***И.Н. Паршиков, В.С. Иванов,  
ассистент Н.С. Васильев***

**Генерация карт с использованием WFC**

**Аннотация**. Показано использование алгоритма коллапса волновой функции для генерации игровых карт на языке C#.

**Ключевые слова:** алгоритм коллапса волновой функции, C#, генерация, игровой мир.

***I.N. Parshikov, V.S. Ivanov,***

***assistant N.S. Vasiliev***

**Generating maps using WFC**

**Annotation.**  The use of the Wave Function Collapse algorithm for generating game maps in C# is demonstrated.

**Keywords:** Wave Function Collapse algorithm, C#, generation, game world.

Игровые миры требуют разнообразия и уникальности, с чем помогают различные процедурные генерации.

В данной работе описывается генерация карт при помощи алгоритма коллапса волновой функции.

Wfc (Wavefunction Collapse Algorithm) – алгоритм коллапса волновой функции — это процесс, описывающий, как система квантовых частиц переходит из состояния суперпозиции в одно определенное состояние при измерении. В квантовой механике частицы могут находиться в множестве состояний одновременно, что выражается в виде волновой функции. Когда происходит измерение, волновая функция "коллапсирует" в одно из возможных состояний, что приводит к наблюдаемому результату.

Алгоритм генерации карт с использованием WFC делится на 6 шагов:

1. Разбиение карты на части

Карта делится на небольшие области (регионы), каждый из которых может находиться в "суперпозиции" — иметь несколько возможных биомов.

1. Поиск элемента с наибольшей энтропией

Энтропия региона определяется количеством возможных биомов. Чем больше вариантов, тем выше энтропия. Алгоритм выбирает регион с наименьшей определенностью (наибольшей энтропией).

1. Выполнение коллапса для найденного элемента

Для выбранного региона случайным образом выбирается один из возможных биомов, учитывая ограничения, наложенные соседними регионами.

1. Обновление состояний

После коллапса список возможных биомов для соседних регионов обновляется в соответствии с правилами совместимости.

1. Выполнение шагов 2-4 пока есть частицы в суперпозиции

Для реализации данного алгоритма использовался язык программирования C#.

Рассмотрим реализацию некоторого шага алгоритма в коде.

Коллапс для найденного элемента осуществляется так:

// Находим максимальное количество соседей без биома

var maxNeighbors = regions.Max(region =>

region.Biome == null ? region.Neighbors.Count(neighbor => neighbor.Biome == null) : 0);

if (maxNeighbors == 0 && regions.All(x=>x.Biome != null)) break; // Если нет доступных соседей, выходим из цикла

if (maxNeighbors == 0)

{

var region = regions.FirstOrDefault(x=> x.Biome == null);

region.Biome = region.PossibleBiomes.FirstOrDefault();

Console.WriteLine($"установлен биом для региона {n}");

n++;

continue;

}

В результате выполнения программы создается картинка формата .png в корне проекта.

Пример сгенерированной карты можно увидеть на рис.1.

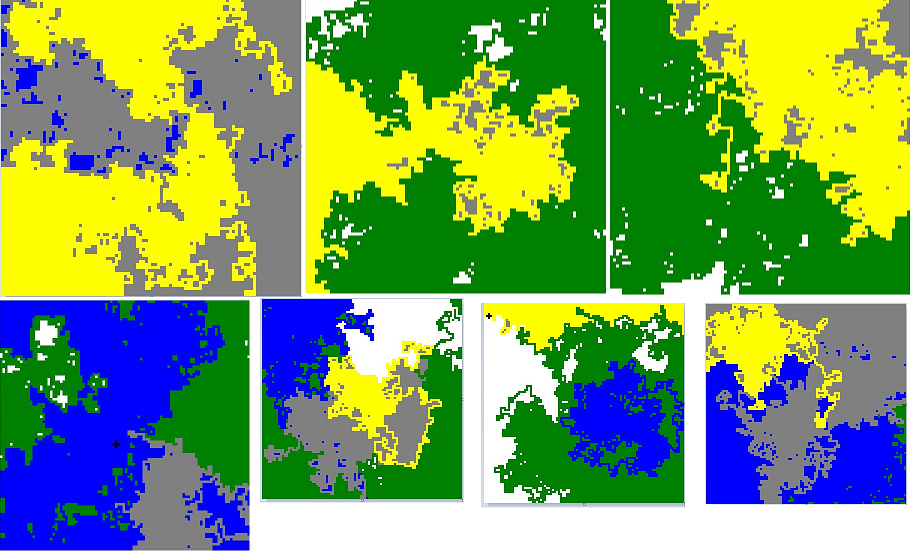


Рис.1. Пример сгенерированной карты 100\*100.

Разработанный алгоритм позволяет значительно ускорить процесс создания игровых карт, снижая затраты на ручную работу. Его можно адаптировать для различных типов игр — от стратегий до RPG. Дальнейшие исследования могут быть направлены на оптимизацию алгоритма для трехмерных ландшафтов или динамического изменения карт в реальном времени.