УДК 004.42:534.2

ГЕНЕРАЦИЯ МУЗЫКИ С ПОМОЩЬЮ L −СИСТЕМ

Кузьмицкий М.С., Сарело Е.К., студенты
Белорусский государственный университет информатики и радиоэлектроники
г. Минск, Республика Беларусь

Примичева З.Н. – канд. физ.-мат. наук, доцент

Аннотация. Изучено применение L −систем для алгоритмической генерации музыки. Совершен обзор существующих подходов, выявлены ограничения детерминированных моделей. Разработана программа на Python с использованием библиотеки pretty_midi, реализующая стохастическую L −систему с гармонической привязкой к аккордовой прогрессии.

Ключевые слова. L −система (система Линденмайера), алгоритмическая композиция, генеративная музыка, стохастическая грамматика, фрактальная структура, MIDI, Python, pretty_midi, гармоническая привязка, аккордовая прогрессия.

Введение. L −системы, или системы Линденмайера, представляют собой формальный аппарат параллельного переписывания, являющийся разновидностью формальных грамматик. Структурно L − система базируется на алфавите символов, служащем строительным материалом для строк, наборе порождающих правил, определяющем подстановку для каждого знака, начальной строке («аксиоме»), запускающей процесс генерации, и механизме интерпретации, трансформирующем полученную строку в геометрические или иные структуры. Теоретический фундамент был заложен в 1968 году Аристидом Линденмайером, венгерским биологом и ботаником из Утрехтского университета. Изначально Линденмайер применял данный математический инструмент для описания динамики клеток растений и симуляции процессов онтогенеза. В дальнейшем сфера применения расширилась: L − системы стали использоваться для моделирования морфологии разнообразных биологических организмов, а также для генерации самоподобных фрактальных структур, родственных системам итерируемых функций.

После определения параметров L − система запускает процесс эволюции, строго следуя заданным правилам. Точкой отсчета служит аксиома — начальное состояние системы. В ходе дальнейшего развития строка, кодирующая состояние объекта, претерпевает изменения. Этот процесс носит циклический характер и состоит из итераций. На каждом шаге строка сканируется последовательно, однако ключевой особенностью является параллелизм применения правил: замена символов происходит одновременно для всей строки текущей итерации. Для каждого символа подбирается правило, при котором данный символ выступает символом-предшественником для строки-преемника. При отсутствии явного правила для конкретного символа X применяется неявное правило тождества $X \to X$, что сохраняет символ без изменений. Если же правило найдено, символ-предшественник замещается на строку-преемник, указанную в правиле. Именно механизм параллельной замены отличает L −системы от классических грамматик Хомского и обеспечивает рост фрактальных структур.

Основная часть. Наиболее распространенными и функциональными разновидностями L −систем являются стохастические, контекстозависимые и параметрические.

Стохастические (или недетерминированные) системы внедряют элемент вероятностной неопределенности в процесс развития. В отличие от классических моделей, здесь для одного символа-предшественника может существовать несколько правил подстановки. Выбор конкретного правила осуществляется случайным образом на каждой итерации, согласно заданным вероятностям. Каждому варианту правила присваивается вес — числовое значение от нуля до единицы, сумма вероятностей для одного символа обычно равна единице. Это позволяет имитировать естественную вариативность, наблюдаемую в живой природе.

Контекстозависимый подход обобщает классическую модель, также допуская множественные правила для одного предшественника. Однако выбор правила диктуется не случайностью, а локальным окружением — контекстом. Учитываются символы, расположенные непосредственно слева или справа от обрабатываемого знака. Глубина контекста (количество учитываемых соседей) напрямую влияет на сложность системы: чем шире контекст, тем более тонкие взаимосвязи могут быть смоделированы.

Параметрические L −системы расширяют классический алфавит, снабжая символы числовыми параметрами. Эти параметры функционируют как переменные, позволяющие управлять количественными характеристиками при моделировании сложных структур. Правила переписывания в таких системах могут зависеть от текущих значений параметров, которые сами изменяются в процессе итераций по заданным математическим функциям. Это дает возможность моделировать не только форму, но и динамику роста, толщину ветвей или возраст органов.

Отдельного внимания заслуживает возможность генерации древовидных структур, которая является классическим примером применения L –систем в компьютерной графике. Ниже на рисунке 1 представлен код для генерации дерева с помощью L –системы, а на рисунке 2 изображено само дерево, получившееся в результате этой генерации.

```
def draw(commands, angle, step):
    stack = []
    for cmd in commands:
        if cmd == "F":
            turtle.pencolor("green")
            turtle.forward(step)
        elif cmd == "X":
            turtle.pencolor("red")
            turtle.dot(6)
        elif cmd == "+":
            turtle.right(angle)
        elif cmd == "-":
            turtle.left(angle)
        elif cmd == "[":
            stack.append((turtle.position(), turtle.heading()))
        elif cmd == "]":
            pos, head = stack.pop()
            turtle.penup()
            turtle.goto(pos)
            turtle.setheading(head)
            turtle.pendown()

axiom = "X"
rules = {
    "X": "F[+X][-X]FX",
    "F": "FF"
}
```

Рисунок 1 — пример кода для генерации дерева с помощью L –системы

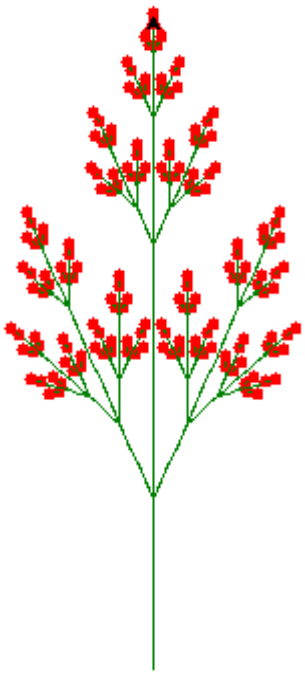

Рисунок 2 — Пример дерева, сгенерированного L –системой

На рисунке 2 представлено дерево, построенное с использованием L – системы после 4 итераций. Отчетливо видна рекурсивная самоподобная структура: каждая ветвь повторяет форму всего дерева в уменьшенном масштабе, что является характерным признаком фрактальных объектов.

Пусть горизонтальные отрезки в визуализации соответствуют длительности звучания, а вертикальные — высоте тона. Такой подход впервые систематически связал фрактальные структуры с музыкальной композицией. Метод стал фундаментом для всех последующих разработок в области алгоритмической музыки. Работа продемонстрировала, что самоподобные математические структуры могут порождать музыкально осмысленные паттерны.

Рассмотрим ключевые существующие решения для генерации музыки с использованием потенциала L −систем.

1. Работа Пшемыслава Прусинкевича (1986). Прусинкевич впервые интерпретировал строки L-систем как музыкальные партитуры по аналогии с черепашьей графикой: горизонтальные отрезки задавали длительность, вертикальные — высоту тона. Этот метод создал систематическую связь между фрактальной геометрией и композицией, став фундаментом для дальнейших исследований алгоритмической музыки.
2. Musical L-Systems Стелиоса Манусакиса (2006). Манусакис создал модульную среду на базе Max/MSP/Jitter, охватывающую все временные масштабы — от волновых форм до структуры произведения — через иерархические сети L-систем. Три метода синтеза (клеточный, слоистый, древовидный) генерировали фрактальные тембры, а ветвления строки интерпретировались как полифонические голоса. Работы автора были отмечены на конкурсе Gaudeamus.
3. L-системы в Symbolic Composer. Специализированная среда алгоритмической композиции, где L-системы работали параллельно с другими генеративными методами. Это позволяло комбинировать фрактальные структуры с классическими техниками в рамках одного проекта, демонстрируя потенциал гибридных подходов и способствуя популяризации алгоритмической композиции среди практикующих авторов.

После детального анализа решений для генерации музыки с использованием потенциала L −систем был получен вывод о том, что перечисленные инструменты остаются преимущественно математическими моделями. Музыка, сгенерированная исключительно этими методами, часто воспринимается слушателем как «генеричная или бездушная». Классические L −системы по своей природе детерминированы: идентичный ввод неизменно порождает идентичный вывод. Однако живая мелодия требует микровариаций — незаметных для глаза, но важных для уха отклонений в темпе, динамике и артикуляции. Чистые L − системы звучат «механически» именно вследствие своей математической точности и отсутствия человеческой небрежности, свойственной исполнительскому искусству.

В рамках нашей исследовательской работы была разработана программа на языке Python с использованием библиотеки pretty_midi, предназначенная для генерации музыкальных последовательностей на основе стохастической L −системы и последующего экспорта результата в формате .midi. В качестве алфавита L −системы использовался набор символов $\{F, +, -, D, d\}$, где F обозначает «проиграть текущую ноту», символы $+$ и $-$ соответствуют движению вверх или вниз, а D и d увеличивают или уменьшают длительность ноты в два раза. Аксиомой служила строка « F ». Дальнейшее развитие системы осуществлялось по стохастическим правилам переписывания, что, аналогично генерации фрактальных растений, позволяло получать различные музыкальные структуры при выборе разных наборов правил.

```
CHORD_PROGRESSION = [
    ("Am", 12.5), ("F", 12.5), ("C", 12.5), ("G", 12.5),
    ("Am", 12.5), ("Dm", 12.5), ("E", 12.5), ("Am", 12.5),
]

AXIOM = "F"
RULES = {
    "F": [
        ("F+F-F", 0.25),
        ("F-F+F", 0.25),
        ("F+dF", 0.15),
        ("F-DF", 0.15),
        ("+F-F", 0.10),
        ("-F+F", 0.10),
    ],
    "+": [("+", 0.7), ("-", 0.3)],
    "-": [("-", 0.7), ("+", 0.3)],
    "D": [("D", 1.0)],
    "d": [("d", 1.0)],
}
```

Рисунок 3 — пример аккордовой последовательности и правил стохастической L −системы

Во избежание атональности и улучшения звучания генерация мелодии осуществлялась с опорой на заранее заданную аккордовую последовательность. На основе текущего аккорда определялся набор допустимых нот, а операции «+» и «−» интерпретировались как перемещение внутри выбранной гармонической структуры. Функция интерпретации представлена ниже на рисунке 4, функции реализации стохастической L −системы на рисунке 5, а на рисунке 6 продемонстрирован фрагмент сгенерированного .midi в piano roll.

```python
def parse_lsystem_to_slots(lsystem_string):
    slots = []
    current_pitch_offset = 0 # текущая высота
    current_duration = 0.5  # начальная длительность (восьмая)
    for char in lsystem_string:
        if char == "F":
            slots.append((current_pitch_offset, current_duration))
        elif char == "+":
            current_pitch_offset = min(9, current_pitch_offset + 2)
        elif char == "-":
            current_pitch_offset = max(-9, current_pitch_offset - 2)
        elif char == "D":
            current_duration = min(2.0, current_duration * 2)
        elif char == "d":
            current_duration = max(0.25, current_duration / 2)
    return slots
```

Рисунок 4 — Интерпретация результата L —системы

```python
def _choose_rule(options):
    total_prob = sum(prob for _, prob in options)
    rand = random.uniform(0, total_prob)
    cumulative = 0
    for replacement, prob in options:
        cumulative += prob
        if rand ≤ cumulative: return replacement
    return options[-1][0]

def generate_lsystem(axiom, rules, iterations):
    s = axiom
    for _ in range(iterations):
        new_s = "".join(_choose_rule(rules[char]) if char in rules else char for char in s)
        s = new_s
        if len(s) > 5000: break # не более 5000 нот
    return s
```

Рисунок 5 — Реализация стохастической L —системы

Рисунок 6 — фрагмент сгенерированного .midi в piano roll

Заключение. В работе исследовано применение L —систем для алгоритмической композиции. Установлено, что детерминированные модели порождают механистичные последовательности, однако использование стохастических правил в разработанной программе на Python с библиотекой pretty_midi позволяет преодолеть это ограничение. Гармоническая привязка к аккордовой прогрессии обеспечивает музыкальную осмысленность генерируемых паттернов при сохранении фрактальной логики. Эксперименты подтвердили эффективность подхода для создания вариативного музыкального материала. Перспективы развития связаны с расширением параметрической модели для управления тембром и динамикой.

Список использованных источников:

1. Линденмайер А. Математические модели для клеточных взаимодействий в развитии / А. Линденмайер // Журнал теоретической биологии. — 1968. — т. 18, № 3. — с. 280–299.

2. Прусинкевич, П. Алгоритмическая красота растений / П. Прусинкевич, А. Линденмайер. — м. : мир, 1990. — 256 с.

3. Прусинкевич, П. Музыкальные l-системы / П. Прусинкевич // Труды международной компьютерной музыкальной конференции (icmc). — 1986. — с. 115–122.

4. Манусакис, С. музыкальные l-системы / С. Манусакис // proceedings of the 2006 international computer music conference. — new orleans, 2006. — с. 363–370.

5. Таубе, Г. Записки с метауровня: введение в вычислительное музыкальное мышление / г. Таубе. — м. : прогресс-традиция, 2004. — 320 с.

6. Raffensperger c. pretty_midi: a python library for working with midi files [электронный ресурс]. — 2019. — режим доступа: https://github.com/craffel/pretty-midi (дата обращения: 21.03.2026).

UDC 004.42:534.2

MUSIC GENERATION USING L – SYSTEMS

Kuzmitski M.S., Sarelo E. K., students
Belarusian State University of Informatics and Radioelectronics, Minsk, Republic of Belarus

Primicheva Z.N. – PhD in Physics and Mathematics

Annotation. The application of L –systems for algorithmic music generation is studied. Existing approaches are reviewed and the limitations of deterministic models are identified. A Python program using the pretty_midi library was developed to implement a stochastic L-system with harmonic alignment to a chord progression.

Keywords. L –system (Lindenmayer system), algorithmic composition, generative music, stochastic grammar, fractal structure, MIDI, Python, pretty_midi, harmonic binding, chord progression.