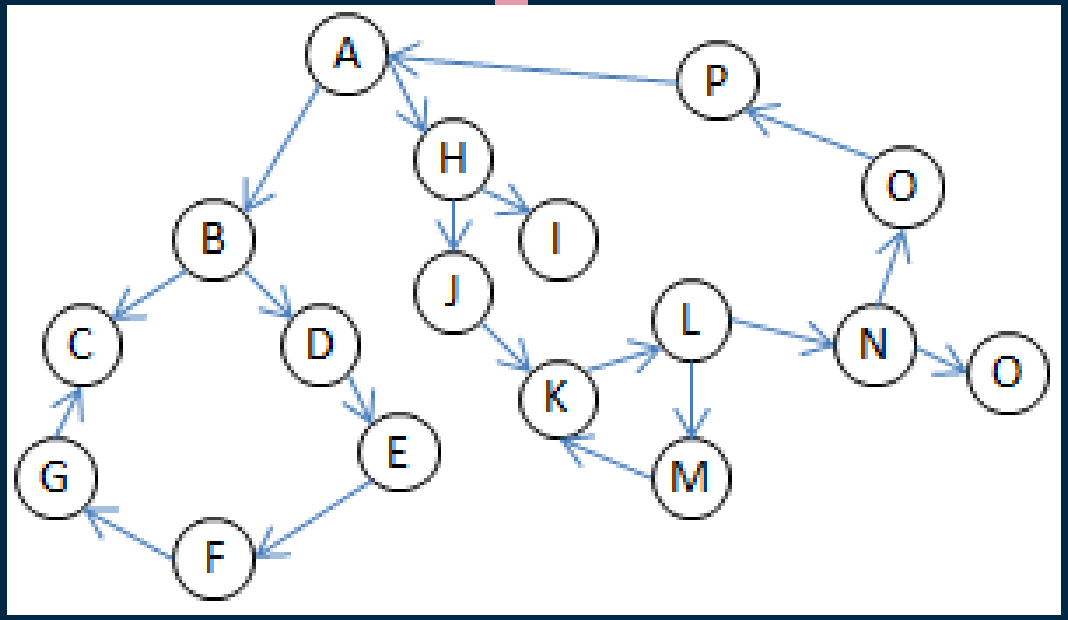
Apache Giraph i distribuirana obrada grafova

Šta je graf?

- Skup čvorova (V)
- Skup grana (E)
- Grane povezuju parove čvorova
- Može biti usmeren ili neusmeren

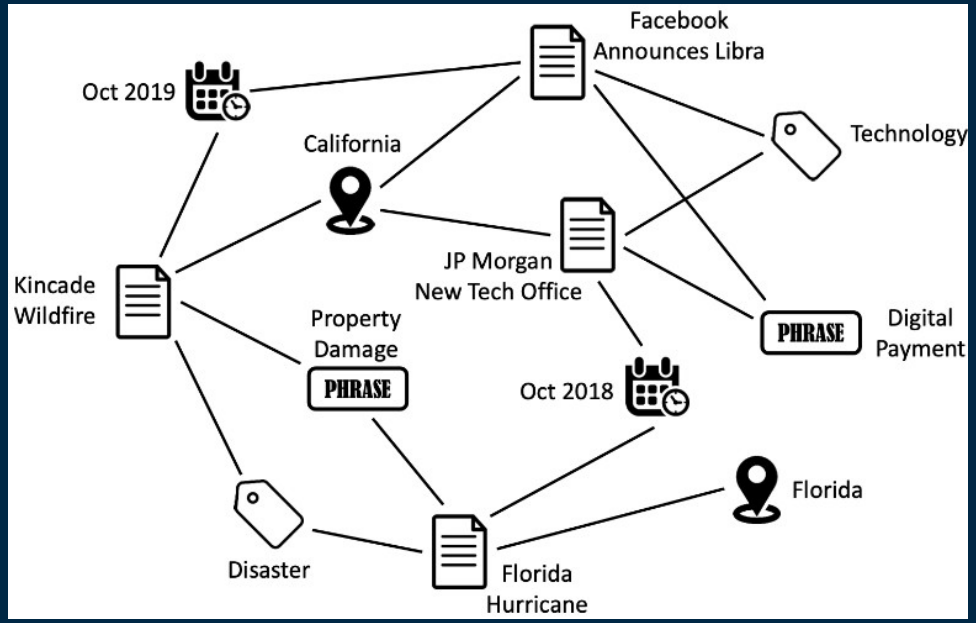


Eksplozija graf podataka

- Ogroman rast čvorova i veza
- Dinamika i promenljivost podataka – novi čvorovi i veze se stalno dodaju
- Primena:
  - Društvene mreže
  - Sistemi za preporuke
  - Navigacioni sistemi i mape
  - Finansijske mreže i transakcije
  - ...

Izazovi pri obradi grafova

1. Nelinearna i nepravilna struktura:
   - Svaki čvor može imati različit broj veza.
   - Čvorovi i veze mogu biti različitih tipova.
   - Grafovi nisu uniformni kao tabele



Izazovi pri obradi grafova

2. Visoka zavisnost među podacima:
   - Algoritmi zahtevaju informacije o susednim čvorovima.
   - Sprečava nezavisnu paralelnu obradu.

3. Memorijska zahtevnost:
   - Veliki grafovi (npr. društvene mreže) ne staju u RAM.
   - Potrebna je distribucija podataka na više računara.

Izazovi pri obradi grafova

4. Iterativna priroda algoritama:
   - Algoritmi zahtevaju više iteracija.
   - Rezultat jedne iteracije utiče na sledeću.

5. Potreba za paralelizacijom:
   - Povezani čvorovi mogu biti na različitim računarima.
   - Komunikacija između računara povećava troškove.
   - Smanjenje međuračunarske komunikacije je izazov.

Izazovi pri obradi grafova

6. Dinamična struktura grafova:
   - Čvorovi i veze se stalno dodaju i menjaju.
   - Tipovi veza i atributi čvorova se stalno dodaju i menjaju.

Rešenje? Specijalizovani sistemi

Apache Giraph:

- Dizajniran za paralelnu i distribuiranu obradu grafova.
- Dizajniran za velike i kompleksne grafove.
- Smanjuje memorijske i komunikacione izazove.

Istorijat

2010. – Google predstavio Pregel model, vertex-centric paradigma

2011. – Yahoo! započinje razvoj Giraph-a kao open-source implementaciju Pregela

2012. – Giraph preuzima Apache Software Foundation, postaje inkubacioni projekat

2015. – Objavljena knjiga „Practical Graph Analytics with Apache Giraph“

2023–2024. – Premještanje projekta u Apache Attic, dostupno, ali bez aktivnog razvoja

Upotreba

Facebook – analiza društvenih mreža sa stotinama miliona korisnika i milijardama veza

Twitter – obrada velikih mreža za preporuke i analizu interakcija

LinkedIn – analiza profesionalnih mreža i detekcija sličnih profila

Porgramski model – Bulk Synchronous Parallel

Bulk Synchronous Parallel (BSP) model:

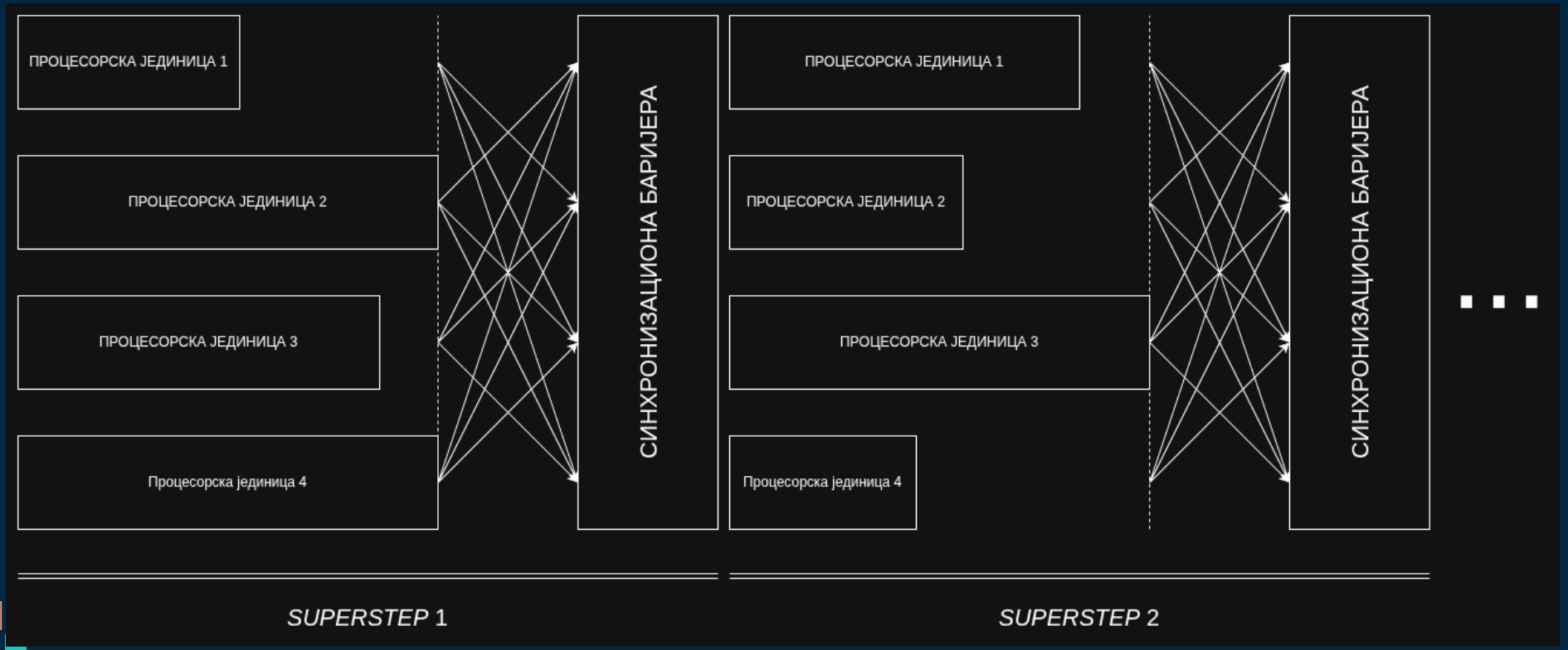
- Osnova za paralelnu i distribuiranu obradu podataka.
- Giraph primenjuje BSP za skalabilnu obradu grafova.

Superstep - jedna iteracija paralelnog algoritma.

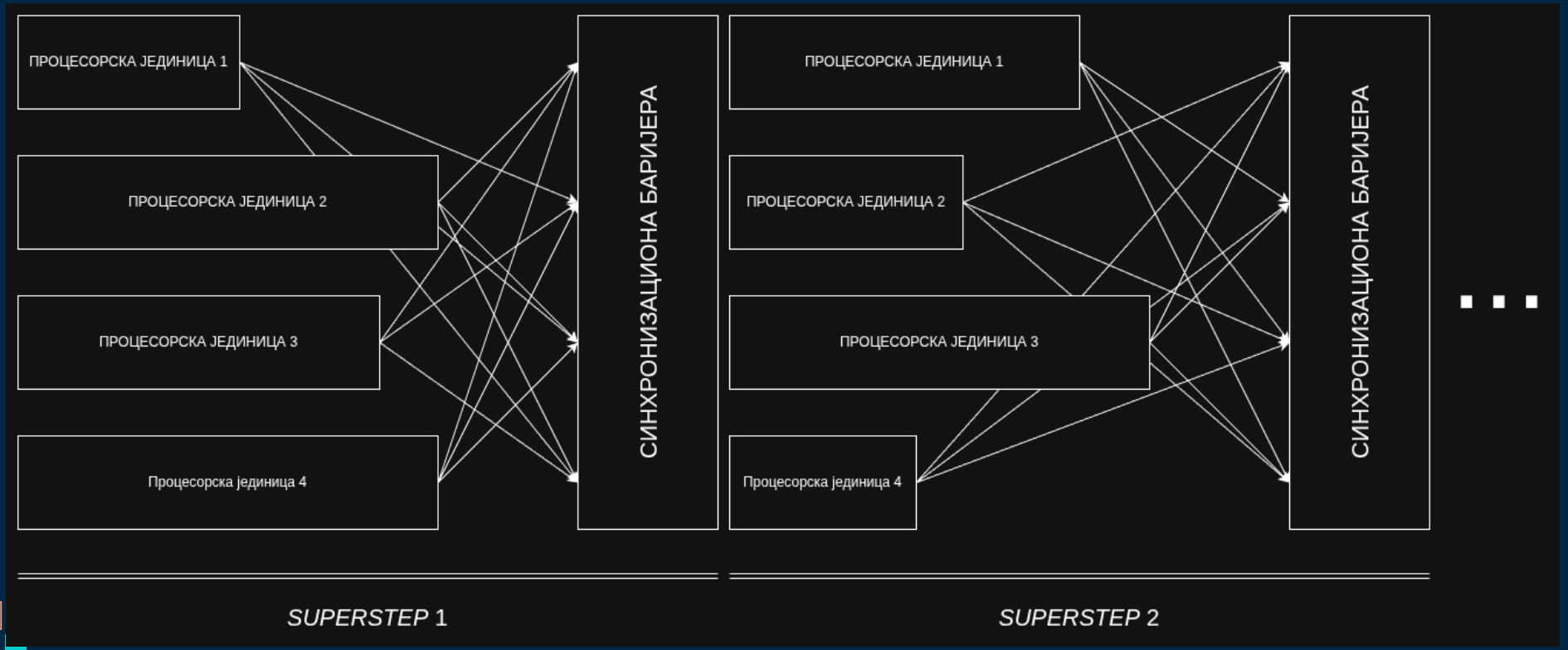
Tri faze superstep-a:

- Lokalna obrada – procesorske jedinice rade nezavisno na svom delu grafova.
- Komunikacija – slanje i primanje poruka između procesorskih jedinica.
- Sinhronizacija – čekanje da sve jedinice završe pre prelaska na sledeći superstep.

Porgramski model – Bulk Synchronous Parallel



Porgramski model – Bulk Synchronous Parallel



Porgramski model – Vertex-Centric

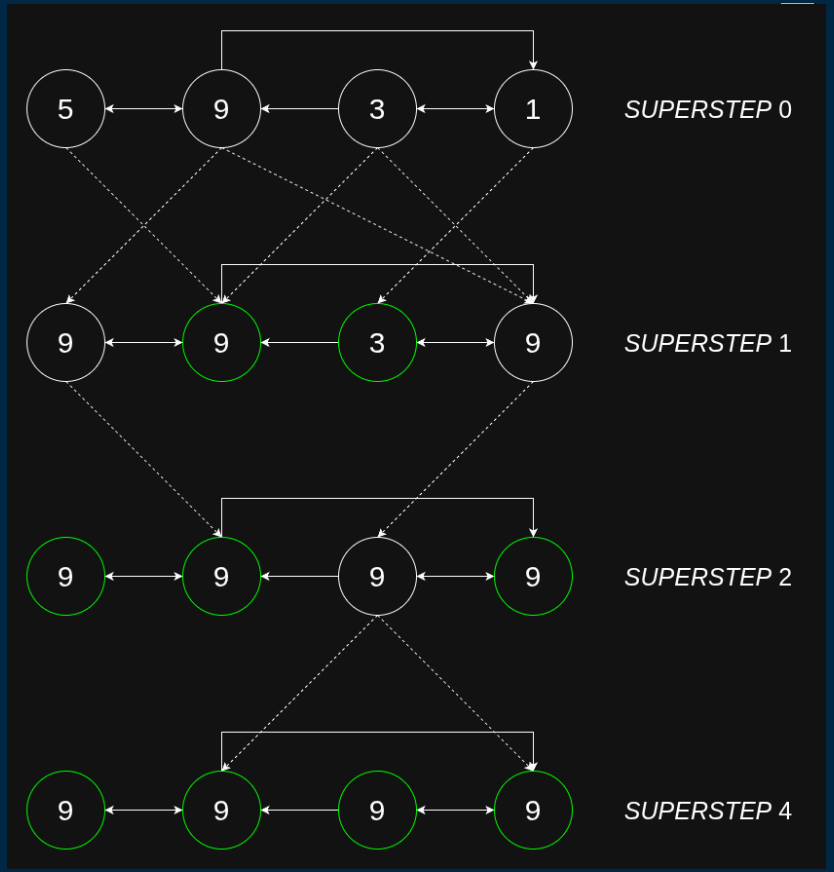
Vertex-Centric model:

- Giraph primenjuje Vertex-Centric model za paralelnu obradu grafova.

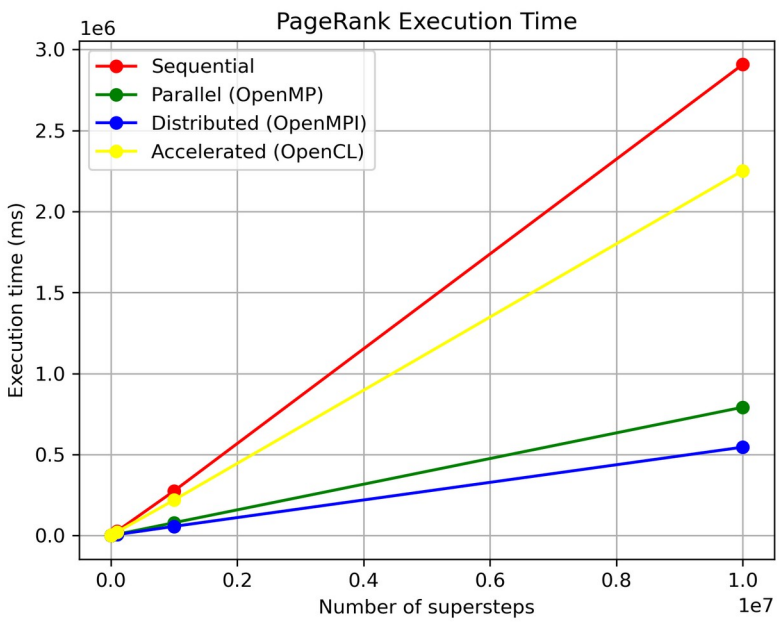
Ključne ideje:

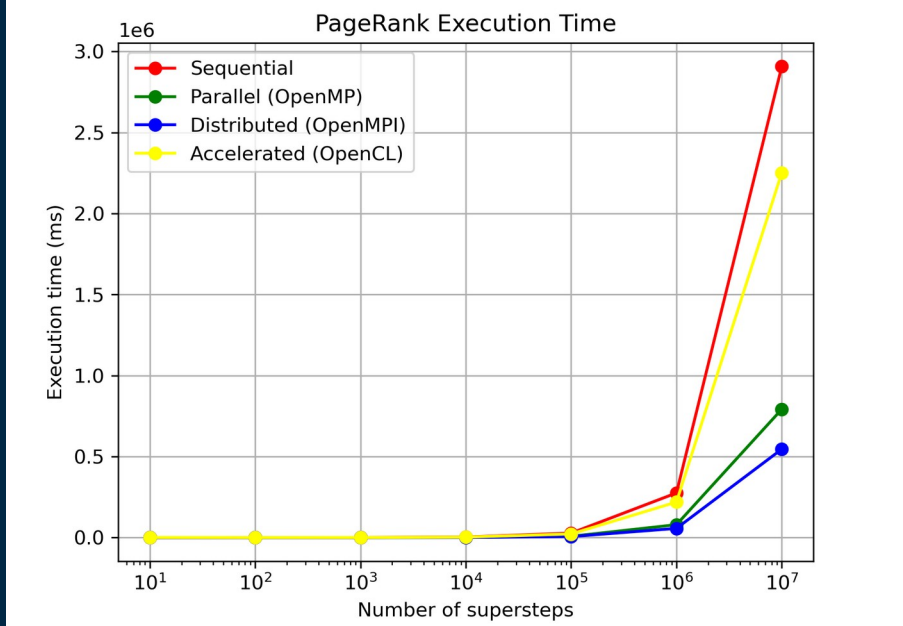
- Svaki čvor (vertex) u grafu obrađuje svoje podatke nezavisno.
- Čvor prima poruke od suseda, ažurira svoje stanje i šalje poruke dalje.
- Čvor se deaktivira ako nema novih poruka ili promena.
- Algoritam traje dok svi čvorovi nisu deaktivirani.
- Primenjuje se u iterativnim superstep-ovima.

Porgramski model – Vertex-Centric

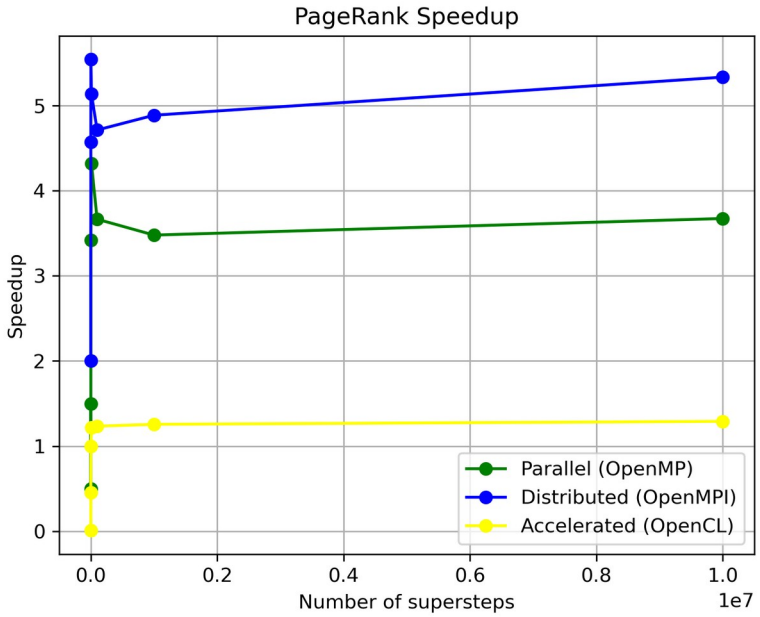


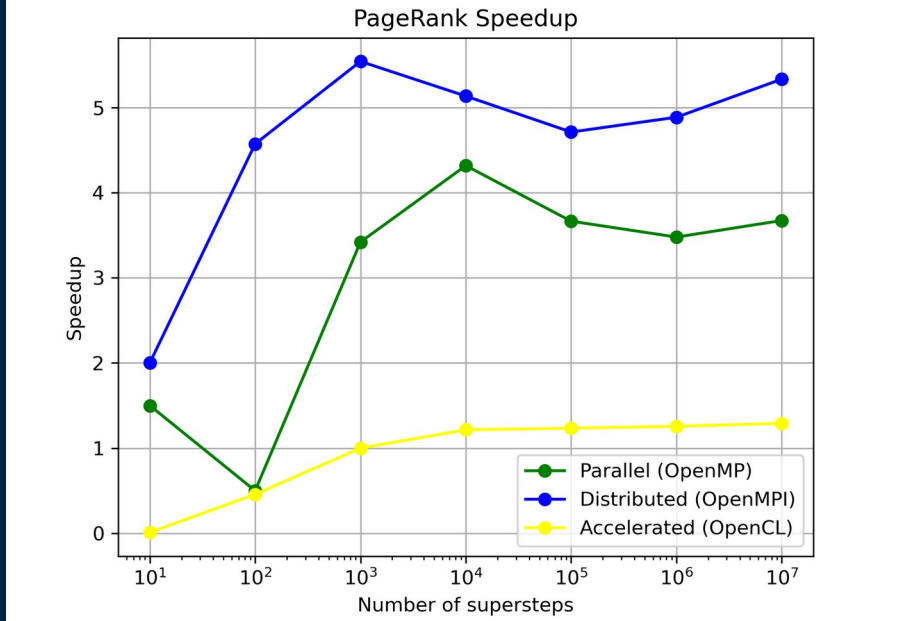
Porgramski model – BSP + Vertex-Centric





Porgramski model – BSP + Vertex-Centric





Arhitektura

Tri osnovna servisa.

Master:

- Upravljanje tokovima superstep-ova
- Dodeljuje graf pariticije worker-ima
- Praćenje stanja, statistike i oporavak u slučaju grešaka

Worker:

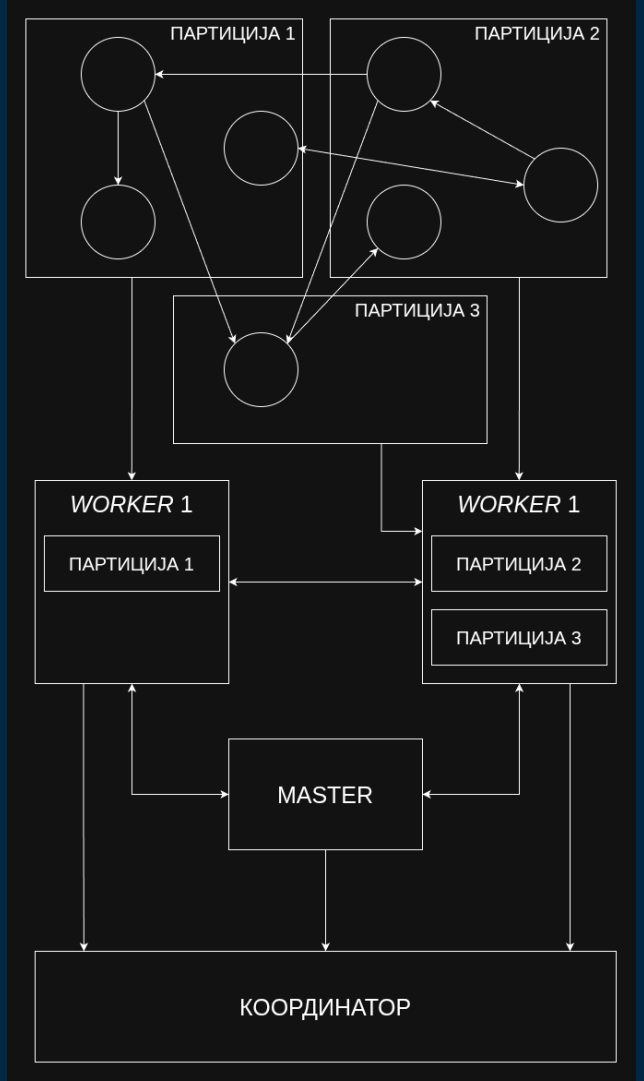
- Obrada lokalnih čvorova u dodeljenim paritacijama
- Razmena poruka sa drugim worker-ima
- Dinamičko rebalansiranje i checkpointing

Arhitektura

Koordinator (Coordinator)::

- Čuva konfiguracije, metapodatke i mapiranje paritacija
- Apache ZooKeeper se često koristi za ovu funkciju

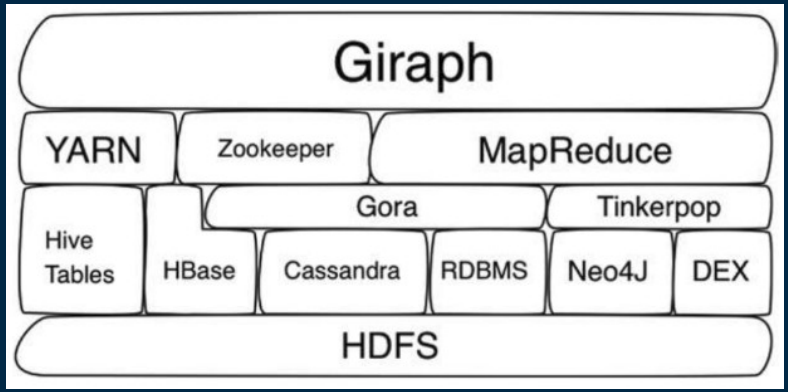
Arhitektura



Tehnologije

Apache Hadoop:

- YARN - upravljanje resursima u klasteru
- HDFS - skladištenje ulaznih i izlaznih podataka
- Integracija sa: MapReduce, HBase, Cassandra, Hive, HCatalog, Gora, Hama, Mahout, Nutch



Tehnologije

Apache ZooKeeper:

- Koordinator (Coordinator)
- Koordinacija i sinhronizacija master-a i worker-a

APACHE ZooKeeper™

Tehnologije

Java:

- Implementacija sistema - Apache Giraph je u potpunosti razvijen na JVM-u
- Programiranje - API i algoritmi se pišu u Javi
- Portabilnost i kompatibilnost - radi na različitim operativnim sistemima i u Hadoop ekosistemu

API – org.apache.giraph.graph.Vertex

Predstavlja čvor u grafu, osnovna jedinica vertex-centric algoritma.

Ključne metode:

- getId() – vraća jedinstveni ID čvora
- getValue() – vraća vrednost/stanje čvora
- setValue(V value) – postavlja novu vrednost čvora
- getEdges() – vraća kolekciju izlaznih grana
- getNumEdges() – broj izlaznih grana
- getEdgeValue(I targetId) – vrednost grane ka ciljanom čvoru
- setEdgeValue(I targetId, E value) – menja vrednost grane

API – org.apache.giraph.graph.Vertex

- addEdge(Edge<I,E> edge) – dodaje izlaznu granu
- removeEdges(I targetId) – uklanja sve grane ka određenom čvoru
- voteToHalt() – deaktivira čvor

API – org.apache.giraph.graph.Edge

Predstavlja grane između čvorova.

Ključne metode:

- getTargetVertexId() – vraća ID ciljnog čvora
- getValue() – vraća vrednost grane
- setValue(E value) – postavlja ili menja vrednost grane

API – org.apache.giraph.graph.BasicComputation

Definiše logiku obrade jednog čvora i razmenu poruka tokom svakog superstep-a u vertex-centric BSP modelu.

Ključne metode:

- compute(Vertex<I,V,E> vertex, Iterable<M> messages) – glavna metoda koja obrađuje čvor i njegove poruke
- getSuperstep() – vraća trenutni broj superstep-a
- getTotalNumVertices() – ukupno čvorova u grafu
- getTotalNumEdges() – ukupno grana u grafu
- sendMessage(I targetId, M message) – šalje poruku ciljanom čvoru
- sendMessageToAllEdges(Vertex<I,V,E> vertex, M message) – šalje poruku svim susedima

API – org.apache.giraph.graph.BasicComputation

- addVertexRequest(I vertexId) – dodaje novi čvor tokom izvršavanja
- removeVertexRequest(I vertexId) – uklanja čvor tokom izvršavanja

API – org.apache.giraph.io.formats

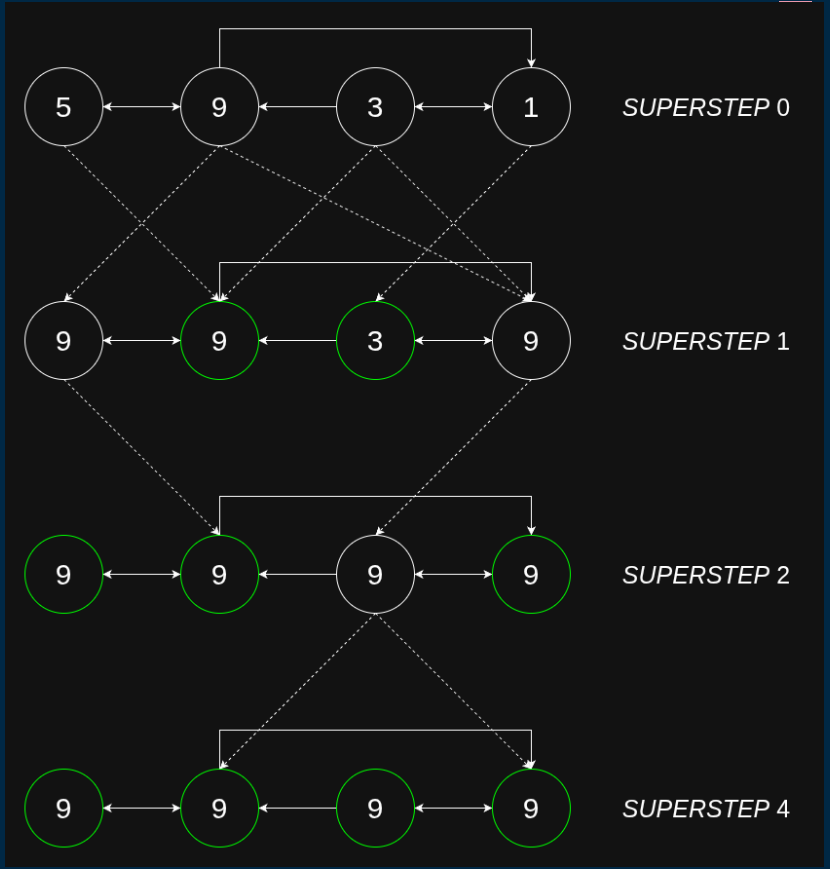
org.apache.giraph.io.formats.TextVertexInputFormat:

- Apstraktna klasa za čitanje čvorova iz tekstualnog fajla.

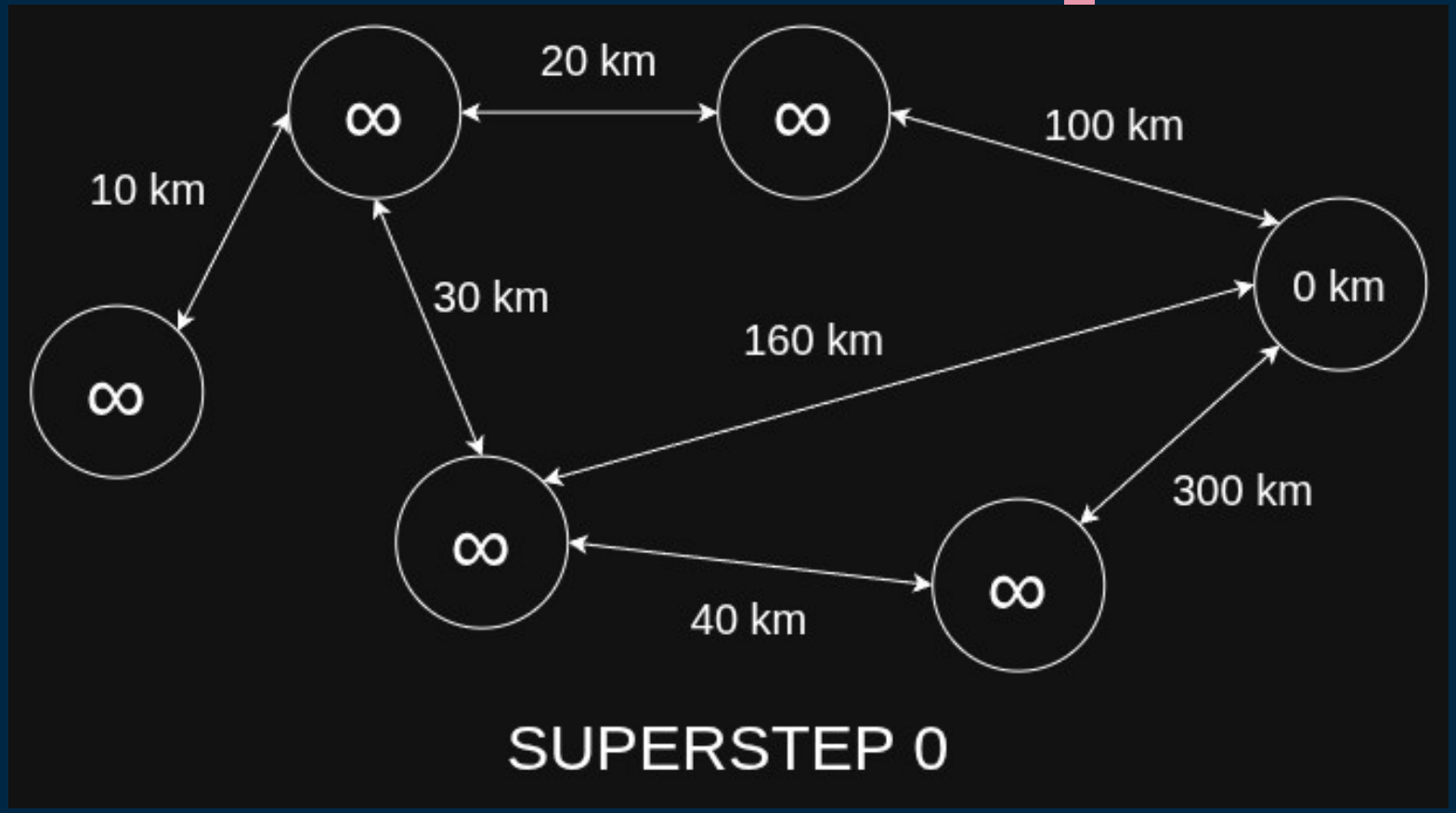
org.apache.giraph.io.formats.TextVertexOutputFormat:

- Apstraktna klasa za pisanje rezultata u tekstualni fajl.

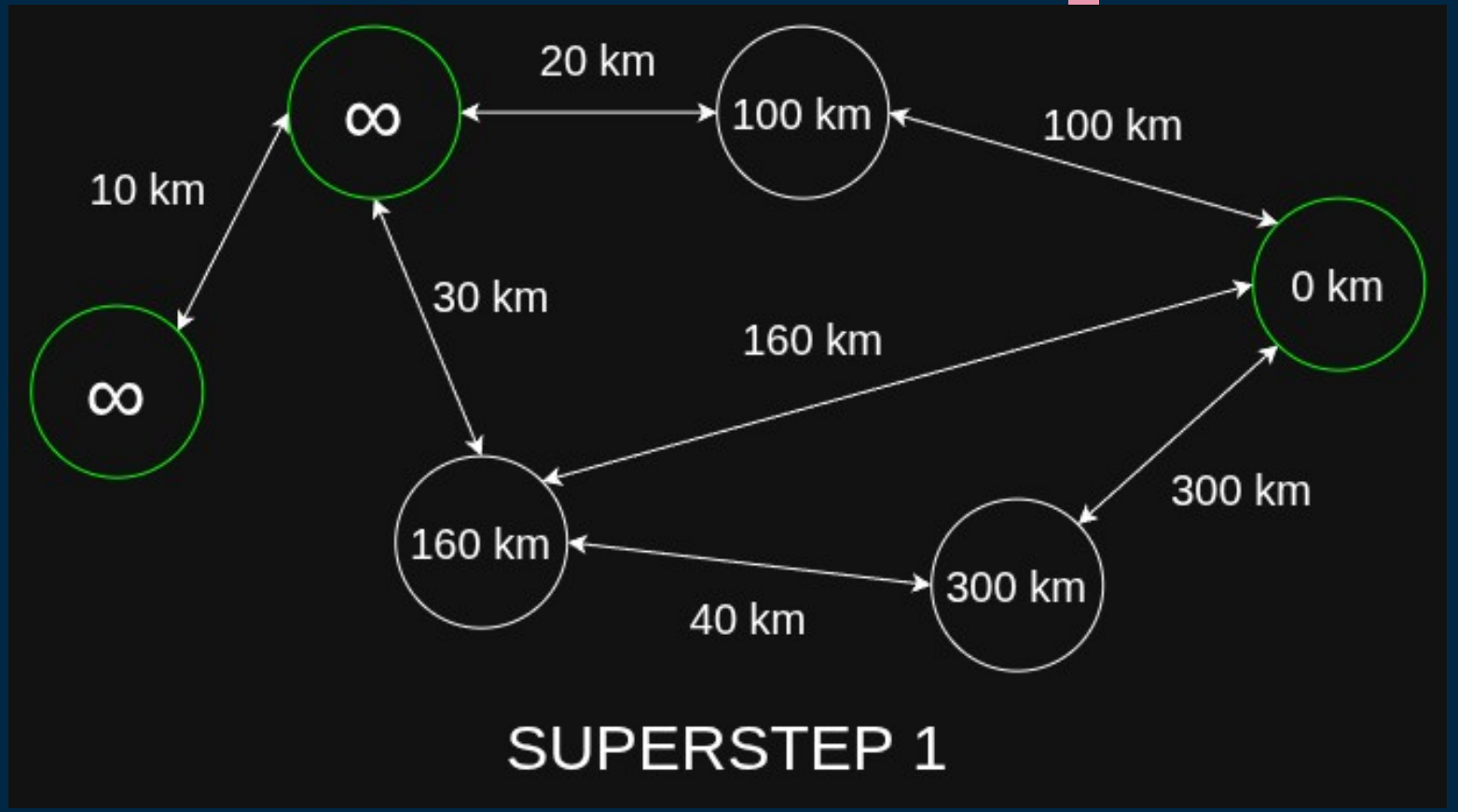
Primer – Pronalazak najveće vrednosti u grafu



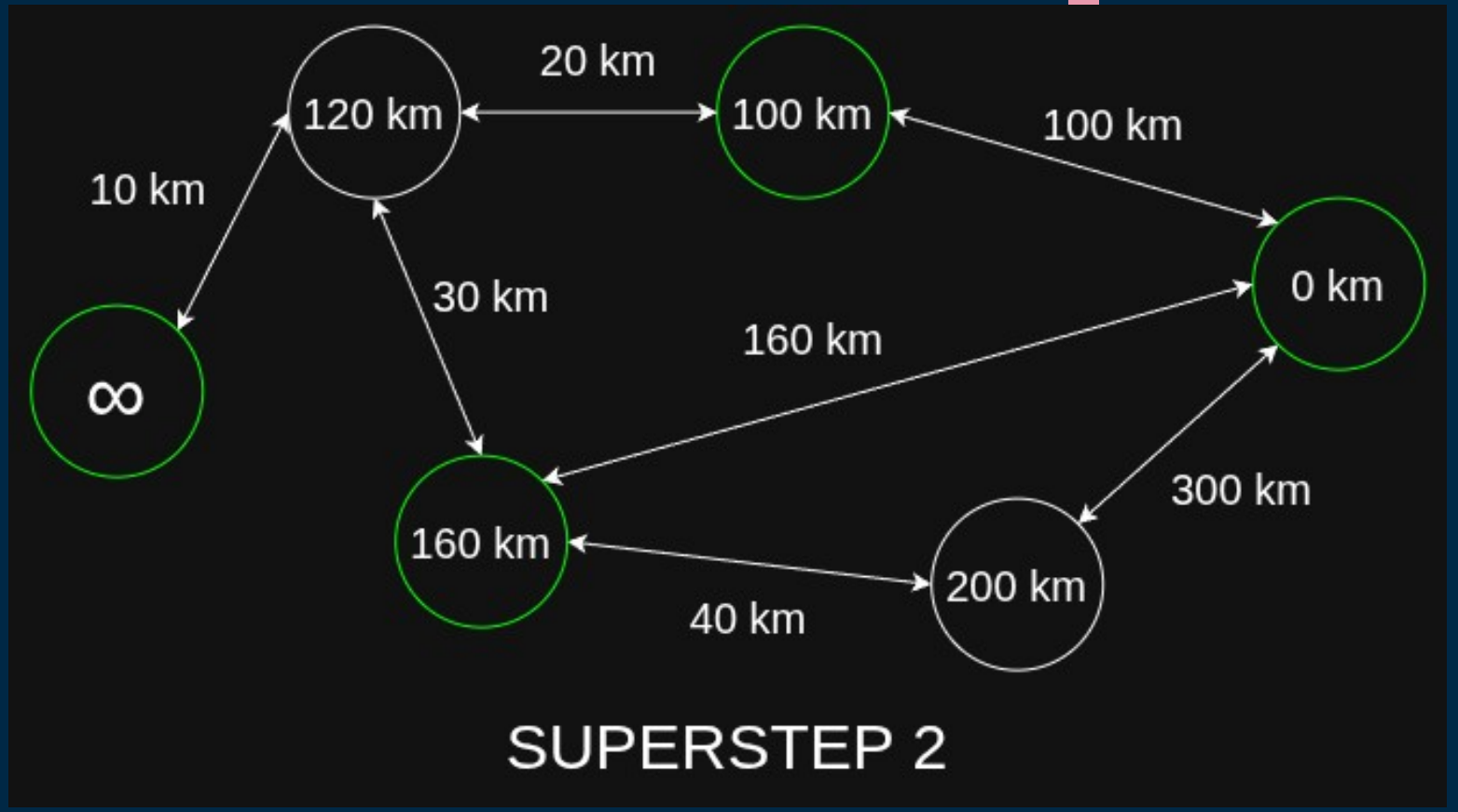
Primer – Pronalazak najkraćeg puta do glavnog grada



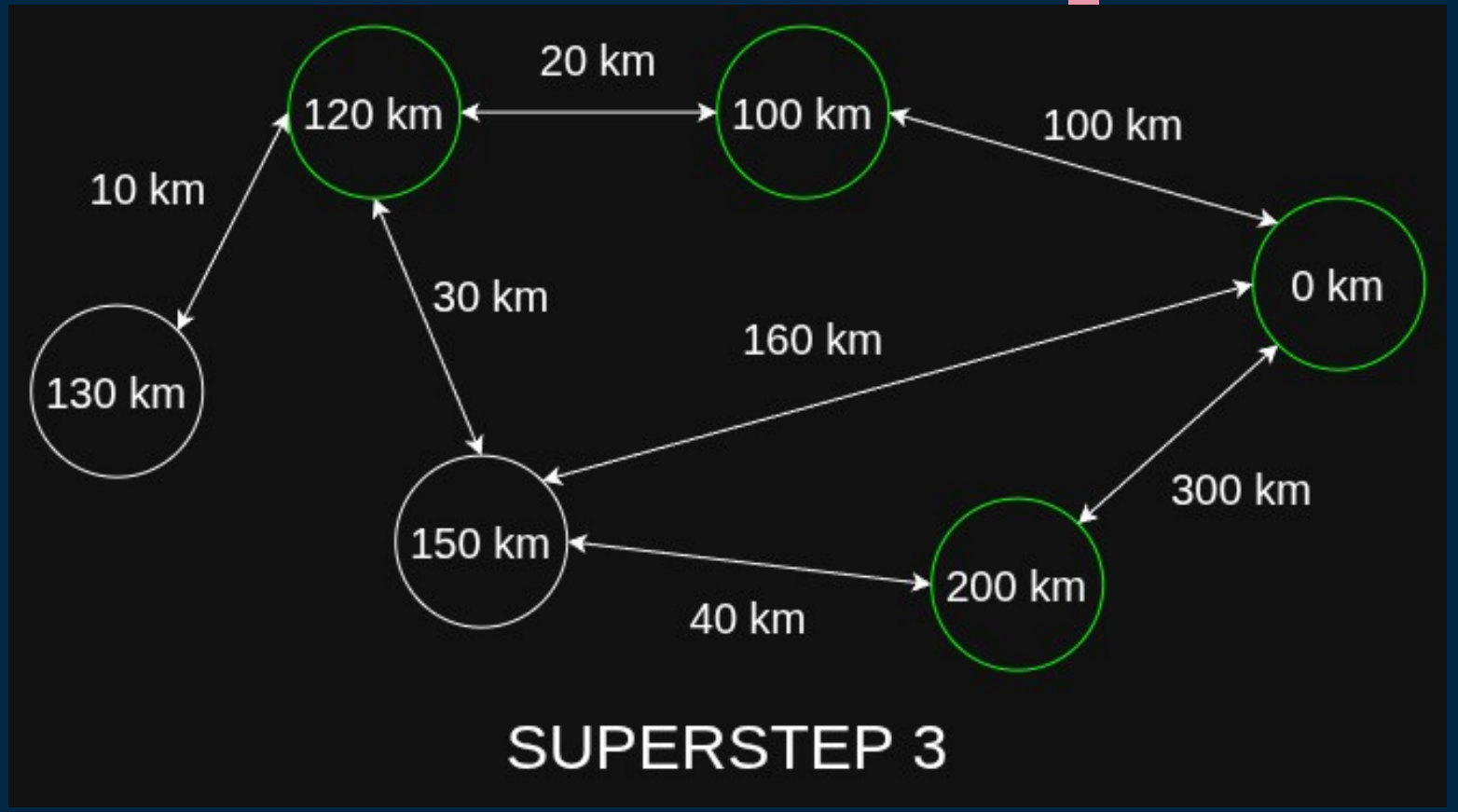
Primer – Pronalazak najkraćeg puta do glavnog grada



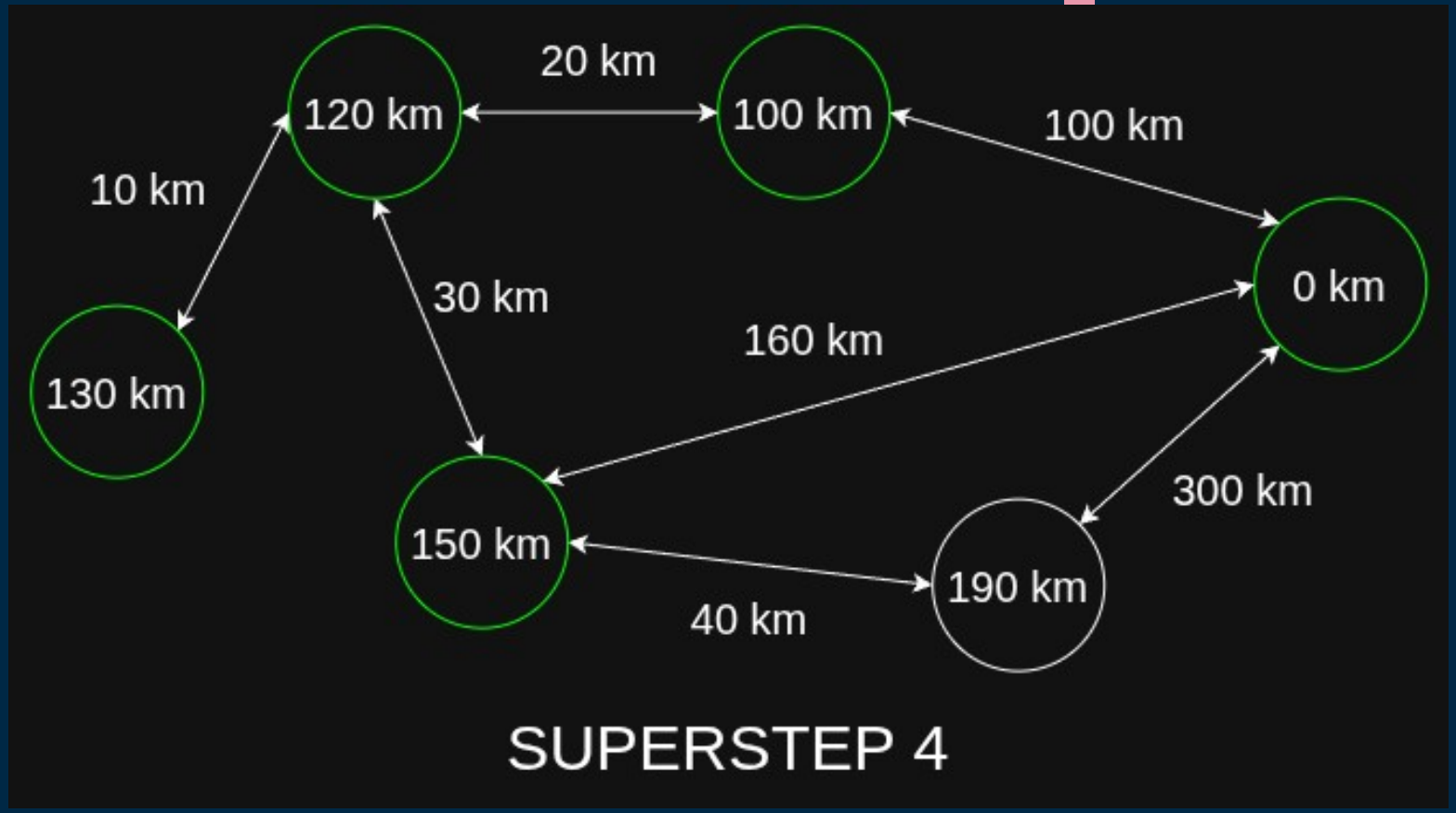
Primer – Pronalazak najkraćeg puta do glavnog grada



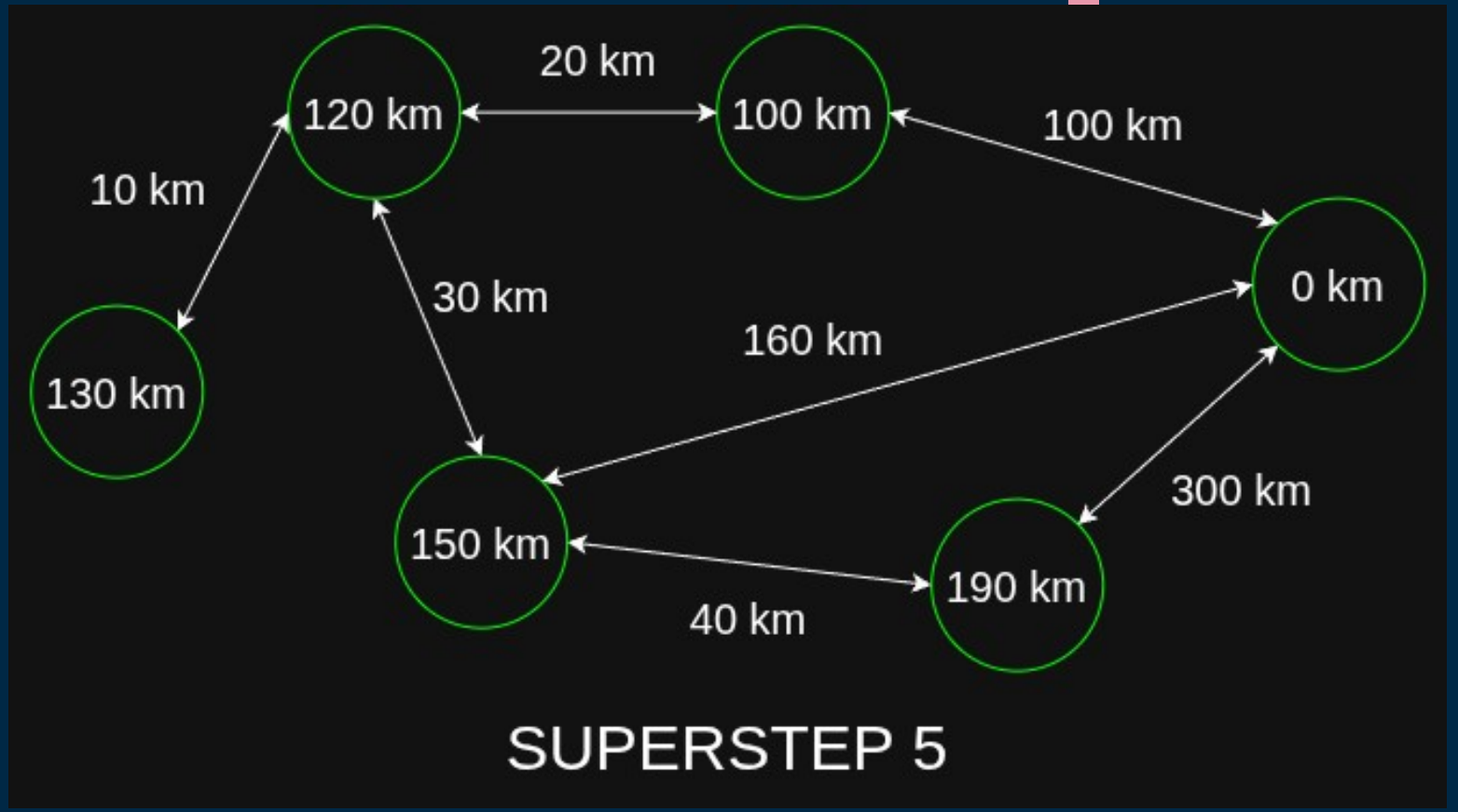
Primer – Pronalazak najkraćeg puta do glavnog grada



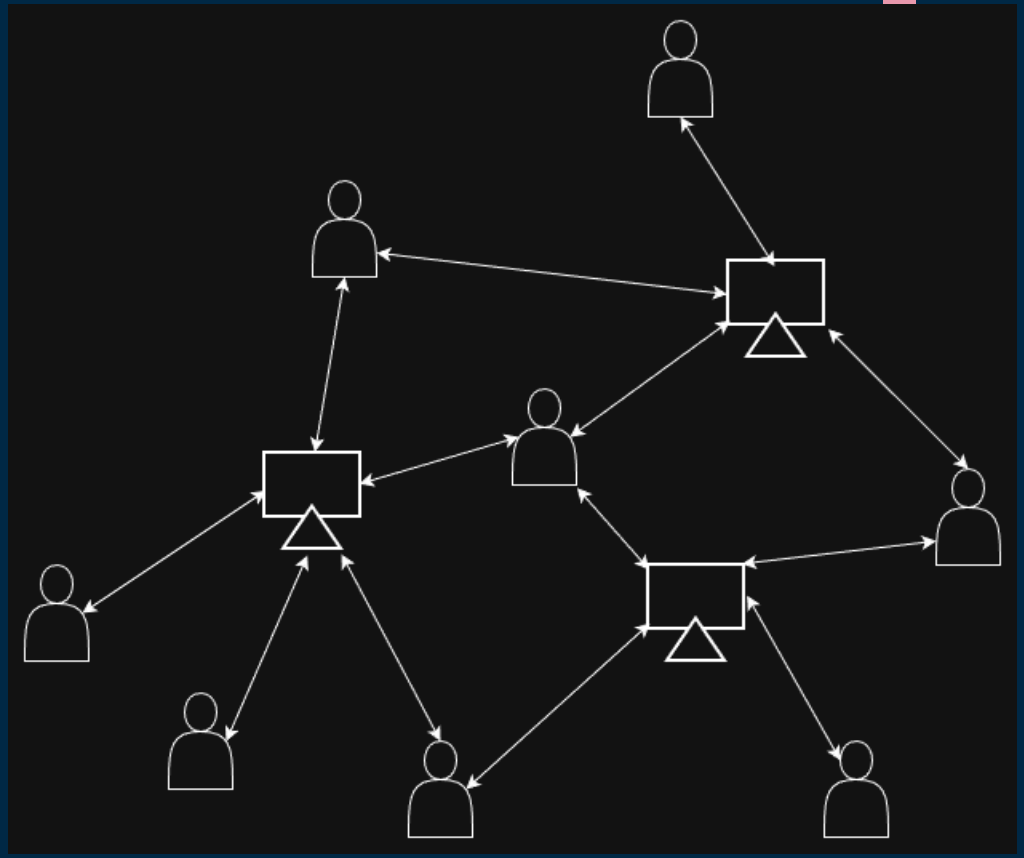
Primer – Pronalazak najkraćeg puta do glavnog grada



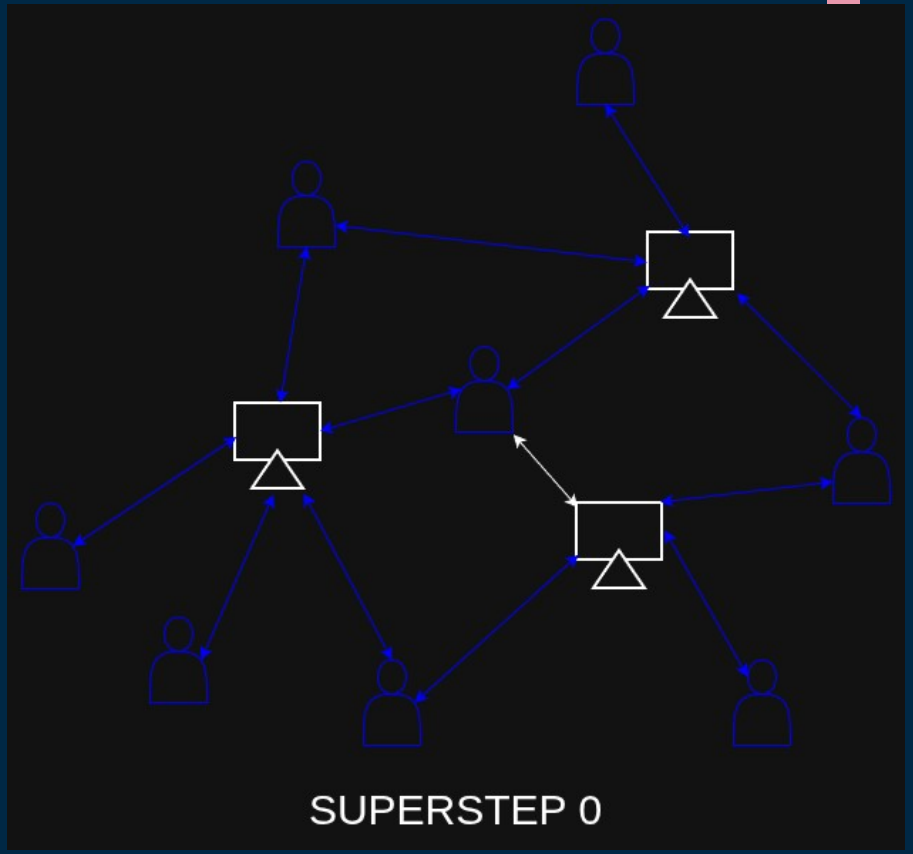
Primer – Pronalazak najkraćeg puta do glavnog grada



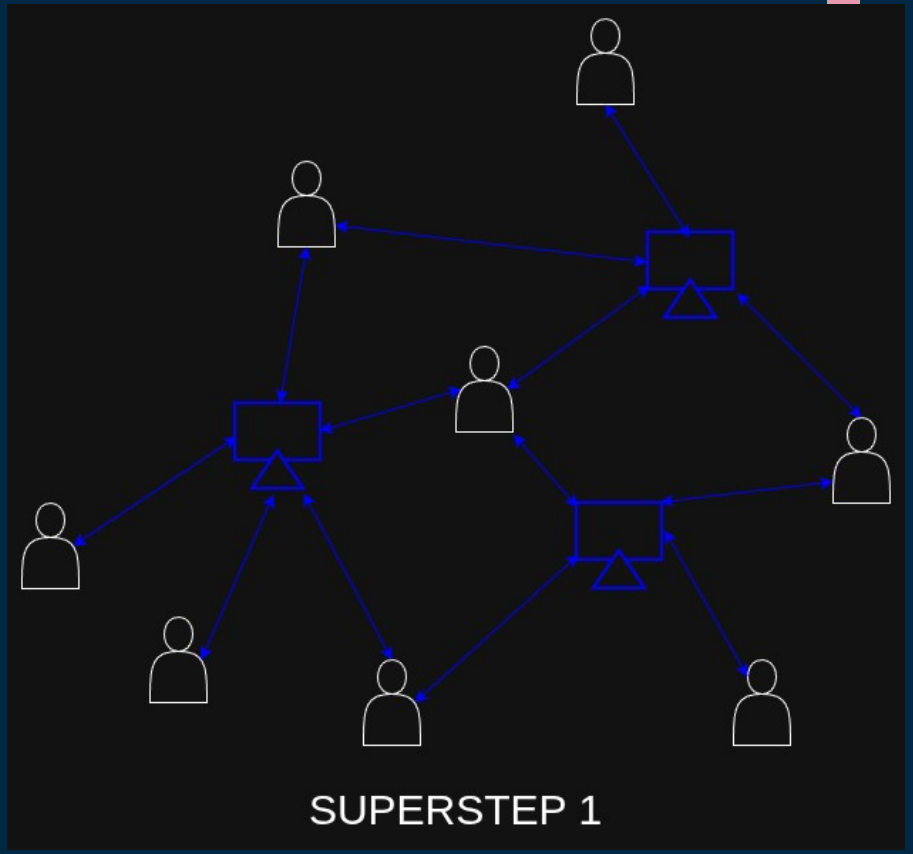
Primer – Preporuka filmova



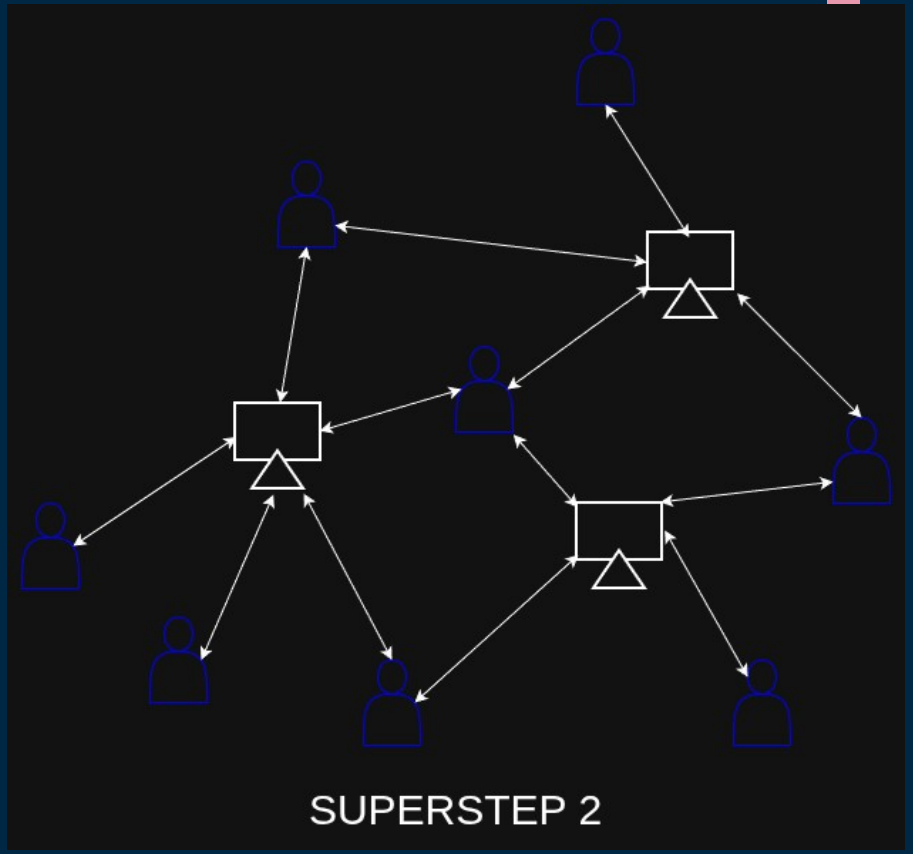
Primer – Preporuka filmova



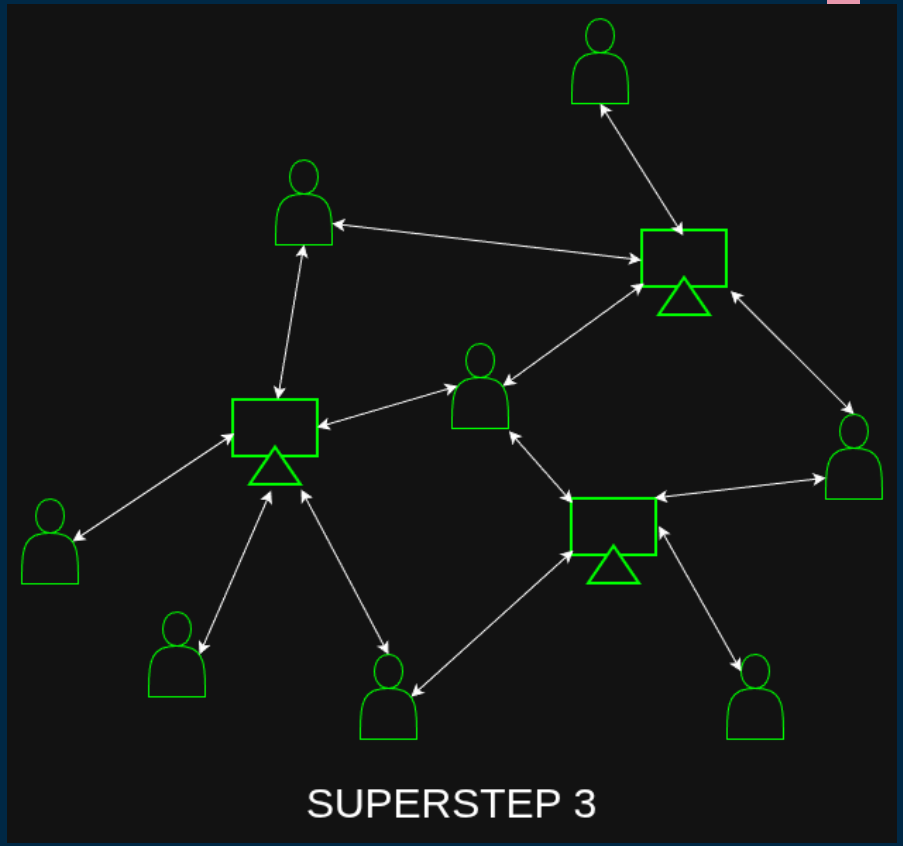
Primer – Preporuka filmova



Primer – Preporuka filmova



Primer – Preporuka filmova



HVALA NA PAŽNJI