

СТАТИСТИЧЕСКИ ЗАМЕРВАНИЯ В ДИСКРЕТЕН ВЕРОЯТНОСТЕН МОДЕЛ НА ХАЗАРТНА ИГРА

Research described in this tutorial was partially supported by the National Scientific Program "Information and Communication Technologies for a Single Digital Market in Science, Education and Security (ICTinSES)", financed by the Ministry of Education and Science.

Developed by Velbazhd Software LLC

Първоначалният сериозен тласък за развитието на теорията на вероятностите идва от хазарта. И до ден днешен хазартната индустрия е един от основните потребители на изчисления в областта на вероятностите и статистиката. Основен дял в хазартната индустрия държат ротативките или известни още под названието „едноръки бандити“. В този вид хазартни игри дискретен вероятностен модел определя какви печалби ще достигне играчът.

Съвременните ротативки са електронни, най-често имат 5 барабана и се показват по 3 реда от всеки барабан. Този вид ротативки използват виртуални барабани на които са разположени различните печеливши символи. Най-често тази информация се представя под формата на двумерен масив и има смисъла на дискретно вероятностно разпределение.



Виртуалните барабани се завъртат, спират на случайни позиции и по редове (най-често от ляво на дясно) се формират печелившите комбинации. Някои печеливши символи се срещат по-често, други по-рядко. По-често срещаните символи носят по-малки печалби, докато по-рядко срещаните символи носят по-големи печалби. Точните стойности на печалбите са представени под формата на таблица с печалби.



Най-важният параметър, който се следи в една ротативка е така нареченото RTP (return to player). RTP се отчита в проценти и представлява общата печалба, разделена на общата загуба, умножена по 100. Един от най-често срещаните RTP проценти е 96%. Статистическият смисъл е, че в дългосрочен план от всеки 100 долара, при едно завъртане на барабаните, играчът ще си върне обратно 96 долара.

При сертифицирането на ротативките производителят е длъжен да представи аналитична формула за пресмятането на RTP процента. Паралелно на това, при моделирането на играта и при нейното тестване се извършват Монте Карло симулации за експериментално установяване на RTP процента и серия други статистически параметри.



Ротативките имат и втора характеристика, която не е от толкова голямо значение за регулаторните органи, но е от съществено значение за играчите, а именно волатилността. Ниско волатилните игри дават малки печалби, но ги дават често, докато високо волатилните игри дават големи печалби, но значително по-рядко.



С оглед на параметъра волатилност в индустрията ротативките се симулират при 100 милиона завъртания, така че волатилността да не изкривява статистиката за RTP процента.



Когато проектантите на ротативки правят модел на играта те се интересуват от множество други статистически замервания. Едно от тези замервания е с каква тежест участва всяка печеливша комбинация във формирането на печалбите. В общия случай печалбите се формират от 3, 4 или 5 последователни символи, подредени по предварително зададени в играта печеливши линии.



За извършването на симулациите често се използва програмен код, който не е част от самия хазартен продукт. В симулатора, наличен в GitHub, липсва точно тази функционалност, която да отчита с каква тежест участва всяка печеливша комбинация във формирането на общата печалба.





Search or jump to...

[Pull requests](#) [Issues](#) [Marketplace](#) [Explore](#)



[VelbazhdSoftwareLLC](#) / [Fruit-Machine-Simulator-with-Excel-Interface](#)

[Unwatch](#) 1

[Star](#) 5

[Fork](#) 1

[Code](#)

[Issues](#) 0

[Pull requests](#) 0

[Actions](#)

[Projects](#) 0

[Wiki](#)

[Security](#)

[Insights](#)

[Settings](#)

Fruit machine simulator with Excel input-output interface.

[Edit](#)

[gambling](#) [monte-carlo-simulation](#) [optimization](#) [slot-machine](#) [Manage topics](#)

[61 commits](#)

[1 branch](#)

[0 packages](#)

[1 release](#)

[1 contributor](#)

[GPL-3.0](#)

Branch: [master](#)

[New pull request](#)

[Create new file](#)

[Upload files](#)

[Find file](#)

[Clone or download](#)



TodorBalabanov Extra Stars rules were implemente.

✓ Latest commit e2e0235 on 5 Dec 2019

.settings	Gradle build script was created.	11 months ago
doc	Gradle build script was created.	11 months ago
gradle/wrapper	Gradle version was changed.	3 months ago
src	Extra Stars rules were implemente.	3 months ago
.classpath	Extra Stars rules were implemente.	3 months ago
.gitignore	Gradle build script was created.	11 months ago
.project	Gradle build script was created.	11 months ago
.travis.yml	Update .travis.yml	6 months ago
LICENSE	Initial commit	2 years ago
README.md	Build command was added.	7 months ago
build.gradle	Fix gradle build.	11 months ago
gradlew	Gradle build script was created.	11 months ago
gradlew.bat	Gradle build script was created.	11 months ago
settings.gradle	Gradle build script was created.	11 months ago

[README.md](#)



Fruit-Machine-Simulator-with-Excel-Interface [build](#) [passing](#)

Към симулатора вече има пресмятане на обобщена статистика за участието на печелившите комбинации в RTP процента. Получава се чрез разделяне на общата печалба от съответната комбинация върху общата загуба. По аналогичен начин може да се пресметне отношението на всяка печеливша комбинация в общата печалба като се раздели не на общата загуба, а на общата печалба.




```

1712         for (int left = 0, right = initialBin; right < freeMaxWin; left = right, right += right
1713             + binIncrement) {
1714             double sum = 0;
1715             for (int value = left; value < right; value++) {
1716                 if (freeWinsHistogram.containsKey(value) == false) {
1717                     continue;
1718                 }
1719                 sum += freeWinHistogram.get(value);
1720             }
1721             System.out.print(sum + "\t");
1722         }
1723     }
1724     System.out.println();
1725     System.out.println();
1726
1727     System.out.println("Base Game Symbols RTP:");
1728     System.out.print("\t");
1729     for (int i = 0; i < baseSymbolMoney.length; i++) {
1730         System.out.print(" " + i + " of\t");
1731     }
1732     System.out.println();
1733     for (int j = 0; j < baseSymbolMoney[0].length; j++) {
1734         System.out.print(SYMBOLS_NAMES.get(j) + "\t");
1735         for (int i = 0; i < baseSymbolMoney.length; i++) {
1736             System.out.print(
1737                 (double) baseSymbolMoney[i][j] / (double) lostMoney
1738                 + "\t");
1739         }
1740         System.out.println();
1741     }
1742     System.out.println();
1743     System.out.println("Base Game Symbols Hit Rate:");
1744     System.out.print("\t");
1745     for (int i = 0; i < baseGameSymbolsHitRate.length; i++) {
1746         System.out.print(" " + i + " of\t");
1747     }
1748     System.out.println();
1749     for (int j = 0; j < baseGameSymbolsHitRate[0].length; j++) {
1750         System.out.print(SYMBOLS_NAMES.get(j) + "\t");
1751         for (int i = 0; i < baseGameSymbolsHitRate.length; i++) {
1752             System.out.print((double) baseGameSymbolsHitRate[i][j] + "\t");
1753         }
1754         System.out.println();
1755     }
1756     System.out.println();
1757     System.out.println("Base Game Symbols Hit Frequency:");
1758     System.out.print("\t");
1759     for (int i = 0; i < baseGameSymbolsHitRate.length; i++) {
1760         System.out.print(" " + i + " of\t");
1761     }
1762     System.out.println();

```

Статистиката се натрупа по отделно за базовата игра и за безплатните завъртания. В базовата игра новото пресмятане се помещава под разбивката на RTP процента и над разбивката за честота на печалбите.



```

1731 }
1732 System.out.println();
1733 for (int j = 0; j < baseSymbolMoney[0].length; j++) {
1734     System.out.print(SYMBOLS_NAMES.get(j) + "\t");
1735     for (int i = 0; i < baseSymbolMoney.length; i++) {
1736         System.out.print(
1737             (double) baseSymbolMoney[i][j] / (double) lostMoney
1738             + "\t");
1739     }
1740     System.out.println();
1741 }
1742 System.out.println();
1743 System.out.println("Base Game Symbols Wins Ratio:");
1744 System.out.print("\t");
1745 for (int i = 0; i < baseSymbolMoney.length; i++) {
1746     System.out.print(" " + i + "of\t");
1747 }
1748 System.out.println();
1749 for (int j = 0; j < baseSymbolMoney[0].length; j++) {
1750     System.out.print(SYMBOLS_NAMES.get(j) + "\t");
1751     for (int i = 0; i < baseSymbolMoney.length; i++) {
1752         System.out.print(
1753             (double) baseSymbolMoney[i][j] / (double) wonMoney
1754             + "\t");
1755     }
1756     System.out.println();
1757 }
1758 System.out.println();
1759 System.out.println("Base Game Symbols Hit Rate:");
1760 System.out.print("\t");
1761 for (int i = 0; i < baseGameSymbolsHitRate.length; i++) {
1762     System.out.print(" " + i + "of\t");
1763 }
1764 System.out.println();
1765 for (int j = 0; j < baseGameSymbolsHitRate[0].length; j++) {
1766     System.out.print(SYMBOLS_NAMES.get(j) + "\t");
1767     for (int i = 0; i < baseGameSymbolsHitRate.length; i++) {

```


Режимът на безплатни завъртания е част от основната игра, но много често се извършва с отделен комплект барабани, които от своя страна имат коренно различно дискретно вероятностно разпределение и поради тази причина статистиките трябва да се събират отделно за режима на безплатни завъртания.




```

1792     System.out.print(" " + i + "of\t");
1793 }
1794 System.out.println();
1795 for (int j = 0; j < freeSymbolMoney[0].length; j++) {
1796     System.out.print(SYMBOLS_NAMES.get(j) + "\t");
1797     for (int i = 0; i < freeSymbolMoney.length; i++) {
1798         System.out.print(
1799             (double) freeSymbolMoney[i][j] / (double) lostMoney
1800             + "\t");
1801     }
1802     System.out.println();
1803 }
1804 System.out.println();
1805 System.out.println("Free Games Symbols Wins Ratio:");
1806 System.out.print("\t");
1807 for (int i = 0; i < freeSymbolMoney.length; i++) {
1808     System.out.print(" " + i + "of\t");
1809 }
1810 System.out.println();
1811 for (int j = 0; j < freeSymbolMoney[0].length; j++) {
1812     System.out.print(SYMBOLS_NAMES.get(j) + "\t");
1813     for (int i = 0; i < freeSymbolMoney.length; i++) {
1814         System.out.print(
1815             (double) freeSymbolMoney[i][j] / (double) freeMoney
1816             + "\t");
1817     }
1818     System.out.println();
1819 }
1820 System.out.println();
1821 System.out.println("Free Games Symbols Hit Frequency:");
1822 System.out.print("\t");
1823 for (int i = 0; i < freeGameSymbolsHitRate.length; i++) {
1824     System.out.print(" " + i + "of\t");
1825 }
1826 System.out.println();
1827 for (int j = 0; j < freeGameSymbolsHitRate[0].length; j++) {
1828     System.out.print(SYMBOLS_NAMES.get(j) + "\t");

```

Ново написания код бива компилиран и пакетиран в JAR архив, с всички външни библиотеки, така че да бъде готов за използване от крайния потребител.



```
MACMINI:Fruit-Machine-Simulator-with-Excel-Interface todorbalabanov$ ./gradlew jarAll
```

Deprecated Gradle features were used in this build, making it incompatible with Gradle 7.0.

Use '--warning-mode all' to show the individual deprecation warnings.

See https://docs.gradle.org/6.0.1/userguide/command_line_interface.html#sec:command_line_warnings

BUILD SUCCESSFUL in 11s

2 actionable tasks: 2 executed

```
MACMINI:Fruit-Machine-Simulator-with-Excel-Interface todorbalabanov$ █
```

Последната стъпка е изпращането на промените в отдалеченото хранилище.




```
MACMINI:Fruit-Machine-Simulator-with-Excel-Interface todorbalabanov$ git status
```

```
On branch master
```

```
Your branch is up to date with 'origin/master'.
```

```
Changes not staged for commit:
```

```
(use "git add <file>..." to update what will be committed)
```

```
(use "git checkout -- <file>..." to discard changes in working directory)
```

```
    modified:   src/main/java/Main.java
```

```
no changes added to commit (use "git add" and/or "git commit -a")
```

```
MACMINI:Fruit-Machine-Simulator-with-Excel-Interface todorbalabanov$ git add .
```

```
MACMINI:Fruit-Machine-Simulator-with-Excel-Interface todorbalabanov$ git commit -m "More statistics were printed."
```

```
[master 046141e] More statistics were printed.
```

```
1 file changed, 32 insertions(+)
```

```
MACMINI:Fruit-Machine-Simulator-with-Excel-Interface todorbalabanov$ git push
```

```
Counting objects: 6, done.
```

```
Delta compression using up to 4 threads.
```

```
Compressing objects: 100% (4/4), done.
```

```
Writing objects: 100% (6/6), 537 bytes | 268.00 KiB/s, done.
```

```
Total 6 (delta 2), reused 0 (delta 0)
```

```
remote: Resolving deltas: 100% (2/2), completed with 2 local objects.
```

```
To https://github.com/VelbazhdSoftwareLLC/Fruit-Machine-Simulator-with-Excel-Interface.git
```

```
1481264..046141e master -> master
```

```
MACMINI:Fruit-Machine-Simulator-with-Excel-Interface todorbalabanov$ █
```

Монте Карло симулациите на хазартните ротативки не дават достатъчно информация за поведението на тези игри погледнато през призмата на залагащите. За да бъдат изследвани поведенческите модели на играчите е необходимо да се реализират симулации на база игрални сесии.

