Svar till övning 6 i Dator- och telekommunikation

När ett system består av flera delar som fungerar på (ungefär) samma sätt så är ofta processimulering bra att använda. Börja med att svara på följande frågor innan du börjar skriva simuleringsprogrammet:

- Vilka processer behövs?
- Vilka är processernas interna tillståndsvariabler?
- Vilka signaler behövs?
- Vad ska en process göra när den får en viss signal?

Uppgift 1

Antag att du har ett system som består av tre buffertar och en betjänare. Betjänaren börjar med att kolla om det finns jobb i buffert 1. Om så är fallet så betjänas ett jobb därifrån. Om det inte finns något jobb där så väntar betjänaren en kort tid, 0.1 s och därefter kollar betjänaren om det finns någon kund i buffert 2 etc. När är klar med buffert 3 så börjar man om med buffert 1 igen.

a) Vilka processer kan vara lämpliga för att simulera detta system?
   Generator, Buffer, Server

b) Vilka signaler ska kunna skickas mellan processerna?
   ready, job, anyJobs, yes, no

c) Beskriv med pseudokod vad en process ska göra när den får en signal.
   För Generator:

```
Om mottagen signal == ready
   sendSignal(ready, this, time + timeToNextCustomer());
   sendSignal(job, sendTo, time);
```

   Vi antar att det finns en metod timeToNextCustomer() som ger tiden till nästa kund som ska genereras och att sendTo är en referens till den buffert som generatorn ska skicka jobb till. Det finns en generator för varje buffert.

   För Buffer:

```
Om mottagen signal == job
   numberInBuffer++;
Om mottagen signal == anyJobs
   if (numberInBuffer == 0)
      sendSignal(no, server, time);
   else {
      sendSignal(yes, server, time);
      numberInBuffer--;}
```

   Här antar vi att det finns en referens till betjänaren (server) och en variable numberOfServer som håller reda på antalet kunde i bufferten.

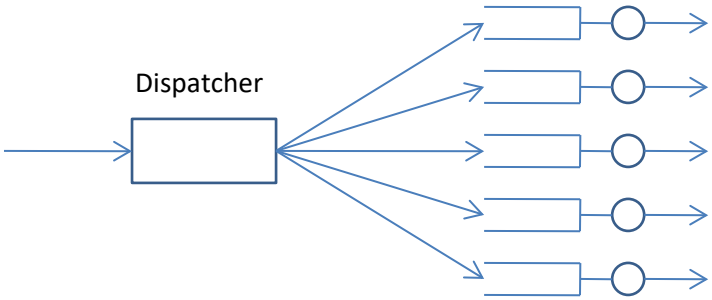
För Server:

```
Om mottagen signal == ready
   actBuffer = nextBuffer();
   sendSignal(anyJobs, actBuffer, time);
Om mottagen signal == yes
   sendSignal(ready, this, time + serviceTime());
Om mottagen signal == no
   sendSignal(ready, this, time + 0.1);
```

Här förutsätter vi att det finns en metod som ger oss betjäningstiderna (serviceTime()) och en som ger oss nästa buffert som ska behandlas (nexteBuffer()). Variabeln actBuffer är en referens till den buffert som håller på att behandlas.

Uppgift 2

Vi ska studera lastdelning, vilket används i många sammanhang, t ex webbservers, datanät och inom logistik. Det finns ett antal kösystem och en fördelare som helt enkelt ska fördela jobb mellan köerna. Fördelaren väljer vilket kösystem som en ankommande kund ska skickas till. Se figuren:



Det finns fem kösystem. Fördelaren kan använda någon av följande algoritmer för att bestämma till vilket kösystem som den ska skicka ett nytt jobb:

i. Den väljer ett system på slump
ii. Den väljer först system 1, sedan 2, sedan 3, sedan 4, sedan 5, sedan 1 igen etc.
iii. Den väljer det kösystem som har minst antal jobb

Lös följande uppgifter:

a) Vilka processer är lämpliga?
b) Vilka signaler?
c) Vad ska en process göra när den får en signal?

Denna uppgift finns löst i ett program som kan hämtas från kursens hemsida strax nedanför detta dokument.

Uppgift 3

Antag att vi har ett kösystem med en oändlig buffert och en betjänare. Vi antar att ankomstintensiteten är λ och medelbetjäningstiden $\bar{x}$. För alla värden på λ,

a) Beräkna sannolikheten att betjänaren är upptagen

$$P(\text{upptagen}) = \begin{cases} \lambda\bar{x} \text{ om } \lambda < \frac{1}{\bar{x}} \\ 1 \text{ om } \lambda \geq \frac{1}{\bar{x}} \end{cases}$$

b) Beräkna medelantal kunder i betjänaren

Samma svar som i a-uppgiften.

c) Beräkna intensiteten med vilken kunder blir färdigbetjänade

$$\text{Om } P(\text{upptagen}) = \begin{cases} \lambda \text{ om } \lambda < \frac{1}{\bar{x}} \\ \frac{1}{\bar{x}} \text{ om } \lambda \geq \frac{1}{\bar{x}} \end{cases}$$

Uppgift 4

Det är ofta av intresse att optimera olika slags system (inte helt förvånande). Vilket tror du är bäst, att ha två betjänare med maximala betjäningsintensiteten 1 per sekund eller en med maximala betjäningsintensiteten 2 per sekund? Vad bör man mena med bäst i detta fall?

Rimligen menar man att de bästa är den som ger kortast medeltid i systemet. I så fall är den bättre att ha en snabb betjänare i stället för två hälften så snabba. Om en kund kommer till ett tomt system blir den kunden betjänad dubbelt så snabbt om man har en snabb betjänare i stället för två hälften så snabba.

Uppgift 5

En simulering är ju ett experiment som man utför i en dator. Syftet är ofta att bestämma en storhet som man inte kan räkna ut. Ofta vill man få en uppfattning om hur noggrann en simulering är, det vill säga hur nära man kommer den okända storheten. Kan du fundera ut något sätt att försöka avgöra om en simulering är noggrann eller inte, utan avancerad matematik.

Det enklaste är att köra simuleringen flera gånger med olika slumptal och se hur mycket resultaten skiljer sig åt. Stora skillnader tyder på att simuleringen inte är så noggrann.