Porazdeljeni sistemi

Izpit, 6. febuar 2025, ob 10:00 v P3

1. Navedite vrste sistemov s porazdeljenim pomnilnikom in podajte njihove glavne značilnosti.

   https://github.com/laspp/PS-2024/blob/main/predavanja/02-arhitekture/arhitekture.md#sistemi-s-porazdeljenim-pomnilnikom

2. Kako je logika modela opravilo kanal izvedena v jeziku go?

   https://github.com/laspp/PS-2024/blob/main/predavanja/05-go/go.md#podpora-za-sočasnost-cg2

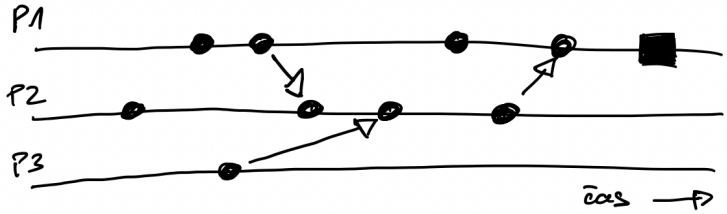
3. Opišite bralno-pisalne ključavnice. S katerimi sinhronizacijskimi elementi jih lahko nadomestimo?

   https://github.com/laspp/PS-2024/blob/main/predavanja/07-sinhronizacija-2/sinhronizacija-2.md#bralno-pisalne-ključavnice-smap910, primera pisatelji-bralci-4.go in pisatelji-bralci-5.go

4. Napišite program v jeziku go, v katerem glavna gorutina zažene *dve* dodatni gorutini. Vsaka dodatna gorutina 100.000 krat za 1 poveča števec v skupnem pomnilniku. Števec je ob začetku izvajanja nastavljen na 0. Poskrbite, da bo končna vrednost števca 200.000 tudi kadar se gorutini izvajata vzporedno. Za sinhronizacijo gorutin uporabite kanale.

   Ščitenje kritičnega odseka, podobno kot v kodi:
   https://github.com/laspp/PS-2024/blob/main/predavanja/06-sinhronizacija-1/koda/filozofi-6.go in čakanje v glavni gorutinu, da vse dodatne zaključijo, na enak način kot v kodi:
   https://github.com/laspp/PS-2024/blob/main/predavanja/06-sinhronizacija-1/koda/pi-10.go

5. Za procese P1, P2 in P3 beležimo interne dogodke in pošiljanja sporočil od začetka izvajanja. Dogodki so označeni s črnimi pikami in črnim kvadratom, pošiljanje sporočil pa s puščicami. Določite vektorsko uro za dogodek označen s črnim kvadratom.

   [5, 4 , 1]

6. Opišite vzorec oddaljenjega klicanja metod (angl. Remote Procedure Call, RPC).

   https://github.com/laspp/PS-2024/blob/main/predavanja/11-posredovanje-sporocil-2/posredovanje-sporocil-2.md#vzorec-rpc

7. Na sistemu za verižno replikacijo želimo podpreti branje iz shrambe preko vseh vozlišč v verigi. Predlagajte rešitev, ki vključuje vpisovanje podatkov v shrambo in branje iz nje.

   https://github.com/laspp/PS-2024/blob/main/predavanja/15-replikacija-1/replikacija-1.md#verižna-replikacija-uds104 (optimizacija)

Porazdeljeni sistemi

Izpit, 6. febuar 2025, ob 10:00 v P3

8. V fazi izbiranja voditelja v algoritmu raft proces prehaja med več stanji. Narišite končni avtomat, na katerem označite stanja in prehode med njimi. Stanja in prehode na kratko komentirajte.

   https://github.com/laspp/PS-2024/blob/main/predavanja/16-replikacija-2/slike/raft-avtomat.png

9. Narišite shemo arhitekure grafičnih pospeševalnikov.

   https://github.com/laspp/PS-2024/blob/main/predavanja/19-gpe/slike/naprava-zgradba.png

10. Popravite spodnji ščepec, da bo pravilno računal skalarni produkt vektorjev `a` in `b` dolžine `n`. Vektorja in skupni pomnilnik `smem` so pravilno alocirani, `dotPtr` je kazalec na spremenljivko s skalarnim produktom. Število niti v bloku je potenca števila 2. V zanki for izvedemo redukcijo po drevesu.

```
__global__ void dotProduct(float *dotPtr, float *a, float *b, int n) {
   extern __shared__ float smem[];
   smem[threadIdx.x] = 0.0;
   int gid = threadIdx.x *+ blockDim.x +* blockIdx.x;
   while (gid < lenn) {
      smem[threadIdx.x] += a[gid] * b[gid];
      gid += blockDim.x * gridDim.x;
   }
   __syncthreads();
   for(int step = blockDim.x / 2; step > 0; idxStepstep /= 2) {
      if (threadIdx.x < step) {
         smem[threadIdx.x] += smem[threadIdx.x + step];
           syncthreads();
      }
      __syncthreads();
   }
   if (threadIdx.x == 0)
      atomicAdd(dotPtr, smem[0]);
}
```