

Telekommunikációs Hálózatok

2. gyakorlat

Elérhetőségek

- gyakorlatvezető: Szalai-Gindl János
- honlap: <http://szalaigj.web.elte.hu/>
- email: szalaigindl@inf.elte.hu
- szoba: 2.507 (déli tömb)

Egyszerű számítások

SZÁMOLÓS FELADATOK

Alapfogalmak

- **Hálózati sávszélesség:** bit per másodpercben
- **Propagációs késés:** az az időtartam, amely a jelnek szükséges ahhoz, hogy a küldőtől megérkezzen a címzetthez.
- **Átviteli késleltetés:** az az időtartam, amely egy csomag összes bitjének az átviteli csatornára tételéhez szükséges.

SI szabvány

$8 \cdot 10^3$ bit/sec	1 KB/s	egy kiló-bájt
$8 \cdot 10^6$ bit/sec	1 MB/s	egy mega-bájt
$8 \cdot 10^9$ bit/sec	1 GB/s	egy giga-bájt
$8 \cdot 10^{12}$ bit/sec	1 TB/s	egy tera-bájt
$8 \cdot 10^{15}$ bit/sec	1 PB/s	egy peta-bájt
$8 \cdot 10^{18}$ bit/sec	1 EB/s	egy exa-bájt

IEC szabvány

$8 \cdot 2^{10}$ bit/sec	1 KiB/s	egy kibi-bájt
$8 \cdot 2^{20}$ bit/sec	1 MiB/s	egy mebi-bájt
$8 \cdot 2^{30}$ bit/sec	1 GiB/s	egy gibi-bájt
$8 \cdot 2^{40}$ bit/sec	1 TiB/s	egy tebi-bájt
$8 \cdot 2^{50}$ bit/sec	1 PiB/s	egy pebi-bájt
$8 \cdot 2^{60}$ bit/sec	1 EiB/s	egy exbi-bájt

Feladat 1

- Egy kép 1024 x 768 képpontos méretű, 3 bájt/képpontos színfelbontású. Tegyük fel, hogy a kép nincs tömörítve. Mennyi ideig tart átvinni ezt a képet egy
 - 56 kb/s-os modemcsatornán?
 - 1 Mb/s-os kábelmodemen?
 - 10 Mb/s-os Etherneten?
 - 100 Mb/s-os Etherneten?

Feladat 2

Tegyük fel, hogy egy aszimmetrikus pont-pont kapcsolat köti össze a földi bázisállomást és egy újonnan felépült holdbázist. A földről a holdra 100 Mbps, míg fordítva 10 Gbps a kapcsolat sávszélessége. A Föld és a Hold távolsága megközelítőleg 385 000 km. Az adatokat rádióhullámok segítségével továbbítjuk, azaz a jelterjedés sebessége mindkét irányban kb. $3 \cdot 10^8$ m/s.

- Számítsa ki a minimális RTT-t a fenti linkre! RTT (Round Trip Time) = az az idő, amire egy csomagnak szüksége van ahhoz, hogy A-ból eljusson B-be, majd onnan vissza A-ba. Tegyük fel, hogy a csomag mérete 0.

Feladat 3

Számítsa ki a késleltetést az első bit elküldésétől az utolsó megérkezéséig a következő esetekben:

a) Adott egy 24 Mbps-os link, melyet

- egy egyszerű switch
- store-and-forward

oszt két szakaszra. A szakaszokon a *propagációs késés* egyenként 15 ms. Mekkora a teljes késleltetés egy 5000 bit méretű csomag átküldésénél?

► A *switch*-en a csomag fogadása és a továbbítása további késés nélkül, közvetlenül egymást után történik.

b) Számítsa ki ugyanezt N darab *switch*-cel! (Minden szakaszon 15 ms a propagációs késés.)

Kis valószínűségszámítás...

- A diszkrét *valószínűségi változó* egy olyan változó, amelynek valamely véletlen kimenetelű kísérlettől (vagy kísérletektől) függően véges vagy megszámlálhatóan végtelen sok értéke lehet, továbbá egy lehetséges értékét valamilyen valószínűséggel veheti fel. (Pl. szabályos dobókockával "kísérletezünk", és a valószínűségi változó értéke a kockán található szám lesz. Ekkor minden $\{1,2,3,4,5,6\}$ értékét $1/6$ valószínűséggel veszi fel.) Annak a jelölése, hogy X (diszkrét) valószínűségi változó c értéket vesz fel:

$$\Pr(X = c).$$

Kis valószínűségyszámítás...

Egy (diszkrét) valószínűségi változó *várható értéke* durván szólva a "hosszan" ismételt kísérletek alatt felvett értékeinek átlaga. Pontosabban:

$$x_1 \cdot p_1 + x_2 \cdot p_2 + \dots + x_n \cdot p_n,$$

ahol x_1, x_2, \dots, x_n az értékek, és p_1, p_2, \dots, p_n a hozzájuk tartozó valószínűségek. (Az előbbi példánál ez az érték $1 \cdot \frac{1}{6} + \dots + 6 \cdot \frac{1}{6} = 3.5$ lesz, mert minden egyes érték "hosszú távon" annyiszor fog megjelenni "arányaiban", mint amennyi az érték valószínűsége.)

Feladat 4

A legtöbb hálózatban az adatkapcsolati réteg úgy kezeli az átviteli hibákat egy linken, hogy a hibás vagy elveszett frame-et újraküldi. Ha annak a valószínűsége, hogy egy frame hibás vagy elveszett p , **ennyi az átviteli kísérletek (küldések) számának várható értéke egy frame sikeres küldéséhez** (ha feltesszük, hogy a küldő minden sikertelen küldésről értesül)?

Feladat 5

- Tegyük fel, hogy van egy bernáthegyi kutyanck, Bundás, amelyet arra képeztünk ki, hogy pálinkásüveg helyett egy dobozt vigyen a nyakában, amelyben három pendrive-ot helyeztünk el. Minden egyes pendrive 64 GB kapacitású. A kutya 18 km/h-s sebességgel odamehet hozzánk, bárhol is tartózkodik éppen. Milyen távolságtartományban van Bundásnak nagyobb adatátviteli sebessége, mint egy 10 Gb/s-os vonalnak (adminisztrációs többlet nélkül)?

JSON, subprocess

PYTHON ALAPOK II.

Subprocess hívások és shell parancsok

Ha nem érdekes az output:

```
import subprocess
subprocess.call(['df', '-h']) # új verziókban run(...)
```

Ha érdekes az output:

```
import subprocess
p = subprocess.Popen(["echo", "hello world"], stdout=subprocess.PIPE)

print(p.communicate()) # eredménye egy tuple (stdout, stderr)

# ('hello world', None)
```

Néha a `shell=True` argumentum is kell, nézd meg a doksit!!!

Hasznos segédletek:

<https://docs.python.org/3/library/subprocess.html>

<https://www.pythonforbeginners.com/os/subprocess-for-system-administrators>

subprocess – PIPE kezelés

Elvárt kimenet: dmesg | grep hda

```
from subprocess import PIPE, Popen

p1 = Popen(["dmesg"], stdout=PIPE)
p2 = Popen(["grep", "hda"], stdin=p1.stdout, stdout=PIPE)

p1.stdout.close() # Allow p1 to receive a SIGPIPE if p2 exits.

output = p2.communicate()[0]
```

subprocess – várakozás a process végére

A process állapotának lekérdezése: poll

```
from subprocess import PIPE, Popen
import time

p1 = Popen(["ping", '-n', '20', 'berkeley.edu'], stdout=PIPE)

while p1.poll() == None:
    print(" még fut " )
    time.sleep(1)
```

A process végének megvárása: wait – a communicate is megvárja a végét...

```
p1 = Popen(["ping", '-n', '20', 'berkeley.edu'], stdout=PIPE)

p1.wait() # várakozás a végére
```

traceroute, ping

HÁLÓZATI ESZKÖZÖK I.

traceroute (linux) – tracert (windows)

Cél a hálózati útvonal meghatározása egy célállomás felé!

Linuxon

```
lakis@dpgk-pktgen:~$ traceroute berkeley.edu
traceroute to berkeley.edu (35.163.72.93), 30 hops max, 60 byte packets
 1 192.168.0.192 (192.168.0.192) 0.292 ms 0.344 ms 0.390 ms
 2 ikoktatok-gate.inf.elte.hu (157.181.167.254) 1.251 ms 1.250 ms 1.265 ms
 3 taurus.centaur-taurus.elte.hu (157.181.126.134) 5.180 ms 5.267 ms 5.325 ms
 4 fw1.firewall.elte.hu (157.181.141.145) 1.271 ms 1.358 ms 1.299 ms
 5 taurus.fw1.fw.backbone.elte.hu (192.153.18.146) 5.626 ms 5.356 ms 5.395 ms
 6 rtr.hbone-elte.elte.hu (157.181.141.9) 2.229 ms 1.245 ms 1.749 ms
 7 tg0-0-0-14.rtr2.vh.hbone.hu (195.111.100.47) 2.377 ms 2.415 ms 2.407 ms
 8 be1.rtr1.vh.hbone.hu (195.111.96.56) 1.945 ms 1.642 ms 1.877 ms
 9 bpt-b4-link.net (80.239.195.56) 1.626 ms 1.581 ms 1.097 ms
10 win-bb2-link.tetelia.net (62.115.143.116) 196.574 ms win-bb2-link.telvia.net (213.155.137.38) 196.993 ms win-bb2-link.telvia.net (213.155.135.222) 180.071 ms
11 ffm-bb4-link.telvia.net (62.115.133.79) 199.425 ms 199.232 ms *
12 * * *
13 prs-bb3-link.telvia.net (62.115.137.114) 180.494 ms 179.986 ms *
14 sjo-b21-link.telvia.net (62.115.119.229) 197.252 ms 197.249 ms 197.264 ms
15 * a100row-ic-300117-sjo-b21.c.telvia.net (213.248.87.118) 196.555 ms *
16 nyk-bb4-link.telvia.net (62.115.142.222) 180.081 ms 54.240.242.148 (54.240.242.148) 200.986 ms 54.240.242.88 (54.240.242.88) 201.877 ms
17 54.240.242.161 (54.240.242.161) 200.935 ms * *
18 * * *
19 * * *
```

traceroute (linux) – tracert (windows)

Cél a hálózati útvonal meghatározása egy célállomás felé!

```
tracert -h 50 jhu.edu
Tracing route to jhu.edu [128.220.192.230]
over a maximum of 50 hops:

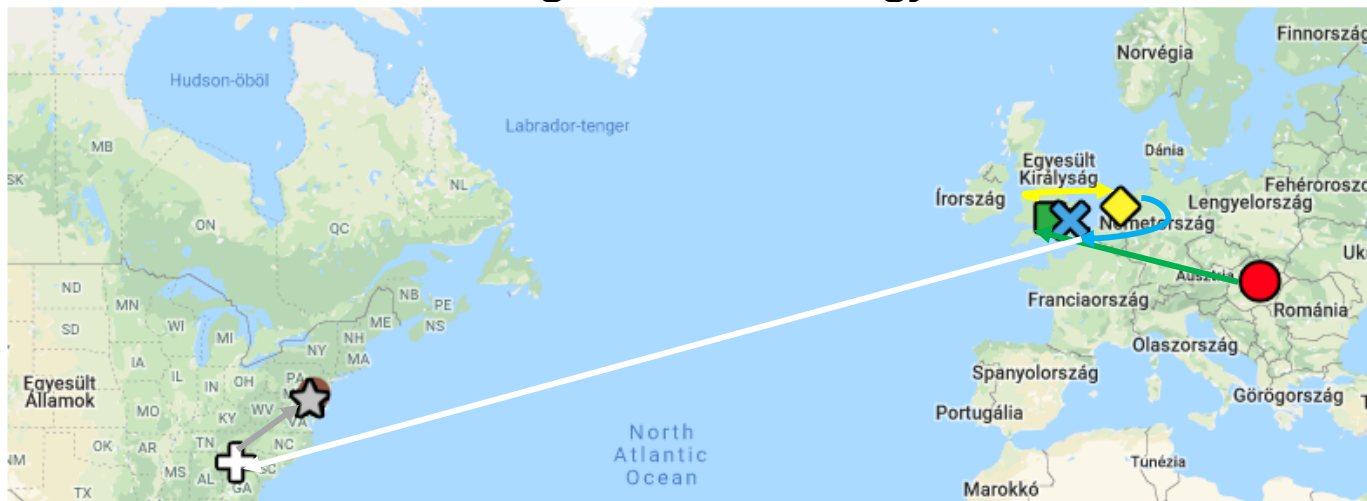
 1  <1 ms  <1 ms  <1 ms  ikoktatok-gate.inf.elte.hu [157.181.167.254]
 2  <1 ms  <1 ms  <1 ms  leo.leo-centaur.elte.hu [157.181.126.66]
 3  <1 ms  <1 ms  <1 ms  taurus.taurus-leo.elte.hu [157.181.126.45]
 4  <1 ms  <1 ms  <1 ms  fw1.firewall.elte.hu [157.181.141.145]
 5  <1 ms   1 ms  <1 ms  taurus.fw1.fw.backbone.elte.hu [192.153.18.146]
 6   1 ms  <1 ms   1 ms  rtr.hbone-elte.backbone.elte.hu [157.181.141.9]
 7   2 ms   1 ms   2 ms  tg0-0-0-14.rtr2.vh.hbone.hu [195.111.100.47]
 8   1 ms   1 ms   3 ms  be1.rtr1.vh.hbone.hu [195.111.96.56]
 9   1 ms   1 ms   1 ms  hungarnet.mx1.bud.hu.geant.net [62.40.124.101]
10  14 ms  19 ms  18 ms  62.40.98.47
11  21 ms  20 ms  21 ms  ae7.mx1.ams.nl.geant.net [62.40.98.186]
12  28 ms  28 ms  28 ms  ae9-mx1.lon.uk.geant.net [62.40.98.129]
13 103 ms 103 ms 104 ms  internet2-gw.mx1.lon.uk.geant.net [62.40.124.45]
14 104 ms 104 ms 104 ms  ae-0.4079.rtsw2.ashb.net.internet2.edu [162.252.70.137]
15 103 ms 103 ms 104 ms  ae-2.4079.rtsw.ashb.net.internet2.edu [162.252.70.74]
16 105 ms 104 ms 104 ms  et-11-3-0-1275.clpk-core.maxgigapop.net [206.196.177.2]
17 107 ms 106 ms 106 ms  hopkins-i2-rtr.maxgigapop.net [206.196.177.70]
18 106 ms 106 ms 106 ms  128.220.255.73
19  *      *      *      Request timed out.
20  *      *      *      Request timed out.
21  *      *      *      Request timed out.
22  *      *      *      Request timed out.
23 106 ms 106 ms 106 ms  boxmigration.jh.edu [128.220.192.230]








Trace complete.
```

Windowson

traceroute (linux) – tracert (windows)

Cél a hálózati útvonal meghatározása egy célállomás felé!



Legend	
	hungarnet.mx1.bud.hu.geant.net [62.40.124.101]
	62.40.98.47
	ae7.mx1.ams.nl.geant.net [62.40.98.186]
	ae9-mx1.lon.uk.geant.net [62.40.98.129]
	ae-0.4079.rtsw2.ashb.net.internet2.edu [162.252.70.137]
	et-11-3-0-1275.clpk-core.maxgigapop.net [206.196.177.2]
	boxmigration.jh.edu [128.220.192.230]

www.iplocation.net
+
www.copypastemap.com

Ping a hoszt elérhetőségének ellenőrzésére és a Round Trip Time (RTT) méréséhez

Linuxon

```
lakis@dppdk-pktgen:~$ ping -c 3 berkeley.edu
PING berkeley.edu (35.163.72.93) 56(84) bytes of data.
64 bytes from ec2-35-163-72-93.us-west-2.compute.amazonaws.com (35.163.72.93): icmp_seq=1 ttl=23 time=194 ms
64 bytes from ec2-35-163-72-93.us-west-2.compute.amazonaws.com (35.163.72.93): icmp_seq=2 ttl=23 time=194 ms
64 bytes from ec2-35-163-72-93.us-west-2.compute.amazonaws.com (35.163.72.93): icmp_seq=3 ttl=23 time=193 ms

--- berkeley.edu ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2002ms
rtt min/avg/max/mdev = 193.093/193.937/194.428/0.786 ms
```

Ping a hoszt elérhetőségének ellenőrzésére és a Round Trip Time (RTT) méréséhez

Windowson

```
C:\Users\laki>ping -n 3 berkeley.edu
```

```
Pinging berkeley.edu [35.163.72.93] with 32 bytes of data:
```

```
Reply from 35.163.72.93: bytes=32 time=200ms TTL=39
```

```
Reply from 35.163.72.93: bytes=32 time=201ms TTL=39
```

```
Reply from 35.163.72.93: bytes=32 time=200ms TTL=39
```

```
Ping statistics for 35.163.72.93:
```

```
    Packets: Sent = 3, Received = 3, Lost = 0 (0% loss),
```

```
Approximate round trip times in milli-seconds:
```

```
    Minimum = 200ms, Maximum = 201ms, Average = 200ms
```

Fontos információ

- A házi feladatokat Python3-ban kell megírni, mert a BE-AD rendszeren keresztül elérhető tesztkörnyezet is abban van!

HÁZI FELADAT I. (1 PONT)

Alexa-top-1M

Az Alexa-top-1M adathalmaz tartalmazza a legnépszerűbb 1 millió website domain nevét népszerűségi sorrendben:

<http://s3.amazonaws.com/alexastatic/top-1m.csv.zip>

Válasszuk ki az első és utolsó 10 nevet a listából, írjunk egy python programot, ami végig megy a leszűkített 20 elemű listán és minden címre lefuttatja a traceroute és ping toolokat, majd az eredményeket rendezett formában két fájlba írja! Ld. subprocess!!! Lehetőség szerint ne az egyetemi hálózaton futassuk az adatbegyűjtést!

Traceroute paraméterek: max. 30 hopot vizsgáljunk

Ping paraméterek: 10 próba legyen

Program paraméterezése:

`./programom.py <top-1m.csv elérési útja>`

Kimeneti fájlok (ld. következő dia):

`traceroute.json`

`ping.json`

A párhuzamos futtatás esetén vigyázzunk és limitáljuk a processek maximális számát!!!

BE-AD rendszer használata

Határidő: 2020-10-04 23:59

traceroute.json

```
traceroute.json:
{
    "date" : "20180916",
    "system" : "windows",
    "traces" : [
        {
            "target" : "www.valami.com",
            "output" : "Tracing route to www. . . "
        },
        ...
    ]
}
```

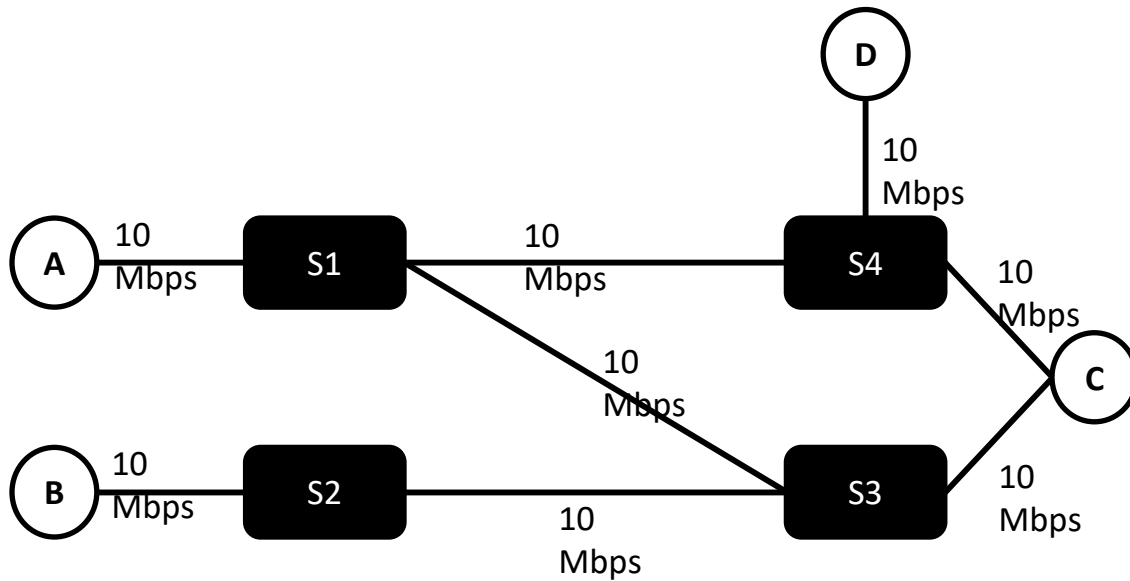
ping.json

```
ping.json:
{
    "date" : "20180916",
    "system" : "linux",
    "pings" : [
        {
            "target" : "www.valami.com",
            "output" : „Pinging www. . . "
        },
        ...
    ]
}
```

Áramkörkapcsolt hálózatok

HÁZI FELADAT II. (1 PONT)

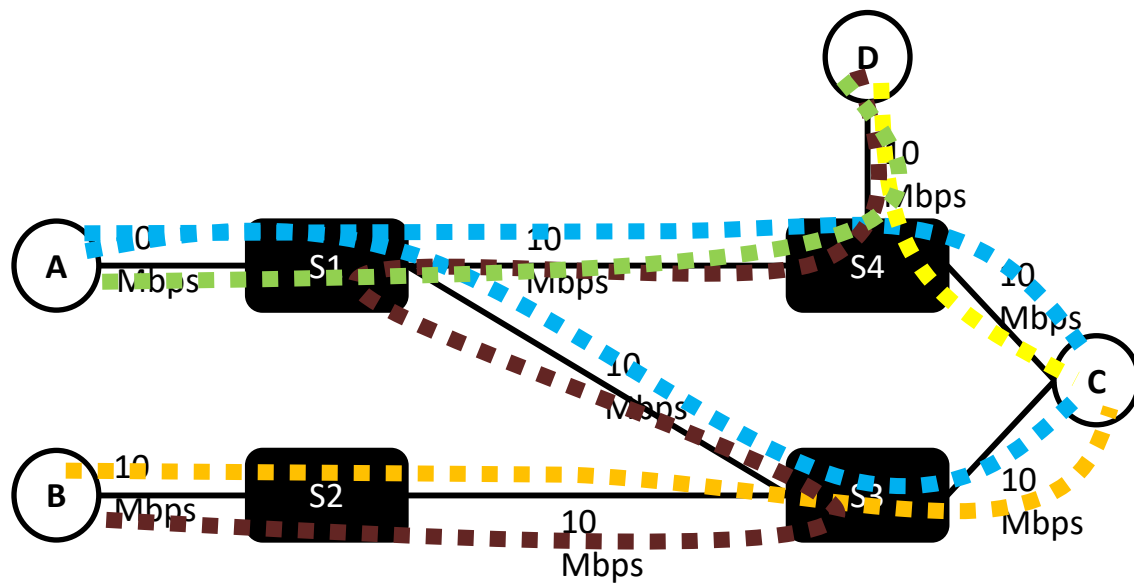
Topológia – cs1.json



Írányítatlan legyen a gráf!!!

```
{
  "end-points": [ "A", "B", "C", "D" ],
  "switches": [ "S1", "S2", "S3", "S4" ],
  "links" : [
    {
      "points" : [ "A", "S1" ],
      "capacity" : 10.0
    },
    {
      "points" : [ "B", "S2" ],
      "capacity" : 10.0
    },
    {
      "points" : [ "D", "S4" ],
      "capacity" : 10.0
    },
    {
      "points" : [ "S1", "S4" ],
      "capacity" : 10.0
    },
    {
      "points" : [ "S1", "S3" ],
      "capacity" : 10.0
    },
    {
      "points" : [ "S2", "S3" ],
      "capacity" : 10.0
    },
    {
      "points" : [ "S4", "C" ],
      "capacity" : 10.0
    },
    {
      "points" : [ "S3", "C" ],
      "capacity" : 10.0
    }
  ]
}
```

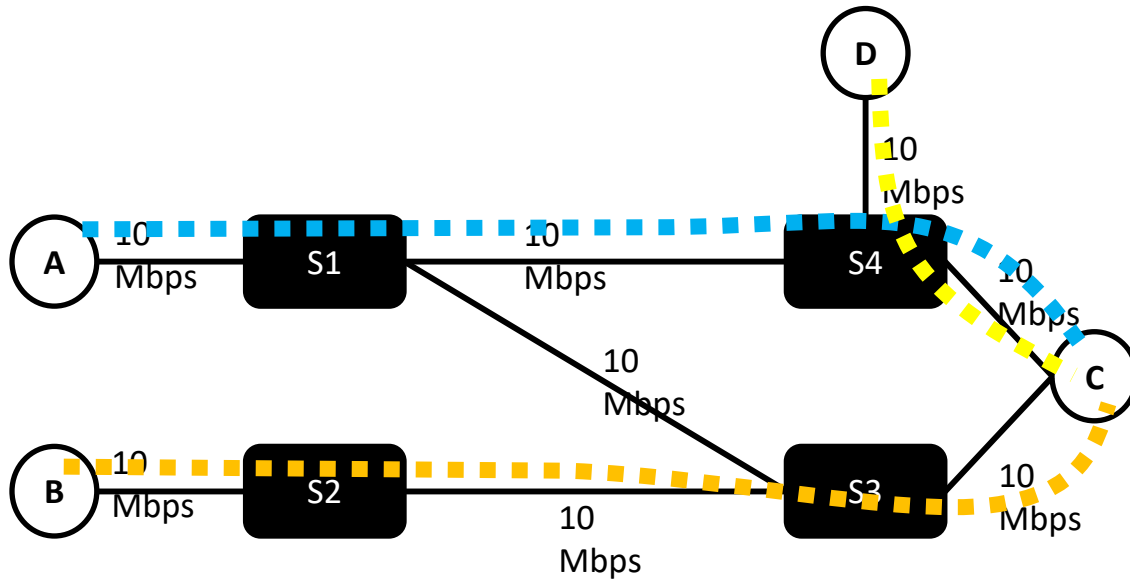
Lehetséges áramkörök – cs1.json



Írányítatlan legyen a gráf!!!

```
"possible-circuits" : [
  [ "D", "S4", "C", "S3", "S1", "A", "B", "S2", "S3", "S4", "D" ],
  [ "A", "S1", "S4", "C", "S3", "S1", "A", "B", "S2", "S3", "S4", "D" ],
  [ "A", "S1", "S3", "C", "S3", "S1", "A", "B", "S2", "S3", "S4", "D" ],
  [ "C", "S4", "S1", "A", "B", "S2", "S3", "S4", "D", "S3", "S1", "A" ],
  [ "C", "S4", "S1", "A", "B", "S2", "S3", "S4", "D", "S3", "S1", "A" ],
  [ "B", "S2", "S3", "C", "S3", "S1", "A", "B", "S2", "S3", "S4", "D" ],
  [ "C", "S4", "S1", "A", "B", "S2", "S3", "S4", "D", "S3", "S1", "A" ],
  [ "B", "S2", "S3", "S1", "A", "B", "S2", "S3", "S4", "D", "S3", "S1", "A" ],
  [ "A", "S1", "S3", "S2", "S3", "S4", "D", "S3", "S1", "A", "B", "S2", "S3", "S4", "D" ],
  [ "B", "S2", "S3", "S1", "S4", "D", "S3", "S1", "A", "B", "S2", "S3", "S4", "D" ],
  [ "D", "S4", "S1", "A" ]
]
```

Igények – cs1.json



Irányítatlan legyen a gráf!!!

```
"simulation": {  
  "duration": 11,  
  "demands": [  
    {  
      "start-time": 1,  
      "end-time": 5,  
      "end-points": ["A", "C"],  
      "demand": 10.0  
    },  
    {  
      "start-time": 2,  
      "end-time": 10,  
      "end-points": ["B", "C"],  
      "demand": 10.0  
    },  
    {  
      "start-time": 6,  
      "end-time": 10,  
      "end-points": ["D", "C"],  
      "demand": 10.0  
    }  
  ]  
}
```

Feladat

Adott a cs1.json, ami tartalmazza egy irányítatlan gráf leírását. A gráf végpont (end-points) és switch (switches) csomópontokat tartalmaz. Az élek (links) kapacitással rendelkeznek (valós szám). Tegyük fel, hogy egy áramkörkapcsolt hálózatban vagyunk és valamilyen RRP-szerű erőforrás foglalo protokollt használunk. Feltesszük, hogy csak a linkek megosztandó és szűk erőforrások. A json tartalmazza a kialakítható lehetséges útvonalakat (possible-cicuits), továbbá a rendszerbe beérkező, két végpontot összekötő áramkörigényeket kezdő és vég időponttal. A szimuláció a t=1 időpillanatban kezdődik és t=duration időpillanatban ér véget.

Készíts programot, ami leszimulálja az erőforrások lefoglalását és felszabadítását a JSON fájlban megadott topológia, kapacitások és igények alapján!

A program bemenete: cs1.json (**első parancssori argumentum**)

A program kimenete: Minden igény lefoglalását és felszabadítását írassuk ki a stdout-ra. Foglálás esetén jelezzük, hogy sikeres vagy sikertelen volt-e. Megj.: sikertelen esetben az igényrel más teendők nincs, azt eldobhatjuk.

Pl.:

1. igény foglálás: A<->C st:1 - sikeres
2. igény foglálás: B<->C st:2 - sikeres
3. igény felszabadítás: A<->C st:5
4. igény foglálás: D<->C st:6 - sikeres
5. igény foglálás: A<->C st:7 - sikertelen
- ...

BE-AD rendszer használata
Határidő: 2020-10-04 23:59

VÉGE
KÖSZÖNÖM A FIGYELMET!