Pszeudorandomszám generátorok összehasonlítása

Tóth Anna Anita

Kivonat

Miért kellenek nekünk a véletlen számok? A kriptográfia, ami először eszünkbe juthat. De emellett többek között algoritmusok futási idejének csökkentéséhez, geometriai problémák megoldásához is használjuk. Mivel igazi véletlen számok előállítása és használata sok nehézségbe ütközik, ezért a gyakorlatban generátorok segítségével hozunk létre őket és így már pszeudorandom számoknak nevezzük.

Kutatásom során R-beli pszeudorandomszám generátorok működésével ismerkedtem meg részletesen, amelyek a következők: Wichmann-Hill, Marsaglia Multicarry, Super-Duper, Mersenne-Twister és a Knuth-TAOCP generátor. Célom ezek összehasonlítása volt, amelyhez több módszert alkalmaztam. Először kiválasztottam a Dieharder tesztcsaládot, mellyel a következőket vizsgáltam: sebesség, periódushossz, és a tesztek kimenetelei. Az eredmények alapján a Mersenne-Twister generátort bizonyult a legjobbnak, majd a Knuth-TAOCP és a Wichmann-Hill, míg legrosszabbul a Super-Duper és a Marsaglia Multicarry teljesített. Majd három véletlen számokat használó algoritmust implementáltam R-ben, hogy ezek segítségével megerősítsem, vagy rácáfoljak a Dieharderrel kapott eredményekre. A gyorsrendezés algoritmussal kezdtem, amellyel a sebességüket mértem, és arra a következtetésre jutottam, hogy a Super-Duperrel éri el a legkisebb futásidőt, a Mersenne-Twister volt a leglassabb, és a többi a középmezőnyben végzett. Ez azon múlt, hogy egy generátorral azért gyorsabb az eredmény, mert jobb véletlen elemet választ, így kisebb a rekurzió mélysége; vagy gyorsan generálja a véletlen elemeket. Ezután a π közelítésére adtam egy Monte Carlo algoritmust, amellyel ugyanazt az eredményt kaptam, mint a Dieharderrel. Végül pedig a Box-Müller transzformációt használó algoritmussal készítettem az egyenletes eloszlású véletlen számokból standard normális eloszlásúakat. Itt a kimenet normalitását vizsgáltam az R-ben alapértelmezett *rnorm* paranccsal létrehozottakhoz képest. Ahol a kapottak alapján nem lehetett nagy eltéréseket kimutatni se az öt generátor, se a generátorok és az alapértelmezett parancs eredményei között.

Összességében arra a következtetésre jutottam, hogy általánosan a Mersenne-Twister generátor a legjobb. Viszont kisebb elemszámú, a generátor meghívását sokszor elvégző algoritmusok esetén jobb a Super-Dupert választani. Fontos megjegyezni, hogy nagyobb szimulációk esetén érdemes jól meggondolni melyik pszeudorandomszám generátort alkalmazzuk.