УДК 004.42

ПРИМЕНЕНИЕ ВАЙБ-КОДИНГА ПРИ ИЗУЧЕНИИ ТЕОРИИ ГРАФОВ

А. И. ЯКИМОВ, В. А. ПОМОЗОВ
Белорусско-Российский университет
Могилев, Беларусь

С развитием информационных технологий и искусственного интеллекта появляются новые методы программирования. Вайб-кодинг (от англ. *Vibe coding*) – это инновационный подход, при котором пользователь формулирует задачу на естественном языке, а искусственный интеллект (ИИ) генерирует, оптимизирует и отлаживает соответствующий код [1, 2].

В настоящее время ИИ хорошо генерирует шаблонные ответы и решает рутинные задачи, не ориентируясь на контекст большой задачи. Вайб-кодинг хорошо зарекомендовал себя при создании простых программ, решении шаблонных или простых задач. Например, при кодировании несложного алгоритма или небольшого программного модуля, идеален для прототипирования и экпериментирования.

Рассмотрим применение вайб-кодинга в образовательном процессе при изучении алгоритмов в теории графов. Пусть требуется написать программу для алгоритма Дейкстры при нахождении минимального пути от точки А до точки В. Промпт для ИИ может быть представлен так: «Реализовать алгоритм Дейкстры на Python с использованием библиотеки networkx и библиотеки для визуализации графа».

ИИ генерирует следующий код на Python (условно сокращена часть кода для визуализации графа):

```
import networkx as nx
import matplotlib.pyplot as plt
import heapq

def dijkstra(graph, start, end):
    """Алгоритм Дейкстры для поиска кратчайшего пути."""
    distances = {node: float('inf') for node in graph}
    distances[start] = 0
    previous = {node: None for node in graph}

queue = [(0, start)]
    while queue:
        current_dist, current = heapq.heappop(queue)
        if current == end:
            break
        if current_dist > distances[current]:
            continue


for neighbor, weight in graph[current].items():
            new_dist = current_dist + weight
            if new_dist < distances[neighbor]:
               distances[neighbor] = new_dist
               previous[neighbor] = current
               heapq.heappush(queue, (new_dist, neighbor))

  # Восстановление пути
    if distances[end] == float('inf'):
        return None, float('inf')
    path = []
    node = end
    while node:
        path.append(node)
        node = previous[node]
    path.reverse()
    return path, distances[end]

# Пример использования
if name == "main":
    # Ориентированный граф (вершина: {сосед: вес})
    graph = {
        'A': {'B': 4, 'C': 2},
        'B': {'D': 3, 'E': 1},
        'C': {'B': 1, 'D': 5},
        'D': {'E': 2,'F':1},
        'E': {'F':4,'D':2},
        'F':{}
```

```
    }
    start = 'A'
    end = 'F'

# Находим кратчайший путь
    path, distance = dijkstra(graph, start, end)
    if path:
        print(f"Кратчайший путь от {start} до {end}:")
        print(" → ".join(path))
        print(f"Длина пути: {distance}")
        # Визуализируем
        visualize_graph(graph, path)
    else:
        print(f"Путь от {start} до {end} не найден")
```

При исполнении кода получена визуализация графа на рис. 1.

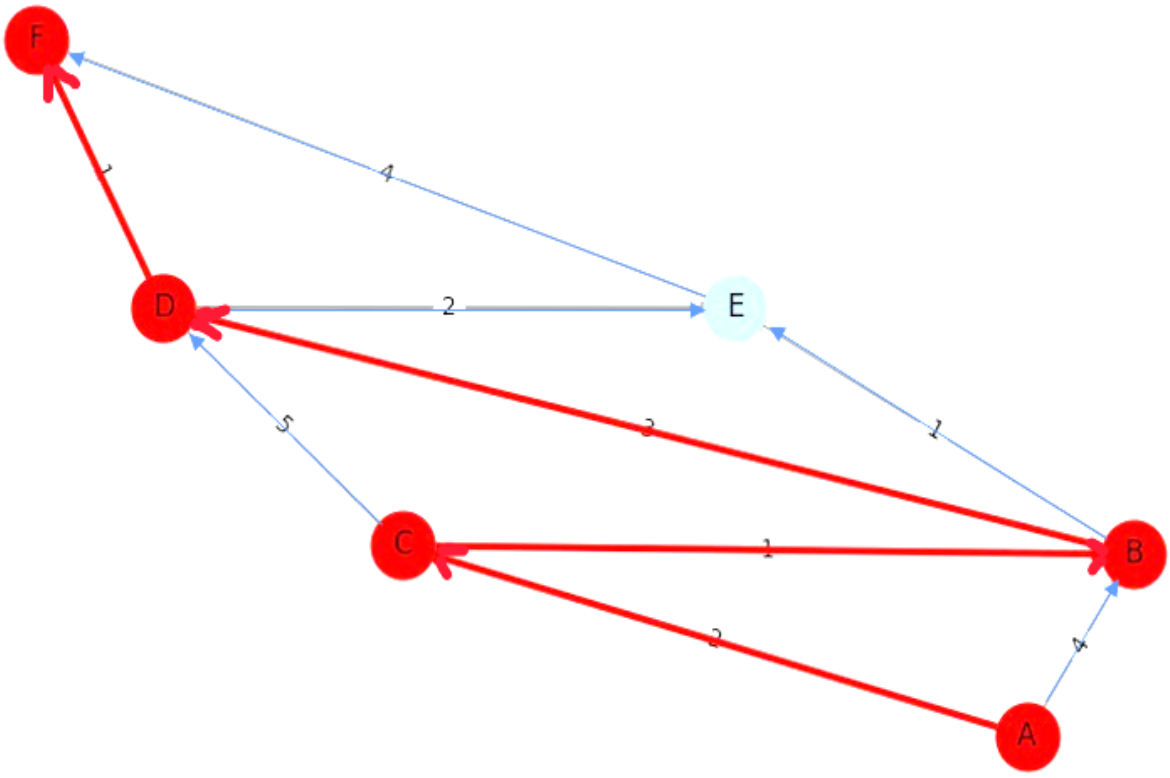

Рис. 1. Граф с кратчайшим путем A–C–B–D–F

Вайб-кодинг делает изучение алгоритмов теории графов увлекательным: студенты создают реальные прототипы (например, визуализаторы графов) за короткое время, усиливая чувство компетентности. Это соответствует теории самодетерминации, удовлетворяя потребности в автономии и релевантности через индивидуальные эксперименты. Вайб-кодинг в образовательном процессе ускоряет освоение алгоритмов, позволяя студентам сосредоточиться на логике

и структуре алгоритмов, а не на сложностях синтаксиса. Это делает процесс обучения более эффективным, открывая перед студентами возможности для творчества и инновационного мышления.

СПИСОК ИСПОЛЬЗОВАННОЙ ЛИТЕРАТУРЫ

1. **Sapkota, R.** Vibe Coding vs. Agentic Coding: Fundamentals and Practical Implications of Agentic AI / R. Sapkota, K. I. Roumeliotis, M. Karkee. – URL: https://doi.org/10.48550/arXiv.2505.19443 (date of access: 20.12.2025).

2. **Horvat, M.** What is Vibe coding and when should you use it (or not)? / M. Horvat. – URL: https://doi.org/10.5281/zenodo.16747092 (date of access: 20.12.2025).