Proiect tehnici de simulare

Ştefan Iuga Iancu Petcu Ştefan Pogonaru

May 22, 2020

1 Descrierea problemei

Propunem analiza activităţii unei firme ce oferă poliţe de asigurare pentru echipamente electronice.

Scopul simulării este studierea eficienţei de vânzare a acestei poliţe în decursul a T de zile.

2 Obiectivele simulării

1. Estimarea probabilităţii ca firma să nu fie ruinată până la momementul T.
2. Estimarea unui capital minim necesar ca probabilitatea estimata de ruină până la momentul T să fie mai mică de 80%.

3 Convenţii

- Pornim cu un număr iniţial de n_0 clienţi şi un capital a_0.
- Clienţii fac cereri de despăgubire conform unui proces Poisson omogen de rată λ.
- Valoarea fiecărei despăgubiri solicitate este o variabilă aleatoare cu funcţia de repartiţie F.
- Potenţialii noi clienţi ai firmei semnează contracte de asigurare conform unui proces Poisson omogen de rată ν.
- Actualii clienţi ai firmei rămân fideli firmei pentru un timp aleator ce corespunde repartiţiei exponenţiale de parametru μ.
- Toţi clienţii firmei plătesc o sumă fixă c per unitate de timp.

4 Datele problemei

- $\lambda = 8/zi$
- $F(x) = \begin{cases} 0 & x < 0 \\ \frac{x^2+x}{2} & x \in [0,1) \\ 1 & x \geq 1 \end{cases}$
- $\nu = 18/zi$
- $\mu = 0.2(zile)$
- $c = 2€/zi$
- $T = 365zile$
- $a_0 = 50000€$
- $n_0 = 12$

5 Cod sursa

```
# Vom folosi paralelism pentru a calcula simularile in paralel
library(parallel)

# Clusterul primeste numarul de core-uri al masinii pe care ruleaza
cl <- makeCluster(detectCores())

# Evaluam variabilele si funtiile si le facem disponibile pentru
    cluster
clusterEvalQ(cl, {

  # Functie de repartitie pentru variabila aleatoare ce determina
    valoarea
  # fiecarei despagubiri
  F <- function(x) {
    if (x < 0)
      return (0)

    if (x < 1)
      return ((x ^ 2 + x) / 2)
    return (1)
  }

  # Rata procesului Poisson pentru cereri de despagubire
  lam <- 8 #/zi

  # Rata procesului Poisson pentru noi clienti
  nu <- 18 #/zi

  # Parametru pentru repartitia exponentiala ce determina timpul
  # pentru care clientii raman fideli firmei
  mu <- 0.2 #zile
```

```
30
31  # Suma platita de clienti per unitate de timp
32  c <- 2 #euro/zi
33
34  # Durata de timp pe care se face analiza
35  T <- 365 #zile
36
37  # Capital de inceput
38  a0 <- 50000 #euro
39
40  # Numarul initial de clienti
41  n0 <- 12 #clienti
42
43  # Probabilitatea aparitiei unui client nou
44  pClientNou <- function(nu, n, mu, lam) {
45    return (nu / (nu + n * mu + n * lam))
46  }
47
48  # Probabilitatea pierderii unui client
49  pPierdereClient <- function(nu, n, mu, lam) {
50    return ((n * mu) / (nu + n * mu + n * lam))
51  }
52
53  # Probabilitatea cererii unei despagubiri
54  pDespagubire  <- function(nu, n, mu, lam) {
55    return ((n * lam) / (nu + n * mu + n * lam))
56  }
57
58  # X ~ Exp
59  getXfromExp  <- function(l) {
60    return (-log(runif(1),exp(1))/l)
61  }
62
63  # Definirea unei simulari primeste ca parametru a0 capital
      initial
64  simulare <- function(a0) {
65    # t = timp de pornire
66    t <- 0
67
68    # a = capitalul
69    a <- a0
70
71    # n = nr de clienti
72    n <- n0
73
74    # X = peste cat timp va avea loc urmatorul eveniment
75    X <- getXfromExp(nu + n * mu + n * lam)
76
77    # tE = timpul urmatorului eveniment
78    tE <- X
79
80    repeat {
81      # Daca timpul urmatorului eveniment depaseste marja de timp
      pentru
82      # care se face simularea (T) inseamna ca firma nu a dat
      faliment
83      # in aceasta perioada de timp
```

```
      if (tE > T) {
        return(1)
      }

      # Adaugam la capital suma pe unitate de timp (c) * timpul de la ultimul
      # eveniment
      a <-  a + n * c * (tE - t)

      # t ia acum timpul evenimentului urmator
      t <- tE

      # Generam o variabila uniforma pentru a vedea ce
      # fel de eveniment are loc
      U <- runif(1)

      # Apare un client nou si incrementam nr de clienti (n) cu 1
      if (U < pClientNou(nu, n, mu, lam)) {
        n <- n + 1

      # Pierdem un client iar numarul de clienti (n) este decrementat
      } else if (U < (pClientNou(nu, n, mu, lam) +
          pPierdereClient(nu, n, mu, lam))) {
        n <- n - 1

      # Are loc o cerere de despagubire
      } else {
  # Generam Y valoarea sumei de despagubire
        U1 <- runif(1)
        Y <- F(U1)

  # Daca suma depaseste capitalul actual, dam faliment
        if (Y > a) {
          return (0)

  # Altfel scadem Y din capital
        } else {
          a <- a - Y
        }
      }

      # Aflam timpul urmatorului eveniment
      X <- getXfromExp(nu + n * mu + n * lam)
      tE <- t + X
    }
  }

})

# Functie ce simuleaza un numar de simulari (nrSim) si calculeaza media lor
calcProcent <- function(a){
  a0 <- a
  nrSim <- 10
  i <- 1
  S <- 0
```

```
  while (i <= nrSim) {
    S <-  S + simulare(a0)
    i <- i + 1
  }
  pr <- S/nrSim
  result <- list("Capital"=a0,"Procent"= pr)
  return (result)
}

# Am observat din rulari anterioare ca suma minima pentru a nu da faliment
# se afla intre 40000 si 50000, astfel rulam in paralel simulari pentru
# toate sumele aflate intre aceste numere
rezultate <- parLapply(cl,seq(40000,50000,500),calcProcent)

# Oprim cluster-ul
stopCluster(cl)

# Probabilitatea de faliment cautata
proc <- 0.2
for(i in 1:length(rezultate)){
  if(rezultate[[i]]$Procent>=proc){
    print(rezultate[[i]])
    break
  }
}
```

6 Rezultate

Am obţinut o probabilitate 0 de faliment cu capitalul iniţial 50000€.

Capitalul necesar pentru ca rata de faliment să fie mai mică de 80% este 42500€.