

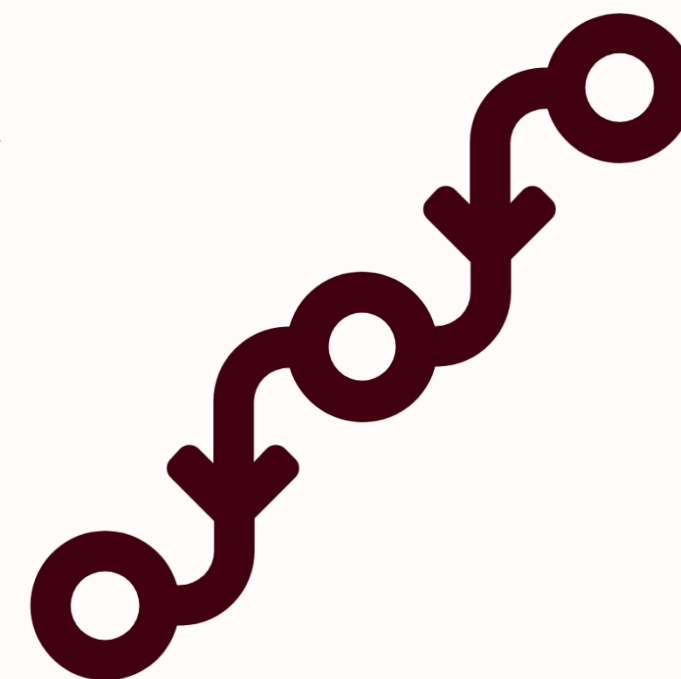
ELEKTROTEHNIČKI FAKULTET, UNIVERZITET U BEOGRADU
PARALELIZAM U ARHITEKTURI RAČUNARA I PROGRAMSKIM PREVODIOCIMA

DATAFLOW MAŠINE

doc. dr Maja Vukasović
as. ms Nikola Stanković

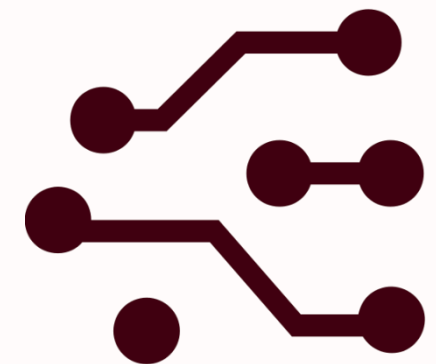
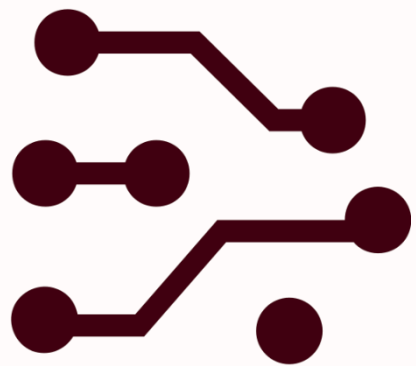
Control flow mašine

- Konvencionalni modeli programa su kontrolom pokretani, jer kontrolna jedinica određuje koja se instrukcija naredna izvršava.
 - Sekvenca instrukcija je precizno određena.
 - Kontrola određuje narednu instrukciju pomoću **programskog brojača/registra** (*instruction pointer*).
 - Ako nema skoka, naredna instrukcija po adresi u memoriji se sledeća izvršava.
 - Ako ima skoka, programom je definisano kako se određuje adresa naredne instrukcije (adresa skoka).

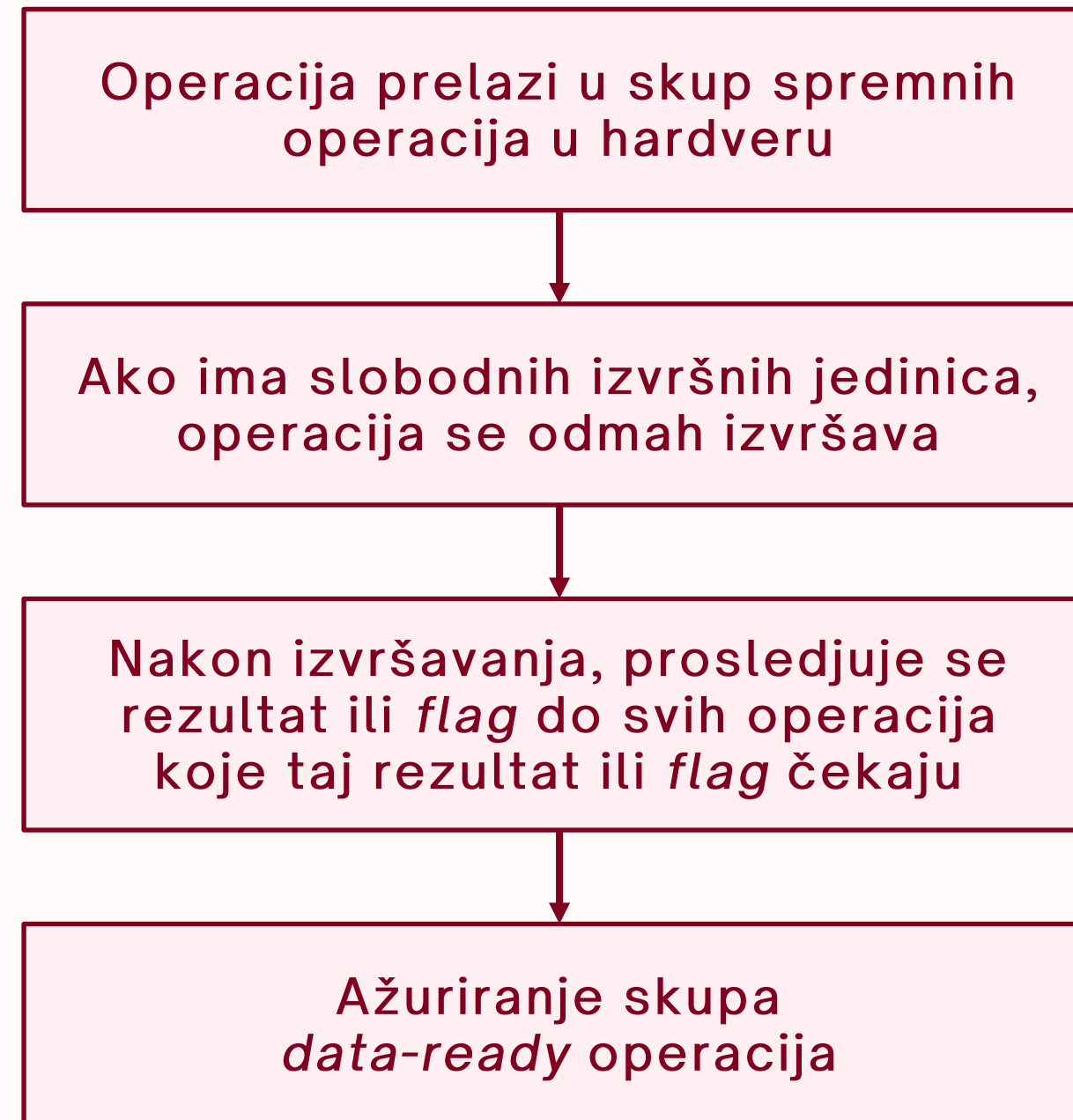


Dataflow mašine

- Uvodjenje zavisnosti po podacima:
 - Odredjivanje operacija spremnih za izvršavanje.
 - Operacije koje nisu zavisne od operacija koje još nisu završene.
 - Operacije koje nisu spremne čekaju da budu izračunati svi ulazi zbog kojih one nisu spremne.
- Razvoj *dataflow mašina* (podacima pokretane mašine).



Princip rada dataflow mašina (1)



Princip rada dataflow mašina (2)

- Operacije se pokreću jer su svi argumenti dostupni, a ne na osnovu programskog brojača.
- Redosled operacija je od značaja samo za referenciranje operacije, a ne za tok izvršavanja.
- Umesto programskog brojača, grafovi zavisnosti po podacima i kontroli se prilagođavaju u oblik pogodan za *dataflow*.
 - Efikasno ažuriranje vrednosti i statusa argumenata i operacija.

Programski jezici pogodni za dataflow

- Lako generisanje grafa zavisnosti po podacima.
- Eliminacija izlaznih i antizavisnosti:
 - SSA forma
 - Lakše se postiže programskim jezicima za funkcionalno programiranje.
- Izbegavaju se pokazivači kako bi se u vreme prevodjenja lakše odredilo da li postoje zavisnosti po podacima.
- Primeri:
 - *VAL (Value-Oriented Algorithmic Language)*
 - *Id (Irvine Dataflow)*
 - Lucid

Graf zavisnosti po podacima

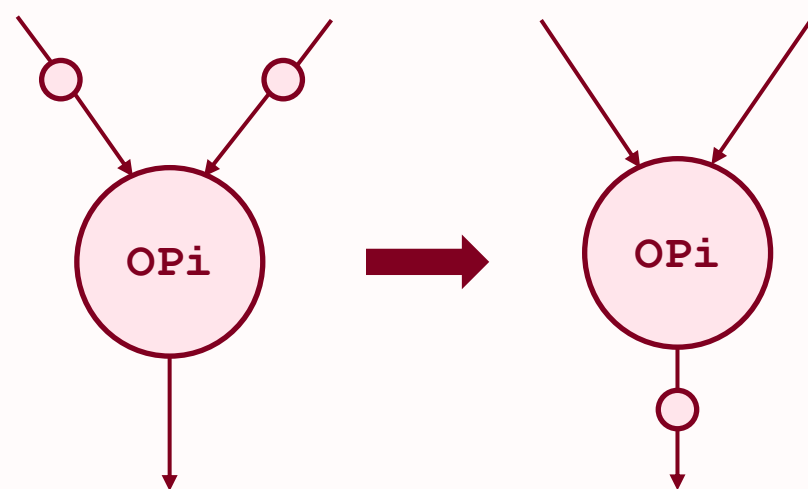
- *Data Dependency Graph* (DDG) formira se u vreme prevodjenja.
- Kojim operacijama treba proslediti izračunati rezultat?
- Pretraživanje grafa u vreme izvršavanja je sporo.
- Potrebno je operacijama dodeliti informaciju kome prosledjuju rezultat.
- Jedinstvena identifikacija operacija:
 - Adresa u memoriji na kojoj se nalazi torka koja opisuje operaciju.

Statičke dataflow mašine

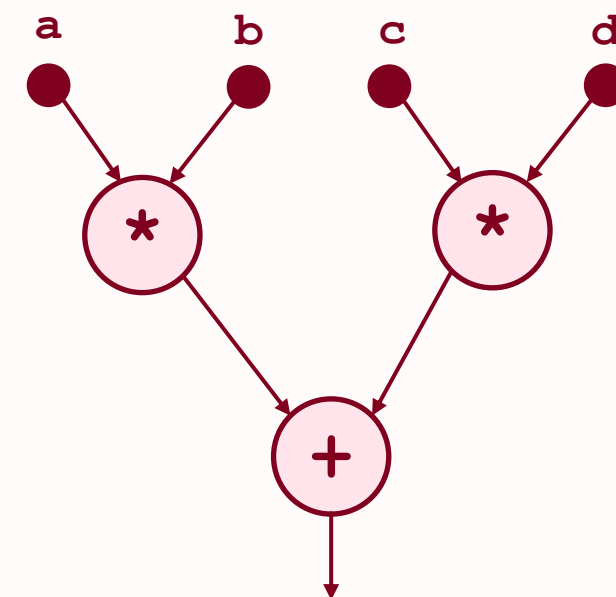
- Prvo su se pojavile statičke dataflow mašine.
- Kako DDG pretočiti u torke koje opisuju operacije?
- Referenciranje svih operacija koje su zavisne od neke operacije.

Primitivne operacije

- Za predstavljanje bazičnih blokova koriste se isključivo primitivne operacije (čvorovi grafa).
- Ne odnose se na kontrolne zavisnosti.
- Grane grafa – zavisnosti izmedju operacija.
- Kružići = torke (**data tokeni**).
 - *Data-ready* vrednosti ulaza čvorova.
 - Rezultat operacije koji se dalje prosledjuje.



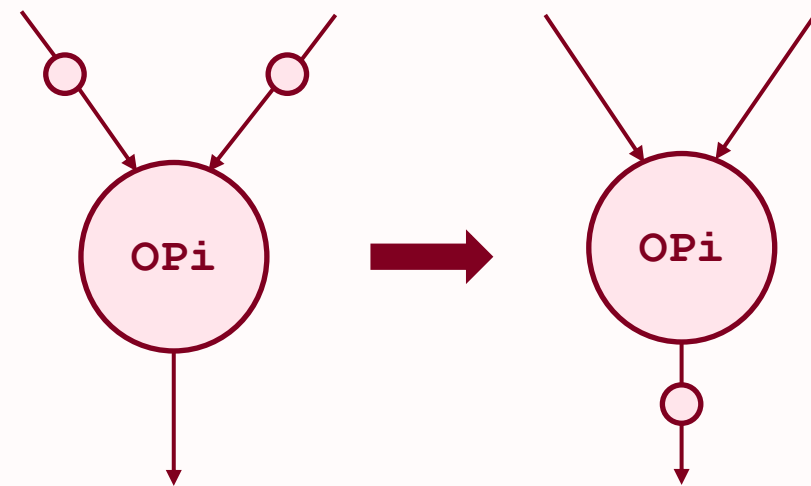
Data tokeni na ulazu i na izlazu čvora



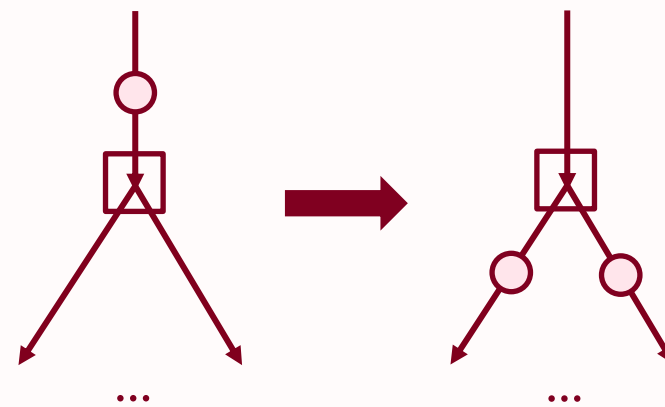
Dataflow graf za bazični blok koji izračunava $a*b+c*d$

Dataflow operatori (1)

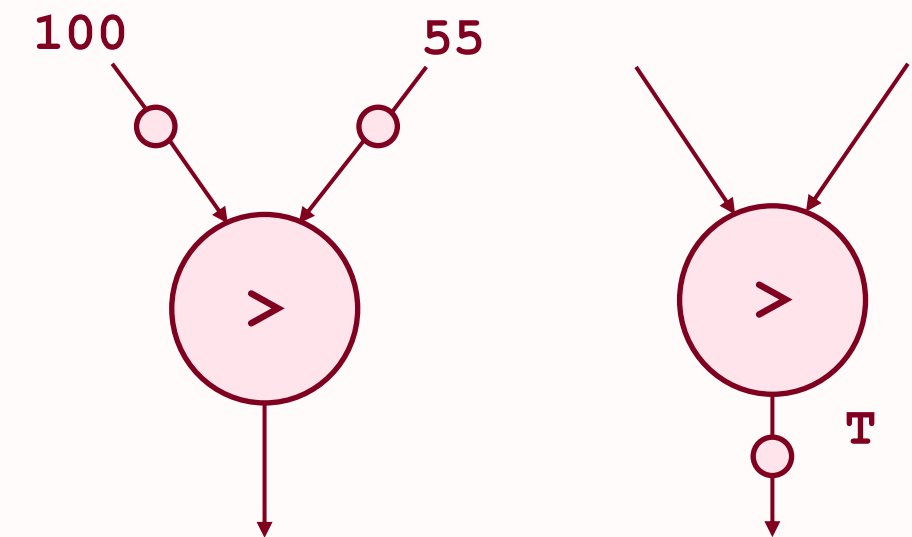
- Izvršavanje operacija je asinhrono.



Primitivna operacija



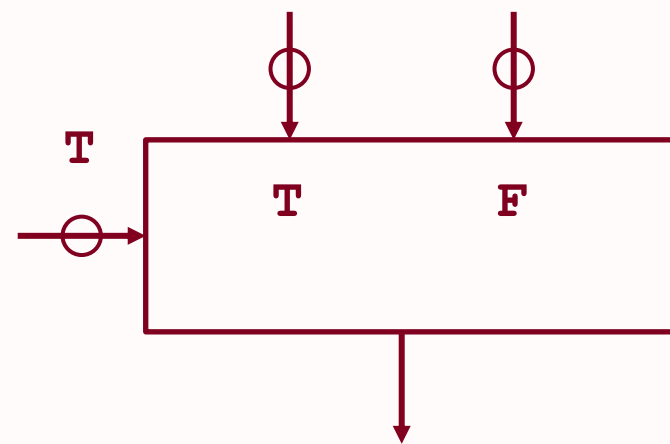
Fork



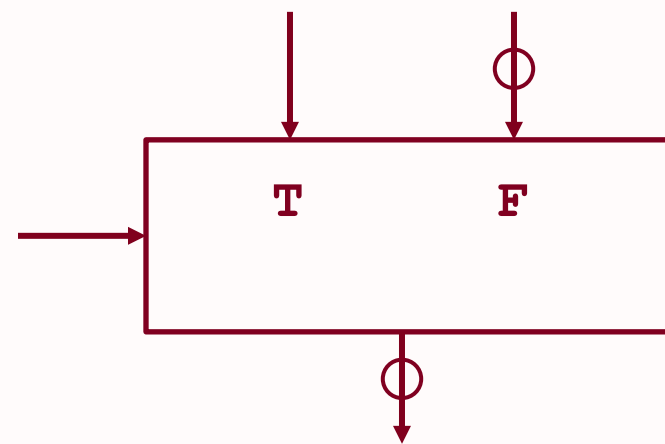
Relacioni operator

Dataflow operatori (2)

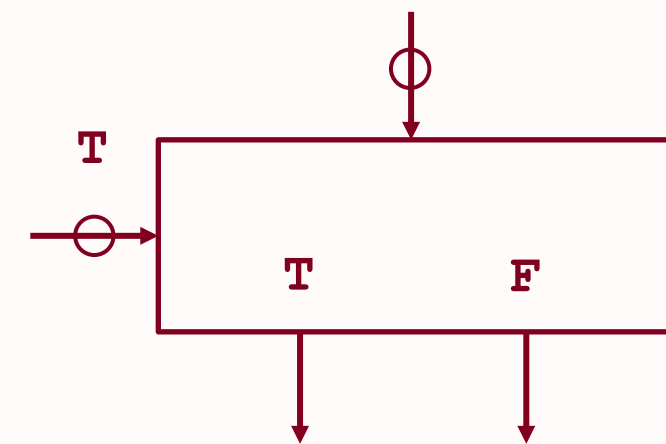
- Izvršavanje operacija je asinhrono.



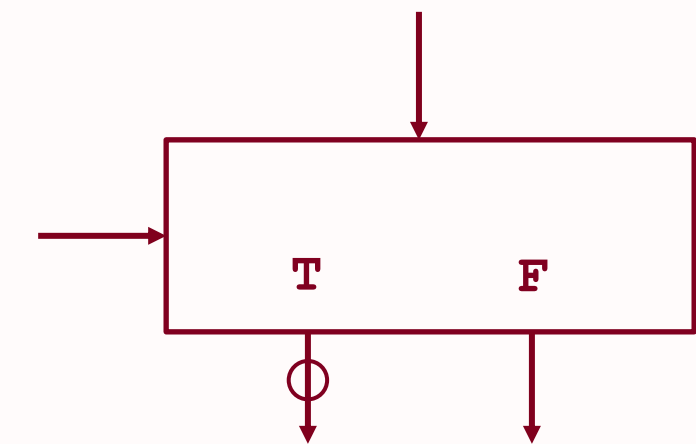
Spajanje (merge) – pre izbacivanja data tokena



Spajanje (merge) – posle izbacivanja data tokena



Switch



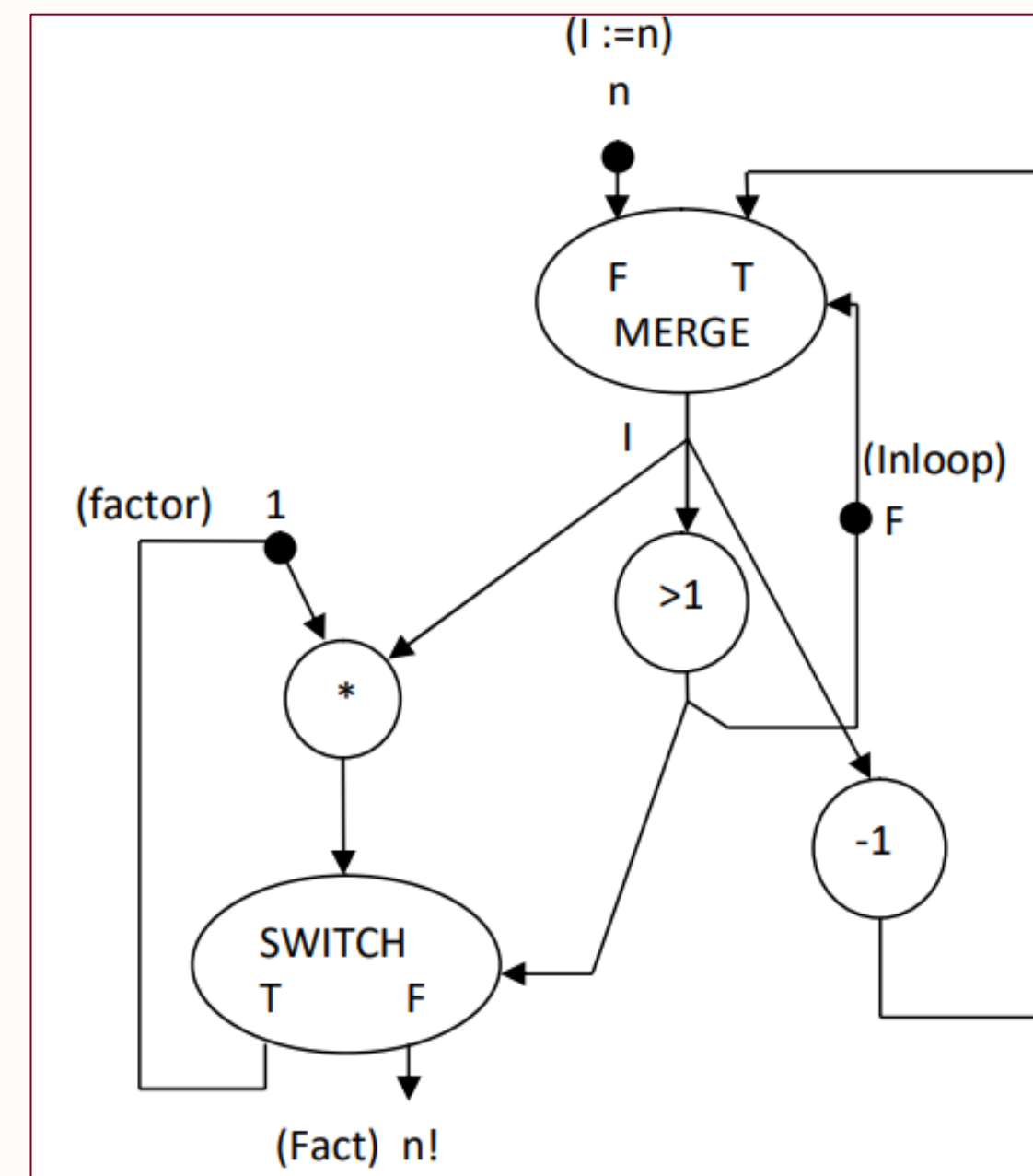
Primer transformacije grafa

- Izračunavanje faktoriijela:

```
long fact(n) {  
    if (n == 0) return 1;  
    else return n*fact(n-1);  
}
```

- Drugi oblik (prilagodjen *dataflow* notaciji):

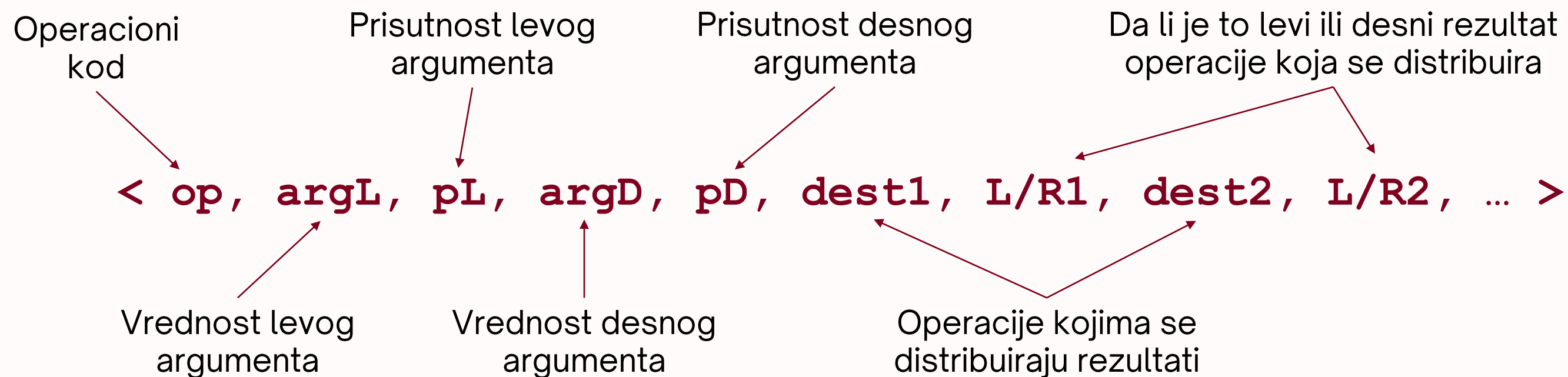
```
Fact(n)  
inloop := false; I := n; factor := 1;  
WHILE I > 1  
    factor := factor * I; I := I - 1; inloop := (I > 1);  
ENDWHILE  
Return Fact := factor;  
END Fact;
```



Dataflow dijagram

Forma operacije za statičke mašine (1)

- Oblik torke:
 - Vrednosti argumenata
 - *Data-ready* biti
 - Informacija kome se distribuiraju rezultati

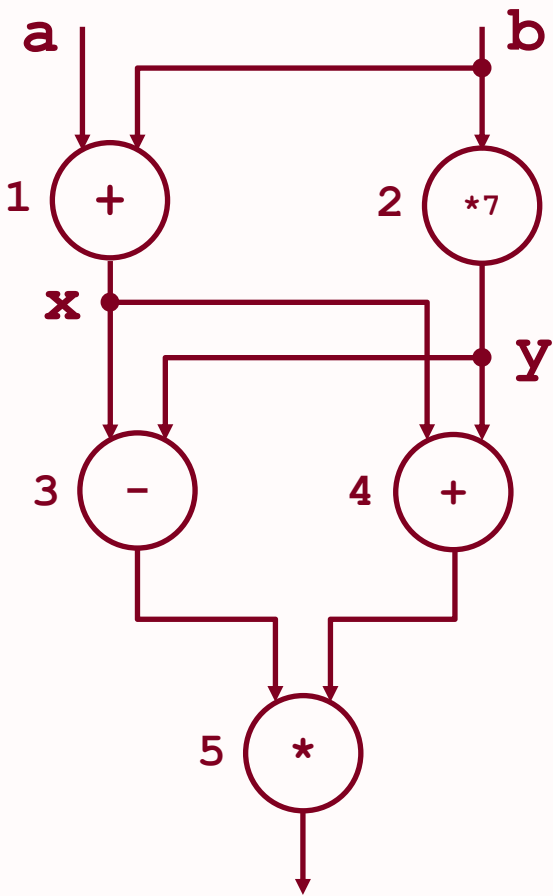


Forma operacije za statičke mašine (2)

- Nepogodno je zbog promenljivog broja destinacija.

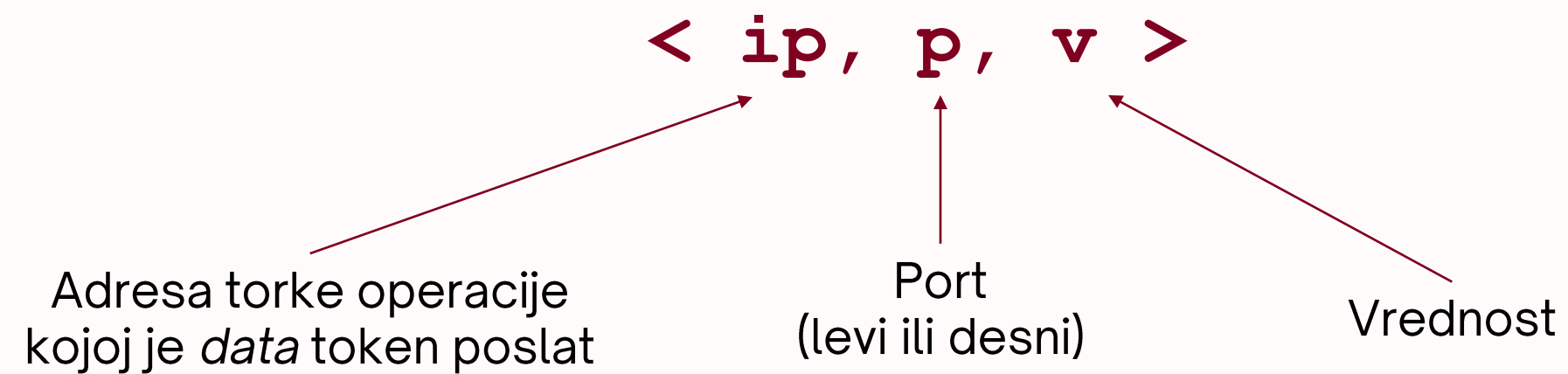
	OpCode	Destination 1	Destination 2		Operand 1		Operand 2
1	+	3L	4L				
2	*	3R	4R				
3	-	5L					
4	+	5R					
5	*	out					

Presence biti ⇔ Ready biti

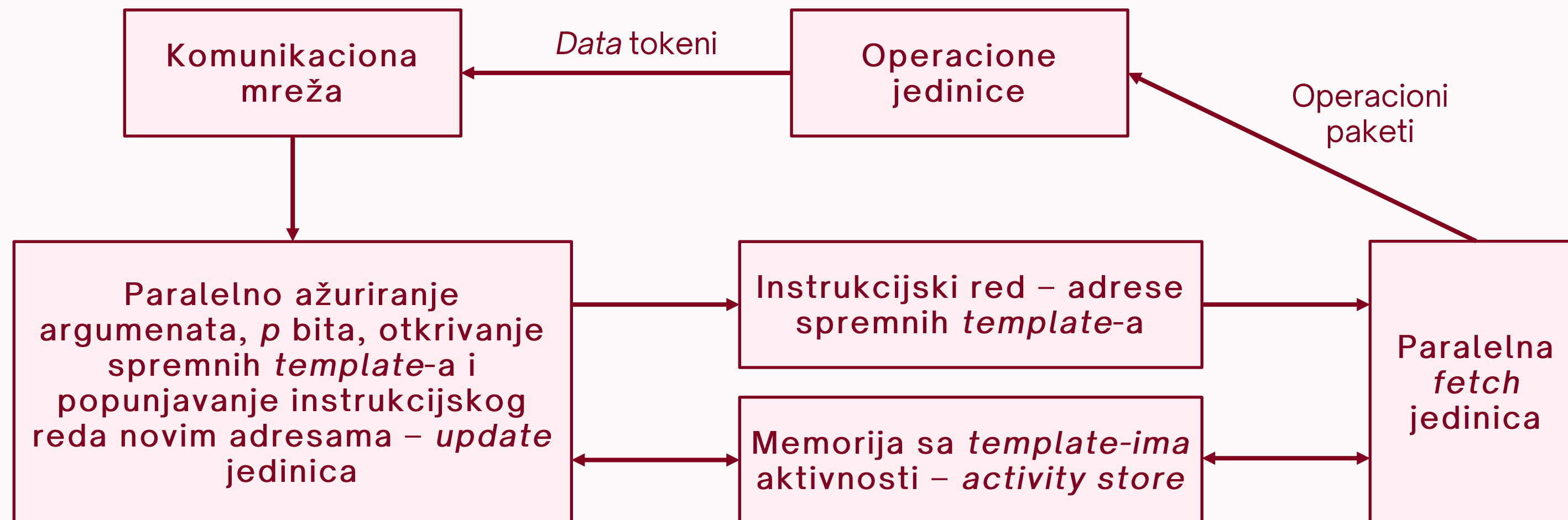


Forma podataka

- Torke za podatke – *data* tokeni:



Konceptualni model statičkih dataflow mašina (1)



Konceptualni model statičkih dataflow mašina (2)

- Problemi:
 - Delovi koda koji se ponavljaju, petlje i procedure koje se pozivaju.
 - *DoAll* petlje.
 - Previše argumenata iz različitih konteksta (iteracija).
 - Nekorektno izvršavanje ukoliko se pomešaju argumenti iz različitih originalnih iteracija.
 - Problemi sa jedinicom za ažuriranje.
- Prošireni uslov izvršavanja operacije:
 1. Dostupni svi argumenti.
 2. Port koji treba da primi token mora biti slobodan (signal potvrde).
- Značajno je smanjen nivo paralelizma.

Dinamički dataflow

- Razdvajanje operacije od detekcije kada je operacija spremna za izvršavanje.
- Više instanci izvršavanja nad kojima se vrši uparivanje argumenata za iste operacije.
- Dodato je polje *frame* u *data* tokenu koje identifikuje iteraciju ili pozivanje.
- Paralelizam je veći, ali je i dalje ograničen.

Prednosti i mane

- Teži se da jedina ograničenja budu zavisnosti po podacima.
- Otežano je debugovanje.
- Paralelizam je limitiran.
- Kompleksna implementacija dinamičkog *dataflow*-a i veće memorijsko zauzeće.
- Kombinovanje dobrih osobina *dataflow* i *control flow* mašina (paradigmi).
- Ugradjena *dataflow* mašina.