

1. Aplikacija generiše *burst* veličine 3MB i trajanja 2ms, svakih 100ms. Mreža nudi propusnost od 60MB/s. *Leaky bucket* ima kapacitet 4MB.
 - a) Skicirati saobraćaj na ulazu i izlazu mrežog uređaja koji izvršava *leaky bucket* algoritam.
 - b) Odrediti minimalni kapacitet *leaky bucket*-a pri kojem neće doći do gubitaka paketa.

Rešenje:

a)

$$R_{ul} = \frac{3MB}{2ms} = 1500MB/s$$

$$t_{preniz} = \frac{3MB}{60MB/s} = 0.05s = 50ms$$

Ulaz: 0-2ms: 1500MB/s; 100-102ms: 1500MB/s; 200-202ms: 1500MB/s; ...

Izlaz: 0-50ms: 60MB/s; 100-150ms: 60MB/s; ...

b)

Tokom trajanja *burst*-a dolazna brzina saobraćaja je 1500MB/s a odlazna 60MB/s. Stoga brzina akumulacije saobraćaja je 1440MB/s. Stoga, nakon 2ms, akumuliraće se:

$$1440 \frac{MB}{s} \cdot 0.002s = 2.88MB.$$

Ovo je ujedno i minimalni kapacitet *leaky bucket*-a pri kojem neće doći do gubitaka paketa.

2. Aplikacija generiše *burst* veličine 0.5MB i trajanja 250ms, svakih 3s. Kapacitet mreže je 10 MB/s. Ukoliko je kapacitet *token bucket*-a 1MB, a brzina punjenja *bucket*-a 2MB/s, koliko dugo se saobraćaj može prenositi bez kašnjenja? Odrediti popunjenost bucketa u trenutku dolaska drugog *burst*-a.

Rešenje:

- a) Inicijalno, izlazna brzina biće 10MB/s (pretpostavljamo da je *token bucket* pun na početku ukoliko nije drugačije naglašeno). S obzirom da je kapacitet *token bucket*-a 1MB, *burst* veličine 1MB može se prenijeti maksimalnom brzinom 10MB/s. *Token bucket* se takođe puni brzinom 2MB/s, pa možemo napisati jednačinu:

$$1MB + 2 \frac{MB}{s} \cdot X = 10 \frac{MB}{s} \cdot X \Rightarrow X = \frac{1}{8} s = 125ms$$

X je vrijeme tokom kojeg se saobraćaj prenosi bez kašnjenja. Tokom ovog vremena *bucket* može prenijeti 1.25MB, što je veće od 0.5MB. Zaključujemo da se prvi *burst* prenosi bez kašnjenja.

Izlaz: 0-50ms: 10MB/s

50-250ms: Ništa

Na kraju ovog perioda količina tokena u *bucket*-u je:

$$1MB + 250ms \cdot 2MB/s - 0.5MB = 1MB$$

Stoga, *bucket* će biti ponovo pun kada dođe sledeći burst, pa se situacija ponavlja naredne 3s.

b) Kada je veličina *bucket*-a 0.2MB, imamo:

$$0.2MB + 2 \frac{MB}{s} X = 10 \frac{MB}{s} X \Rightarrow X = 0.2 / 8 = 0.025s = 25ms$$

Izlaz: 0-25ms: 10MB/s (prenosi se 0.25MB, ostaje 0.25MB prvog *burst*-a)

Za prenos 0.25MB brzinom 2MB/s potrebno je: $\frac{0.25MB}{2MB/s} = 125ms$

25-150ms: 2MB/s

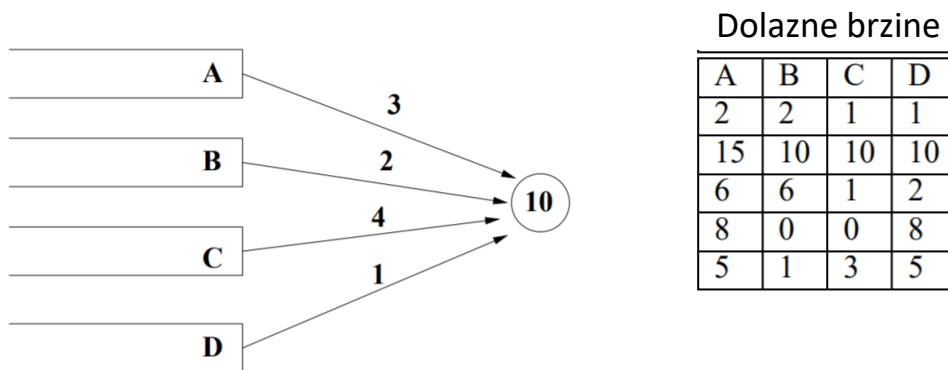
150-250ms: Ništa

Na kraju ovog perioda količina tokena u *bucket*-u je:

$$0.2MB + 250ms \cdot 2MB/s - 0.5MB = 0.2MB$$

pa se situacija ponavlja dok se ne završi prenos svih *burst*-ova.

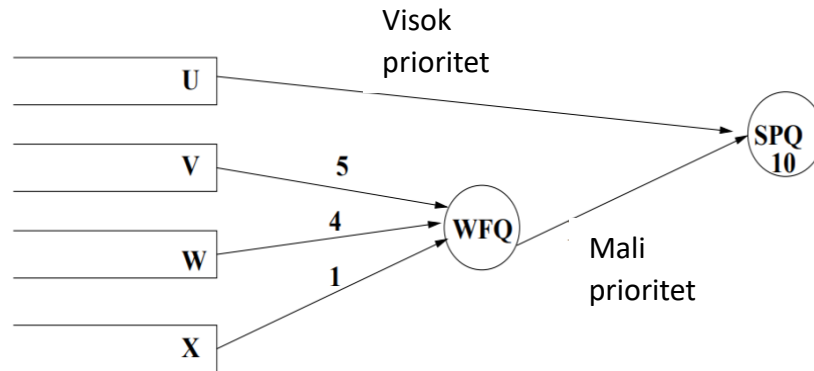
3. Razmotriti sistem sa 4 bafera koji se opslužuju WFQ algoritmom, kao što je prikazano na slici. Težinski faktori bafera (A,B,C,D) su 3, 2, 4 i 1 respektivno. Baferi dijele kapacitet linka od 10Mb/s. Za zadate dolazne brzine paketa u svaki od bafera odrediti rezultujuće brzine opsluživanja.



Rešenje:

| Dolazne brzine | | | | Brzine opsluživanja | | | |
|----------------|----|----|----|---------------------|---|---|-----|
| A | B | C | D | A | B | C | D |
| 2 | 2 | 1 | 1 | 2 | 2 | 1 | 1 |
| 15 | 10 | 10 | 10 | 3 | 2 | 4 | 1 |
| 6 | 6 | 1 | 2 | 4.5 | 3 | 1 | 1.5 |
| 8 | 0 | 0 | 8 | 7.5 | 0 | 0 | 2.5 |
| 5 | 1 | 3 | 5 | 4.5 | 1 | 3 | 1.5 |

4. Razmotrimo scenario sa 4 klase saobraćaja: U, V, W i X koje dijele link kapaciteta 10Mbit/s. Klasa U odnosi se na kontrolni saobraćaj i treba da ima apsolutni prioritet prilikom prenosa. Klase V, W i X se odnose na saobraćaj podataka (npr. HTTP, SMTP). Predviđeno je da klasa V dobija 5 puta veću brzinu prenosa od klase X, a klasa W četiri puta veću brzinu prenosa od klase X. Imajući u vidu ove zahtjeve dodjeliti bafer svakoj od navedenih saobraćajnih klasa i predložiti strategiju opsluživanja. Za zadate brzine popunjavanja bafera odrediti rezultujuće brzine opsluživanja.



Rešenje:

| Dolazne brzine | | | | Brzine opsluživanja | | | |
|----------------|---|----|---|---------------------|-----|-----|-----|
| U | V | W | X | U | V | W | X |
| 2 | 6 | 2 | 1 | 2 | 5 | 2 | 1 |
| 1 | 6 | 10 | 6 | 1 | 4.5 | 3.6 | 0.9 |
| 2 | 1 | 1 | 1 | 2 | 1 | 1 | 1 |