Проект по курсу "Сложность вычислений"

Тема: Проверка 3-раскрашиваемости графа (№ 49)

Студент: Ярослав Спирин (группа 594)

Формулировка:

Требуется имлементировать алгоритм проверки, можно ли покрасить граф из n вершин в 3 цвета, за время $O(c^n)$ для $c \leq 2$. Реализация алгоритма для $c > 1.8$ принесёт не больше 7 баллов, для меньших c — вплоть до 12 при выборе наилучшего известного алгоритма.

Описание работы:

В этом практическом задании требуется имплементировать алгоритм проверки 3-раскрашиваемости графа. В данном проекте мы опишем алгоритм работы нашей программы, его реализацию, докажем его корректность, найдем время работы программы.

Описание алгоритма

Опр. Независимое множество - множество вершин, в котором никакие две вершины из этого множества не соединены ребром, и при этом нельзя добавить еще вершину в этом множество, чтобы оно осталось независимым.

Алгоритм: Найдем все максимальные по включению независимые множества в этом графе. Не теряя общности, покрасим все эти вершины в первый цвет. Проверим можем ли мы покрасить все остальные вершины в оставшиеся два цвета. Если найдем хоть одну такую покраску, то ответ - «существует», иначе «не существует».

Реализация алгоритма

Реалиазация алгоритма будет представлять собой код на C++ и основная идея, используемая в решении этой задачи будет основываться на следующую лемму:

Лемма. Либо вершина либо хотя бы одна из её соседей лежат в максимальном по включению независимом множестве.

Доказательство.

Предположим противное: пусть не лежит ни одна из соседей, тогда просто добавим эту вершину в множество. Будем поддерживать некоторое множество, в котором в конечном результате будет находиться максимальное по включению независимое множество. Возьмем вершину с минимальной степенью. Тогда добавим в искомое множество либо эту вершину, либо одну из её соседей, переберем так для всевозможных способов, в конечном итоге на каждом шаге будем получать некоторое максимальное по включению независимое множество. Для остальных вершин нужно проверить, можем ли мы их покрасить в 2 цвета, а это решается методом обхода в глубину исходного графа.

Корректность

Мы красим некоторое подмножество в один цвет, пусть оно не является максимальным по включению независимым множеством, тогда просто добавим в него вершин, чтобы сделать его таковым, это, как минимум, не ухудшит возможность покраски оставшихся вершин в остальные два цвета. В итоге, мы перебираем всевозможные максимальные по включению независимые множества, и для каждого из них пытаемся раскрасить граф $\Longrightarrow$ рассматриваем все случаи покраски, значит, алгоритм выполняется корректно.

Время работы алгоритма

Оценим число максимальных по включению независимых множеств. Заметим, что на каждом шаге можем брать некоторую вершину с минимальной степенью, тогда время работы алгоритма выглядит следующим образом:

Пусть степень данной вершины равна d, тогда $T(n) \sim (d+1) * T(n-(d+1))$, где $T(n)$ - время работы программы. Тогда $T(n) = (d+1)^{\frac{n}{d+1}}$, легко доказать, что эта функция максимальна при $d = 2$, используя метод математической индукции. Кроме того, это легко следует из того, что в итоге нужно найти минимум функции $k^{\frac{1}{k}}$. Тогда у нас в итоге таких множеств не больше, чем $3^{\frac{n}{3}} \sim 1.44^n$. Кроме того, для каждого такого множества мы делаем 2-раскраску для оставшихся вершин, что занимает $O(n+m)$ методом обхода в глубину.

Итоговая сложность работы программы: $O(1.44^n * (n+m))$, что лучше чем $O(1.8^n)$.

Результаты

В программе так же было написано тестирование для реализованного алгоритма:

- Для проверки корректности работы программы, был так же написан алгоритм 3-раскрашиваемости графа за 3^n , и проверка совпадают ли ответы для данных алгоритмов. В результате проверки все ответы совпали.
- Для проверки времени работы программы, был протестирован алгоритм на достаточно больших данных (для данной задачи), где N достигало порядка $\sim 30 - 40$. В результате проверки программа выдавала ответ в приемлимое для данной сложности время.

Источники

- http://www.sci.brooklyn.cuny.edu/~amotz/GC-ALGORITHMS/PRESENTATIONS/coloring.pdf (http://www.sci.brooklyn.cuny.edu/~amotz/GC-ALGORITHMS/PRESENTATIONS/coloring.pdf)
- http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.341.9463&rep=rep1&type=pdf (http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.341.9463&rep=rep1&type=pdf)