УДК 621.391.25

ПРОГРАММНАЯ РЕАЛИЗАЦИЯ ПАРАДОКСА МОНТИ ХОЛЛА

Ковальков Р. В.
Научный руководитель – Напрасников В.В., к.т.н., доцент

Цель работы – доказать отсутствие парадоксальности путем разработки алгоритма, создающего аналогию случайных экспериментов и показывающего вероятность того или иного выбора.

Парадокс Монти Холла – это известная задача по теории вероятности. Свою популярность обрела благодаря американскому телешоу, на котором была соответствующая игра. Сама задача не представляет собой парадокса, так как не содержит в себе противоречия. Однако им названа по причине неочевидности ответа для человека, не разбирающегося в теории вероятности и той известности, которую задача обрела.

Суть задачи состоит в следующем. Перед игроком три двери: за одной автомобиль, за остальными – козы (рис. 1). Игроку необходимо угадать: за какой дверью находится автомобиль. После первого выбора двери игроком, ведущий открывает одну из двух оставшихся дверей, за которой гарантировано стоит коза.

Игроку после этого разрешается поменять выбор на третью дверь. Вопрос: есть ли смысл менять дверь и с какой вероятностью он выиграет автомобиль, если все-таки решится? Увеличатся ли ваши шансы выиграть автомобиль, если вы примете предложение ведущего и измените свой выбор?

Рис.1. Отображение возможных выборов.

В этой работе предпринята попытка построения программной реализации в среде MathCad для численного моделирования экспериментов при двух разных вариантах стратегии игрока. Текст документа MathCad с комментариями представлен на (рис. 2).

```
GoatAndCar(n) := "//p1 - кол-во положительниых исходов выпадения машины НЕ ИЗМЕНЯ ВЫБОР"
                 p1 ← 0
                 "//p2 - кол-во положительниых исходов выпадения машины ИЗМЕНЯ ВЫБОР"
                 p2 ← 0
                 "Проводим n-ое кол-во эксперементов"
                 for i ∈ 1,2..n
                     "//Ставим машину (случ. образ.) за одну из 3ех дверей. За остальными - козы"
                     car_pos ← ceil(rnd(3))
                     "//Игрок выбирает дверь (случайным образом)"
                     player_choice ← ceil(rnd(3))
                     "//tmp - это кол-во коз за дверьми ЗА ИСКЛ. той двери, которую выбрал игрок"
                     tmp ← 0
                     "Пробегаемся по всем дверям"
                     for j ∈ 1,2..3
                         "Если за дверью НЕ МАШИНА и если дверь НЕ ВЫБРАНА ИГРОКОМ => tmp++"
                         tmp ← tmp + 1 if j ≠ car_pos ∧ j ≠ player_choice
                     "//Если tmp != n - 1 ( != 2) , ТО МАШИНА ЗА ДРУГОЙ ДВЕРЬЮ "
                     "//Если tmp == n - 1 ( == 2) , ТО МАШИНА ЗА ТОЙ ДВЕРЬЮ, КОТОРУЮ ВЫБРАЛ ИГРОК "
                     p2 ← p2 + 1 if tmp ≠ 2
                     p1 ← p1 + 1 otherwise
                 return ( p1/n  p2/n )

GoatAndCar(1000) = ( 0.325  0.675 )
```

Рис. 2. Функция, принимающая аргумент n - количество итераций эксперимента и возвращает вектор из двух значений: первое - вероятность выигрыша автомобиля, если не менять выбор, второе - если поменять его.

Зависимости вероятностей от количества экспериментов при разных стратегиях представлены на (рис. 3). Вывод: вторая стратегия предпочтительна.

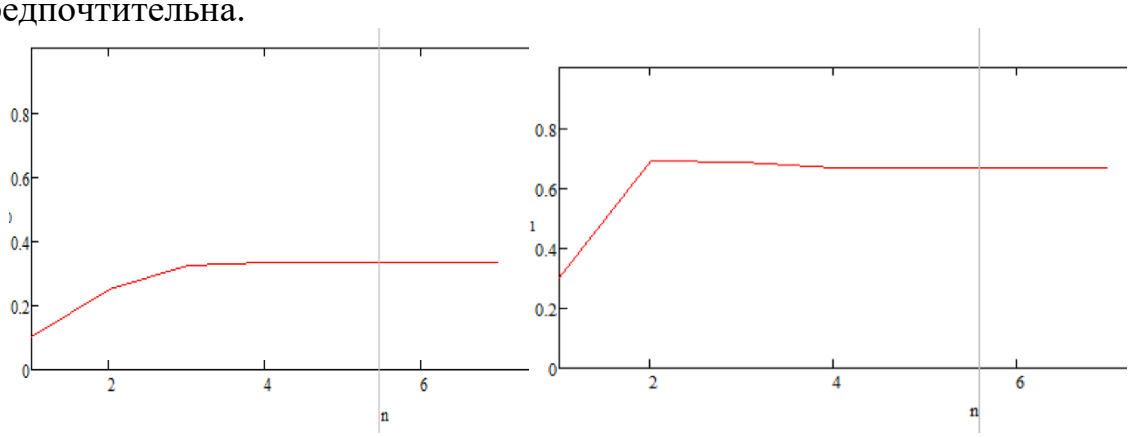

Рис. 3. График функции, отображающей вероятность выигрыша автомобиля, если не менять выбор (слева), если менять выбор (справа)