YoYo leader election i minimum finding algoritam

Bežične mreže osjetila

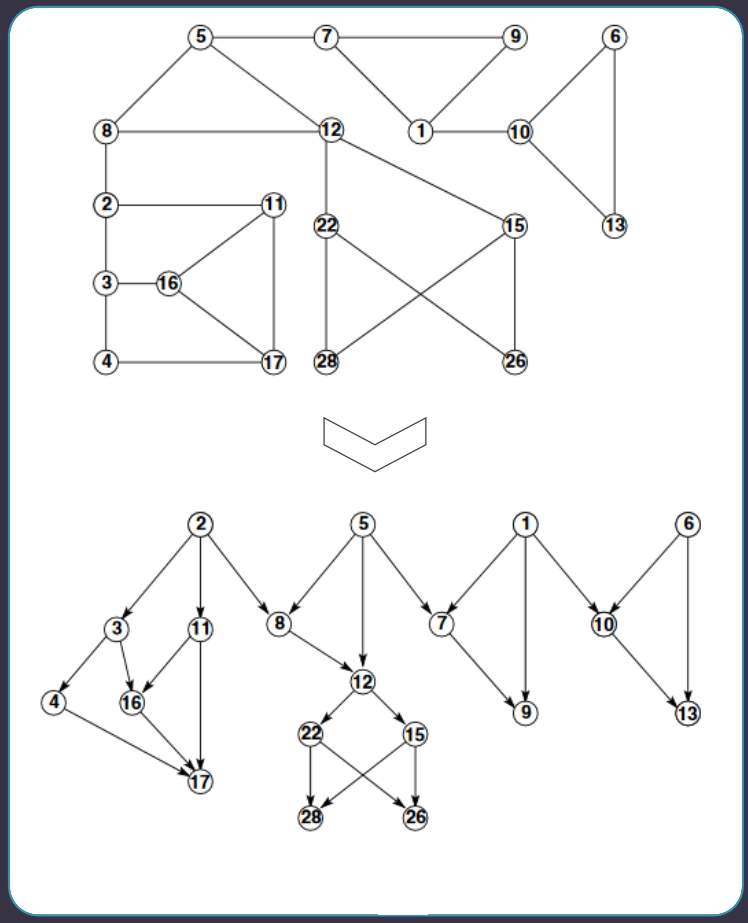
Matko Burul

Uvod

- Cilj leader election algoritma
  - U sustavu entiteta pronaći jedan entitet koji će služiti kao vođa (*Leader*), a ostali entiteti (*Follower*) moraju biti svjesni tog odabira
- Set restrikcija:
  - Standardni set restrikcija R: *Bidirectional Links, Connectivity, Total Reliability*
  - Ove restrikcije nisu dovoljne, te se uvodi dodatna : *Initial Distinct Values* (ID)
  - Standardni set restrikcija za *Election* problem: IR = R ∪ {ID}
- *Minimum Finding* problem se također rješava zbog toga što ovaj algoritam primjenjuje strategiju *Elect Minimum:*
  - Pronađi minimalnu vrijednost
  - Vođa je onaj koji posjeduje tu vrijednost

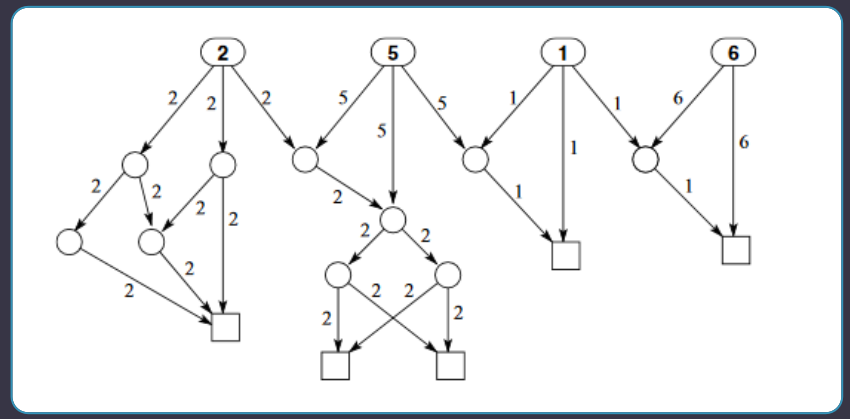
Pred-procesiranje

- U ovoj fazi svaki čvor razmjeni svoj ID sa svim svojim susjedima, te logički orijentira svaki od bridova tako da budu u smjeru onog čvora s većim ID
- Cilj ove faze je stvoriti Usmjereni Aciklički Graf i svaki čvor u ovom grafu ima jedno od 3 stanja:
  - Izvor *(Source)* je čvor koji ima samo susjede s većim ID, taj čvor je lokalni minimum
  - Ponor *(Sink)* je čvor koji ima samo susjede s manjim ID, taj čvor je lokalni maksimum
  - Unutarnji *(Internal)* je čvor koji nije ni Izvor, a ni Ponor
- Nakon dovršenja ove faze kreću iteracije Yo- i -Yo faza



Yo-

- Ovu fazu započinju Izvori tako da svim svojim izlaznim susjedima pošalju svoj ID
- Unutarnji čvorovi čekaju da prime ID od svih svojih ulaznih susjeda, izračunavaju minimalni ID i prosljeđuju ga svim svojim izlaznim susjedima
- Ponori također čekaju da prime ID od svih svojih ulaznih susjeda, te izračunavaju minimalni ID i započinju sljedeću fazu: -Yo fazu

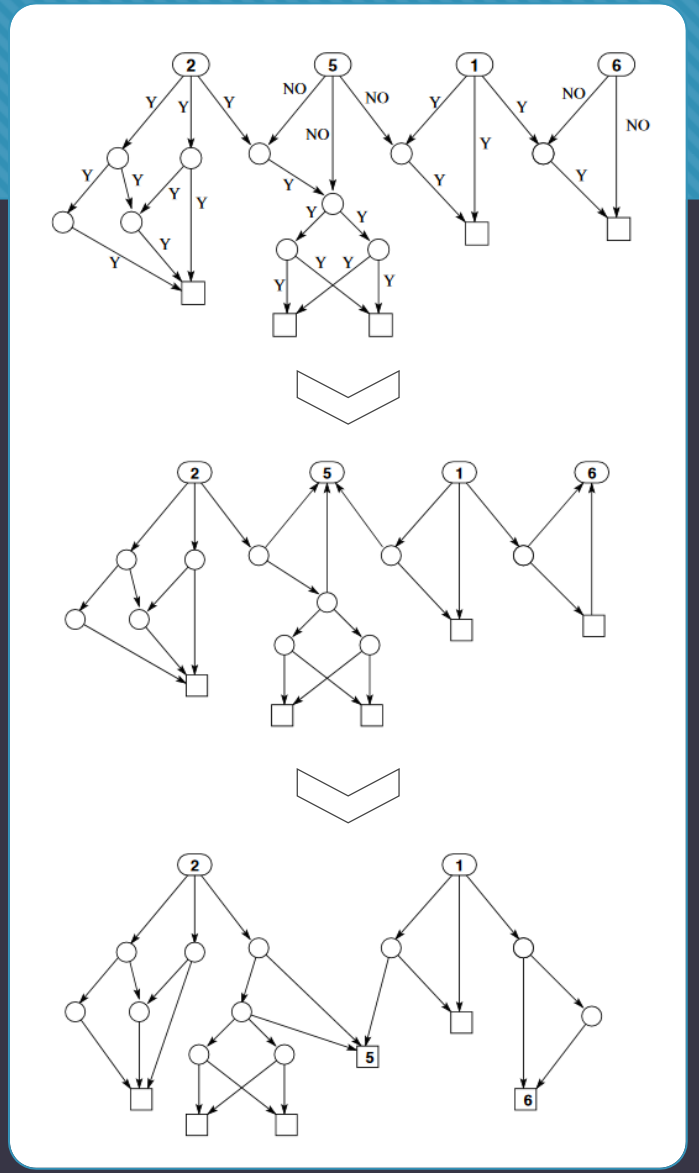


-Yo

- Ovu fazu započinju Ponori tako da pošalju DA svim ulaznim susjedima od kojih je primljena najmanji ID, a NE svim ostalim ulaznim susjedima
- Unutarnji čvorovi čekaju da prime odgovore za sve odaslane ID, te postupaju po pravilu:
  - Ako su svi odgovori DA, šalju DA svim čvorovima od kojih je primljen najmanji ID, a čvorovima od kojih su primljeni drugi ID šalju NE
  - Ako je primljen barem jedan odgovor NE, šalju NE svim čvorovima od kojih je primljen ID, neovisno o ID
- Izvori čekaju da prime odgovore za sve odaslane ID i ukoliko je primljen barem jedan odgovor NE prestaju biti kandidati, a ako su svi odgovori DA nastavljaju u sljedeću iteraciju
- Izvori se eliminiraju pomoću modifikacije smjera bridova koji nose NE odgovore

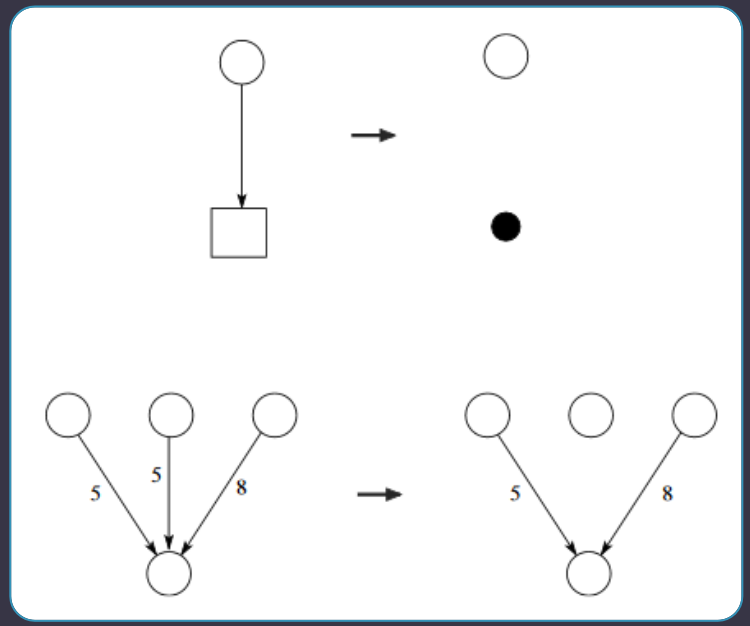
Modifikacija smjera bridova

- Kada čvor x pošalje odgovor NE ulaznom susjedu zaokrenuti će logički smjer brida koji ih spaja, time će taj susjed postati izlazni susjed čvoru x
- Kada čvor y primi odgovor NE od izlaznog susjeda zaokrenuti će logički smjer brida koji ih spaja, time će taj susjed postati ulazni susjed čvoru y
- Kao rezultat ovoga bilo koji Izvor koji primi NE prestati će bit Izvor i može postati Ponor ili Unutarnji čvor
- Također neki Unutarnji čvorovi mogu postati Ponori i obrnuto, no niti jedan Unutarnji čvor ili Ponor neće postati Izvor



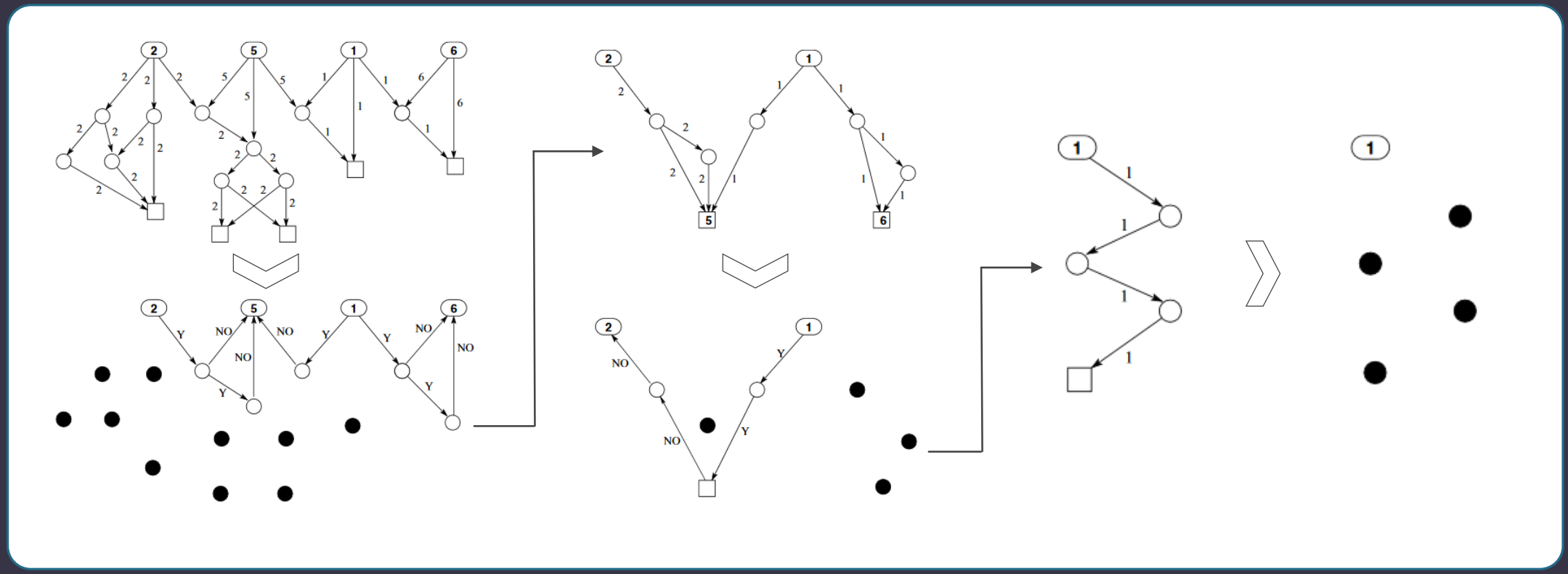
Pruning

- U svrhu smanjenja broja poruka i detekcije završetka algoritma uvodimo dodatni mehanizam *Pruning* koji uklanja čvorove koji nemaju utjecaj na rezulatat iteracije
- Ovaj mehanizam se sastoji od 2 pravila:
  - Ako Ponor ima samo 1 ulaznog susjeda, onda će uvijek odgovoriti sa DA, te će biti uklonjen
  - Ako u Yo- fazi Unutarnji čvor ili Ponor prime vrijednost koja je minimalni ID u toj iteraciji s više od 1 ulaznog čvora, svi bridovi koji su prenijeli tu vrijednost osim 1 biti će uklonjeni
- Pravila se provode slanjem *Prune* zahtjeva u -Yo fazi, tj. Zajedno s DA ili NE odgovorom
- Ukoliko zbog djelovanja 2. pravila čvor postane kandidat za 1. pravilo ono se i primjenjuje



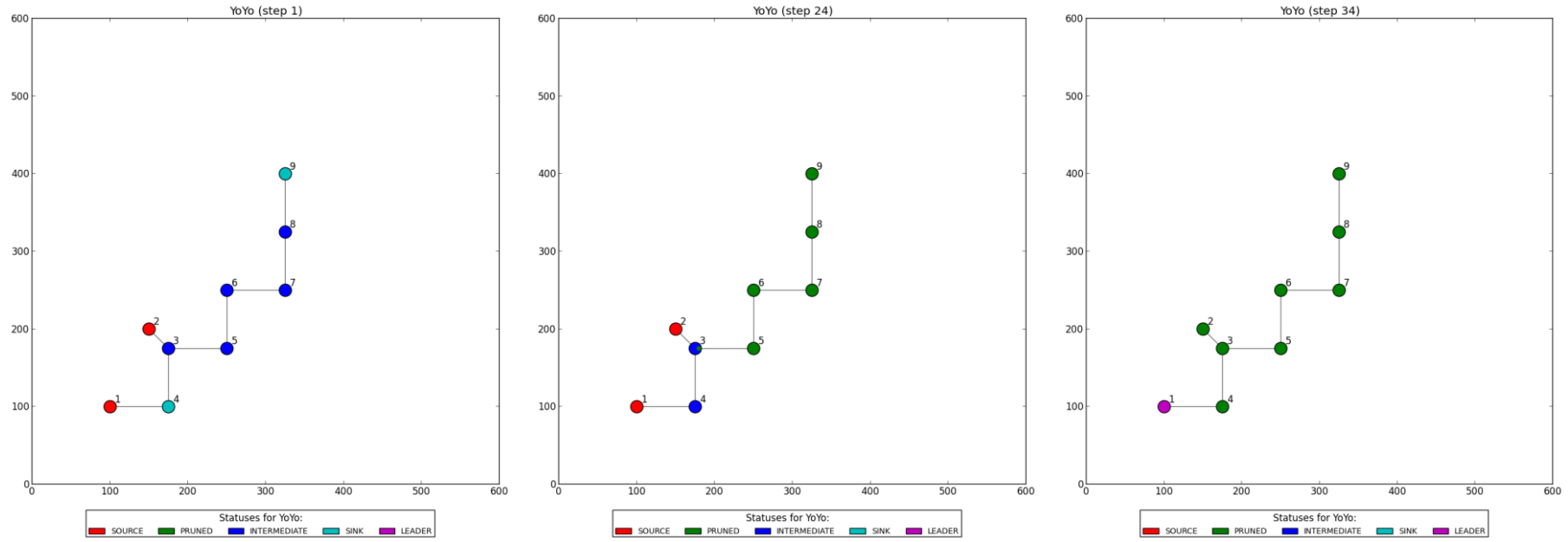
Završetak algoritma

- Kada Izvor ukloni sve svoje izlazne susjede zbog pravila *Prune* mehanizma, on će postati vođa i algoritam završava



Specifičnosti implementacije

- Zbog mogućnosti „miješanja” faza čvorovi mogu primiti ID kada očekuju odgovor, te je dodano polje u memoriji koje sprema ID kada nisu očekivani, te ih nakon zaprimanja potrebnih odgovora čvor prebacuje u polje zaprimljenih ID



1. info
   - id: 3
   - orientation: 5.50574683144
   - position: [175 175]
   - status: INTERMEDIATE
2. communication
   - inbox
     - 1. Message
       - 1 header: response
       - 2 source: <Node id=5>
       - 3 destination: <Node id=3>
       - 4 data: (True, 'prune')
   - outbox
   - range: 100
3. memory
   - InNeighbors: [<Node id=2>]
   - Neighbors: [<Node id=5>, <Node id=4>, <Node id=2>]
   - OutNeighbors: [<Node id=5>, <Node id=4>]
   - id: 3
   - received_ids
     - 2: [<Node id=2>]
   - received_ids_while_waiting
     - 1: [<Node id=4>]
   - received_responses
     - False: [<Node id=4>]
   - requested_pruning: []
   - sent_ids: 2
4. sensors
   - NeighborsSensor: ('', 0)

1. info
   - id: 3
   - orientation: 5.50574683144
   - position: [175 175]
   - status: INTERMEDIATE
2. communication
   - inbox
   - outbox
     - 1. Message
       - 1 header: id
       - 2 source: <Node id=3>
       - 3 destination: <Node id=2>
       - 4 data: 1
     - 2. Message
       - 1 header: response
       - 2 source: <Node id=3>
       - 3 destination: <Node id=2>
       - 4 data: (False,)
   - range: 100
3. memory
   - InNeighbors: [<Node id=4>]
   - Neighbors: [<Node id=5>, <Node id=4>, <Node id=2>]
   - OutNeighbors: [<Node id=2>]
   - id: 3
   - received_ids
     - 1: [<Node id=4>]
   - received_ids_while_waiting
   - received_responses
   - requested_pruning: []
   - sent_ids: 1
4. sensors
   - NeighborsSensor: ('', 0)