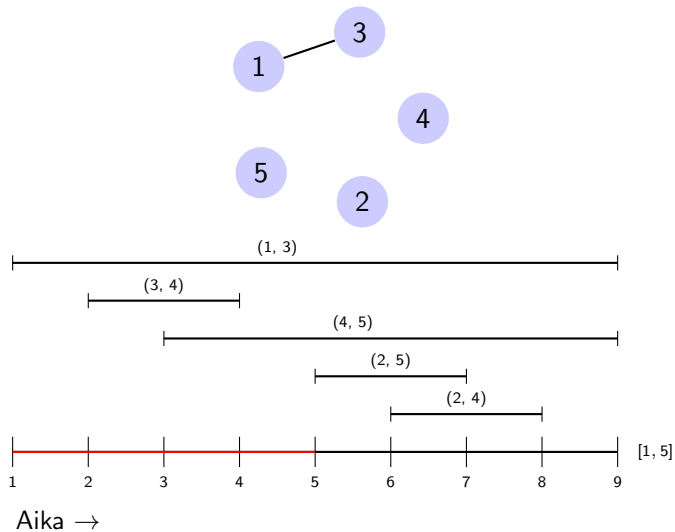
# Offline algoritmi



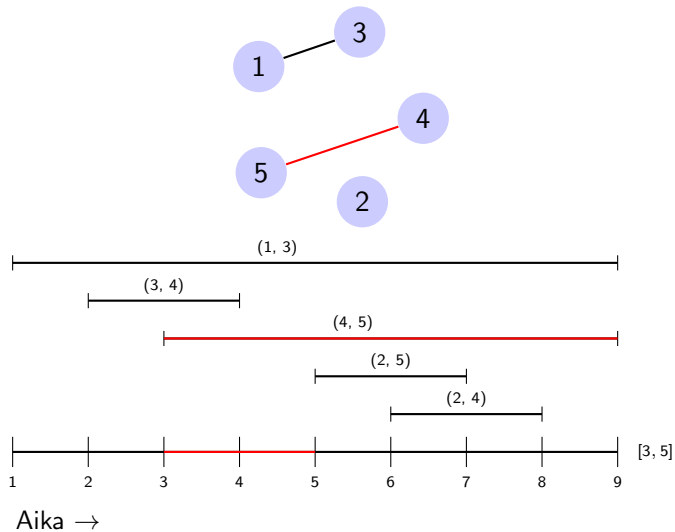
# Offline algoritmi



# Offline algoritmi

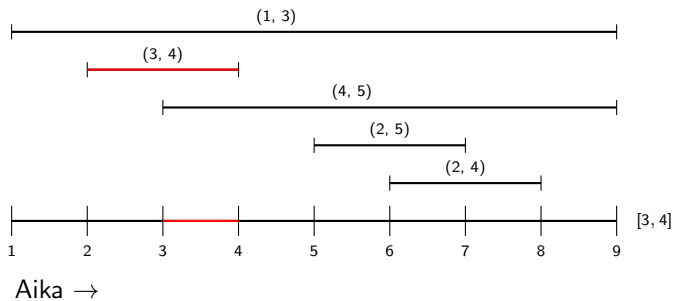
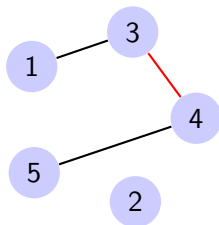


# Offline algoritmi

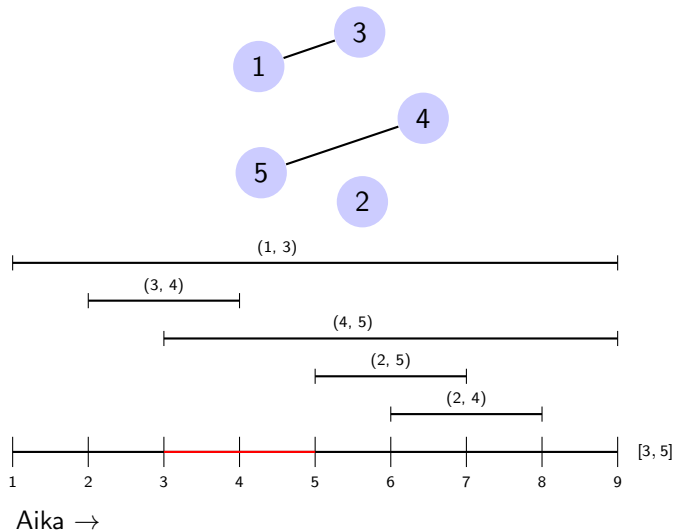




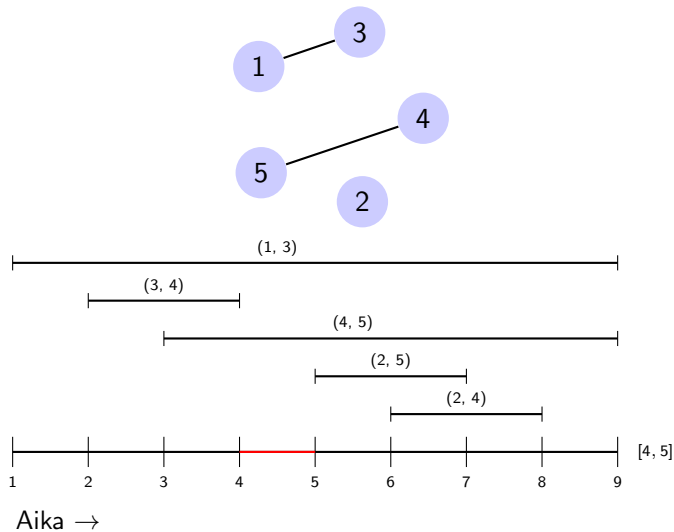
# Offline algoritmi



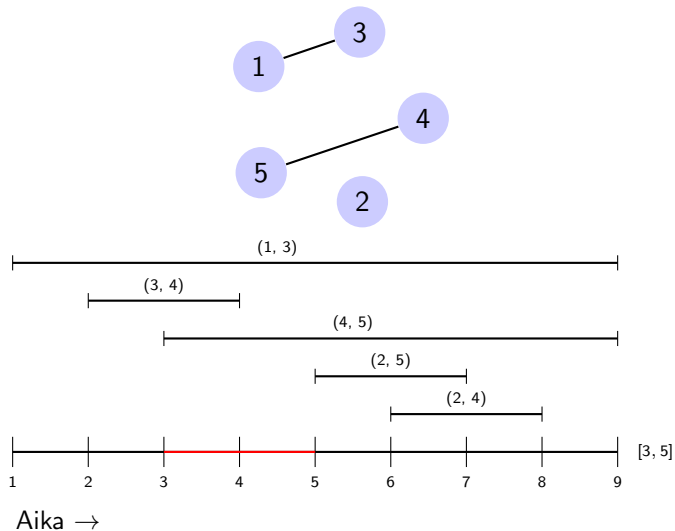
# Offline algoritmi



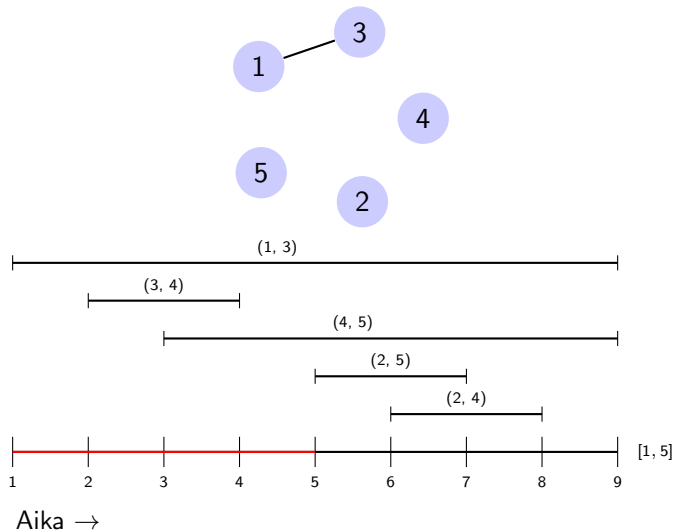
# Offline algoritmi



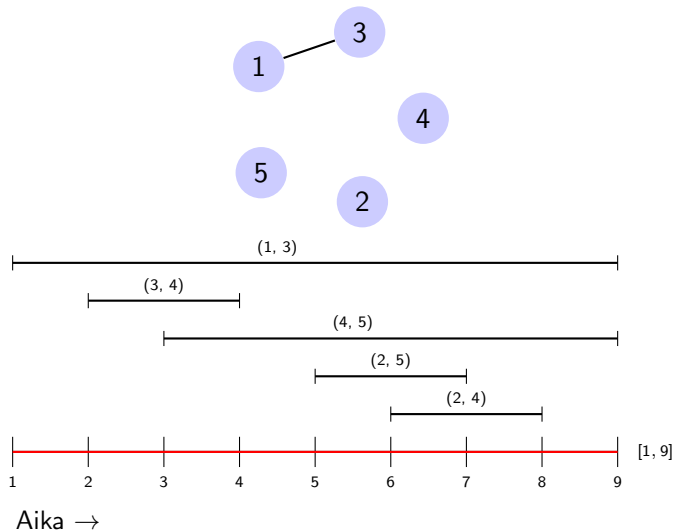
# Offline algoritmi



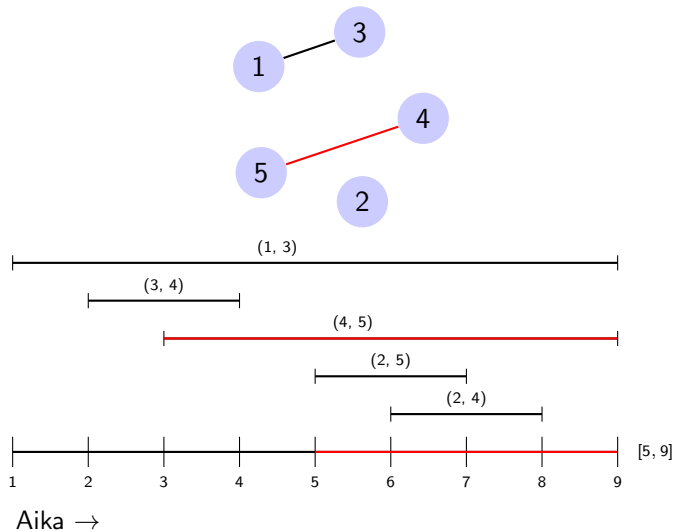
# Offline algoritmi



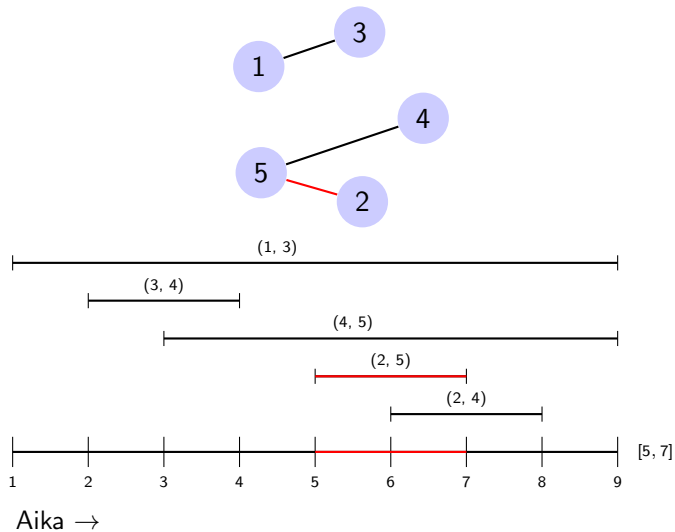
# Offline algoritmi



# Offline algoritmi

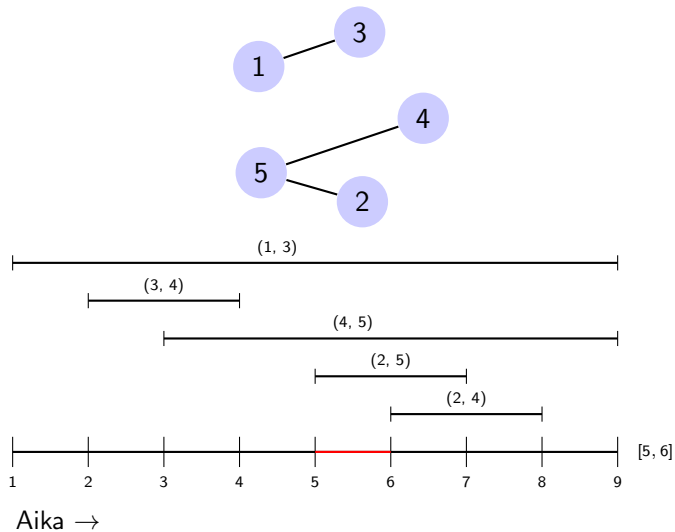


# Offline algoritmi

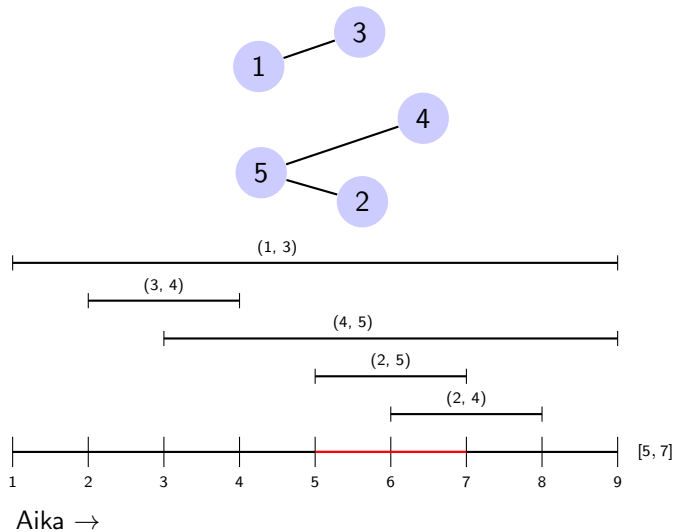




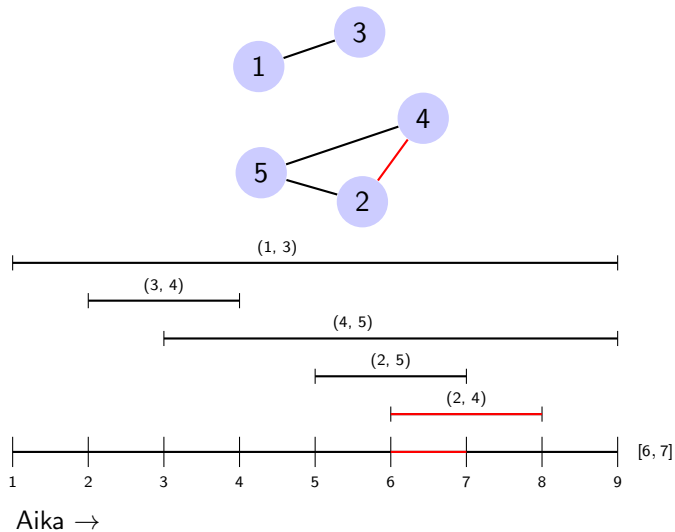
# Offline algoritmi



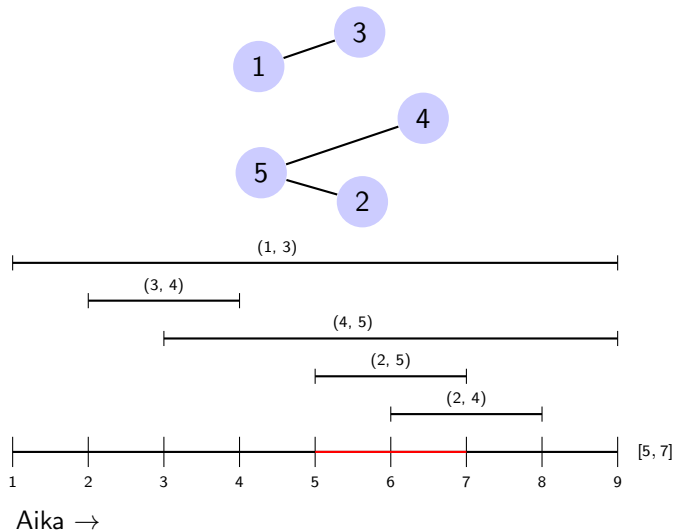
# Offline algoritmi



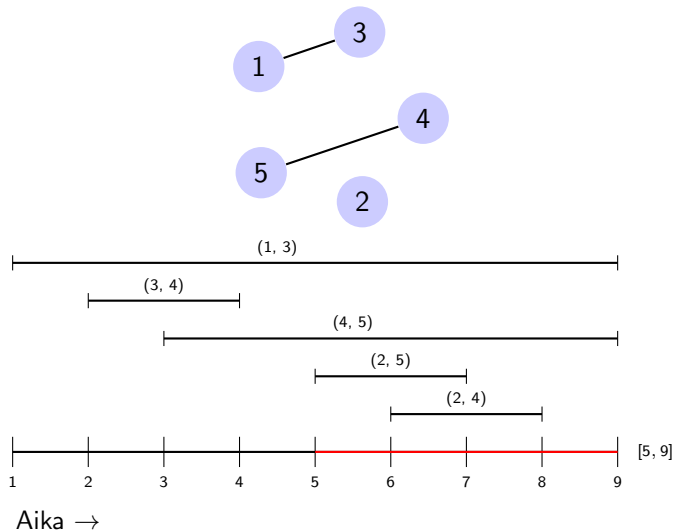
# Offline algoritmi



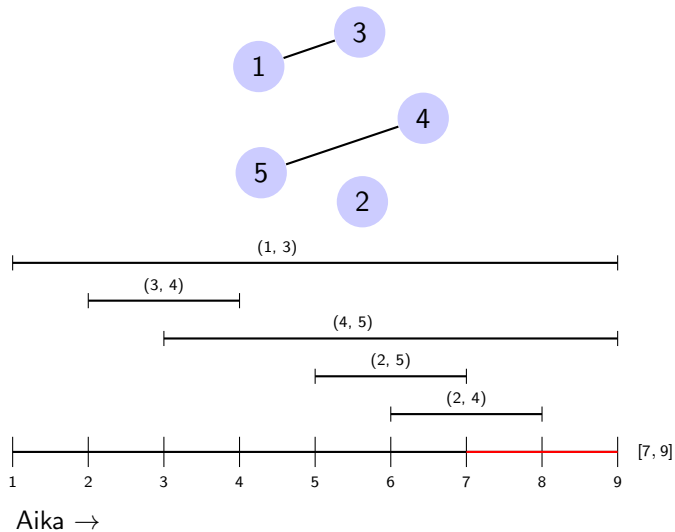
# Offline algoritmi



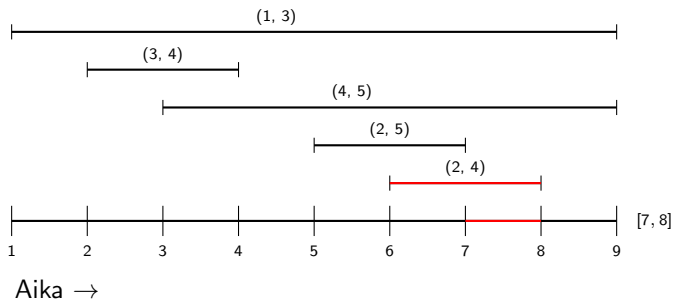
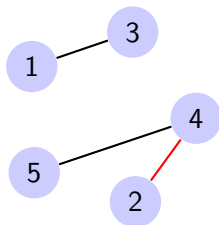
# Offline algoritmi



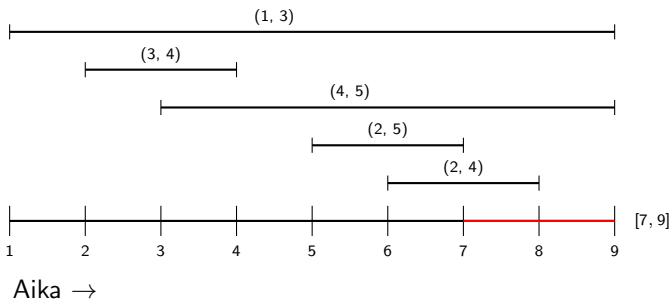
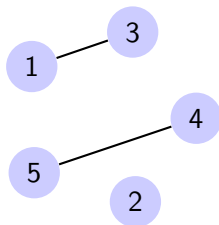
# Offline algoritmi



# Offline algoritmi

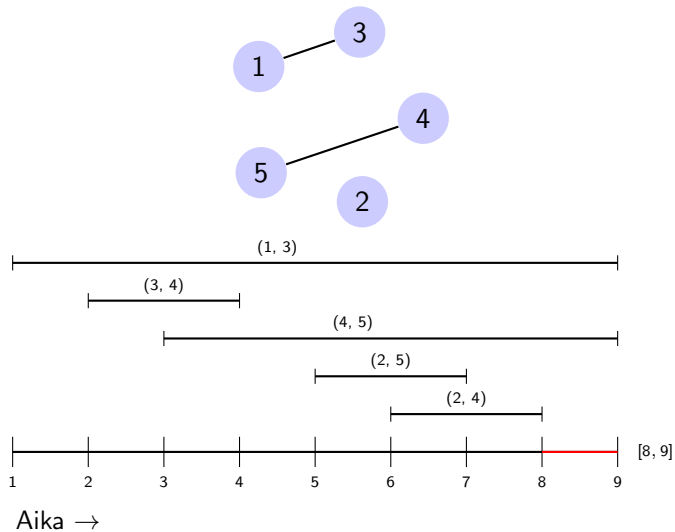


# Offline algoritmi

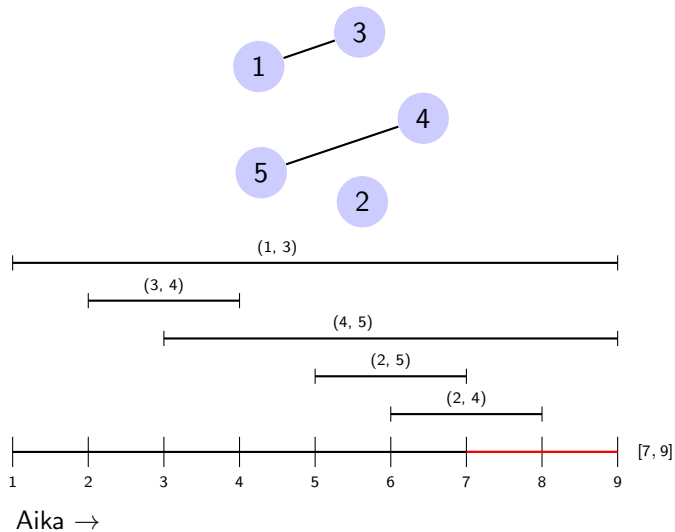




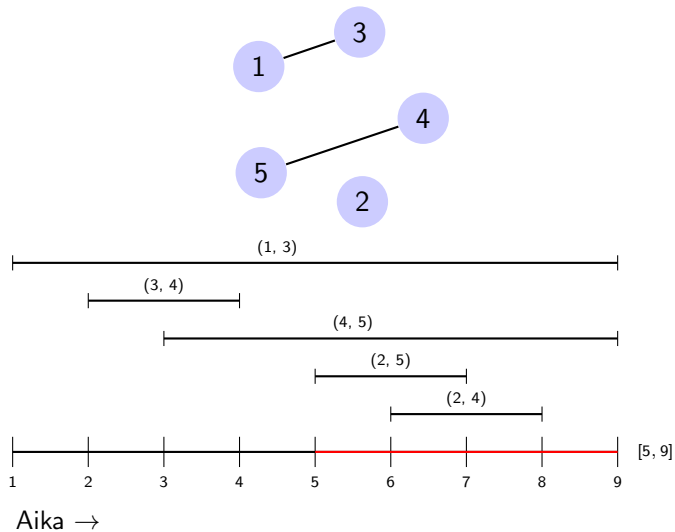
# Offline algoritmi



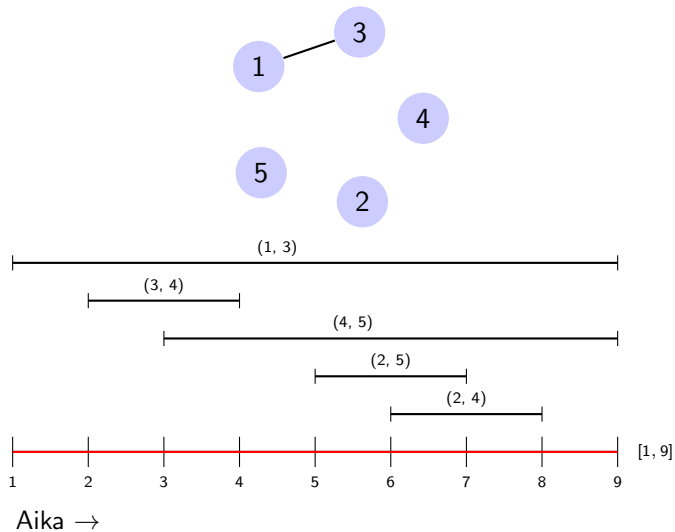
# Offline algoritmi



# Offline algoritmi



# Offline algoritmi



- Jokainen kaari lisätään maksimissaan  $O(\log n)$  kertaa
- Aikavaativuus on siis  $O(n \log^2 n)$
- On olemassa myös toteutus joka ei käytä union-find rakennetta jolloin saadaan  $O(n \log n)$  aikavaativuus

- Jos verkko on metsä, link/cut tree-tietorakenteella voi vastata dynaamisen yhtäisyyden kyselyihin  $O(\log n)$  ajassa
- Paras tunnettu tietorakenne yleiseen dynaamiseen yhtenäisyyteen vastaa kaarien lisäys/poisto kyselyihin  $O(\log^2 n / \log \log n)$  ajassa ja yhtenäisyys kyselyihin  $O(\log n / \log \log n)$  ajassa

- Tämän offline-algoritmin idea ei rajoitu pelkästään verkkojen yhtenäisyyden käsittelyyn
- Samaa ideaa voidaan soveltaa mihin tahansa prosessiin jossa asiat ovat aktiivisia jollain aikavälillä ja voidaan implementoida tietorakenne joka tulee lisäämistä ja undoamista