ИЗВЕСТИЯ
ТОМСКОГО ОРДЕНА ОКТЯБРЬСКОЙ РЕВОЛЮЦИИ
И ОРДЕНА ТРУДОВОГО КРАСНОГО ЗНАМЕНИ ПОЛИТЕХНИЧЕСКОГО
ИНСТИТУТА имени С. М. КИРОВА

Том 243 1972

НЕКОТОРЫЕ ВОПРОСЫ ГЕНЕРИРОВАНИЯ ГРАФОВ НА ЭВМ

Ю. Н. ЕФИМОВ

(Представлена научным семинаром вычислительной лаборатории ТПИ)

Большой класс задач прикладной математики можно свести к задачам, сформулированным в терминах теории графов [1]. Это прежде всего транспортные и комбинаторные задачи, задачи потокораспределения, задачи, связанные с экономной записью и обработкой больших массивов информации, а также большая группа экстремальных задач теории расписаний. Задачи теории расписаний возникают при необходимости упорядочить во времени течение дискретных процессов, выполняемых с помощью ограниченного множества преобразователей. Подобные задачи имеют место при оперативном планировании и управлении предприятиями, при управлении ходом разработок с помощью сетевых методов (системы СПУ), в теории алгоритмов, в теории игр и в различных других специальных дисциплинах.

Все вышеперечисленные задачи являются задачами анализа графов большой размерности. При решении сетевых задач на ЭВМ, видимо, было бы целесообразно уметь генерировать различные сети для имитации реальных отношений. Генераторы случайных сетей могут найти широкое применение в самых разнообразных математических исследованиях, использующих теорию графов.

Программы анализа крупных графов на ЭВМ обычно логически сложны и вызывают большие трудности при отладке. Часто вызывает затруднение определение границ применимости разработанной программы, так что возникает опасение, что программа не будет удовлетворять требованиям какого-либо реального случая. Генерируя случайные сети, можно быстро проверить программу в различных режимах и установить границы применимости.

Точные решения экстремальных сетевых задач теории расписаний найдены лишь для исключительно упрощенных моделей, а основные практические результаты основываются на эвристических соображениях. При этом для оценки степени эффективности эвристического алгоритма и границ его применяемости удобно пользоваться следующей зависимостью (по аналогии с [2]):

$$M[K]=f(P),$$

где K — коэффициент эффективности алгоритма,

P — параметр графа.

Очевидно, что такую зависимость не трудно получить, генерируя сети с переменным P в необходимом интервале.

При поточном решении однотипных задач в сетевой постановке по одной и той же программе необходимо заранее оценить время решения

каждой задачи. Построение функции $t=f(P)$ также легко осуществимо с помощью генератора графов.

В теории графов генератор сетей, очевидно, может найти эффективное применение для исследования графов со случайной топологией.

Этот далеко не полный перечень возможного использования генератора сетей показывает, что разработка вопросов генерирования графов может оказаться весьма плодотворной. Однако из литературных источников нам неизвестно, чтобы такие вопросы ставились и решались. В настоящей работе предлагается один из возможных алгоритмов генерирования случайных сетей на универсальных ЭВМ.

Пусть мы имеем для ЭВМ программный датчик (ДСЧ), генерирующий целые случайные числа a_i, распределенные по какому-нибудь закону (например, равномерному) в диапазоне $\alpha \div \beta$. Назовем α и β соответственно нижней ($a_{нг}$) и верхней ($a_{вг}$) границей чисел a_i.

Граф будем представлять в виде множества вершин J и их отображений Г, так что запись информации для каждой вершины i имеет вид:

i, ω_i
j_1, p_1
j_2, p_2
. . . . . .
$j\omega$, $p\omega$,

где i, j — номера вершин, причем $j \in \Gamma_i$;

ω_i — локальная степень вершины i (количество дуг, исходящих из i);

p — какой-либо параметр p (длительность, пропускная способность, стоимость и т. д.), отнесенный к дуге (i, j).

Поскольку многие алгоритмы анализа графов, особенно в автоматизированных системах управления и системах СПУ, требуют предварительного упорядочения исходной информации о графе по слоям [3], то желательно, чтобы граф генерировался сразу упорядоченным соответствующим образом.

Общее представление о размерности и топологии графа дает знание следующих числовых характеристик:

$|\varphi|$ — количество слоев;

$|\Gamma\varphi|$ — количество вершин в слое φ;

ω_i — количество дуг, исходящих из вершины i.

Очевидно, что вышеперечисленные данные должны явиться той исходной информацией для ЭВМ, на основе которой будет генерироваться сеть. Причем, для большей гибкости генератора сетей следует предусмотреть возможность задания каждого из этих параметров либо в виде соответствующих верхних и нижних границ и в каждом конкретном случае получать параметр с помощью ДСЧ, либо в виде конкретного числа. Второй способ является частным случаем первого и его нетрудно осуществить, придав некоторые дополнительные функции ДСЧ. Например, при $a_{вг}=0$ реализовать выдачу датчиком случайных чисел всегда определенного числа $a=a_{нг}$.

Отметим также, что во многих случаях нулевой слой φ_0 ($\Gamma|\varphi_0=\varnothing$) имеет некоторые специфические особенности по сравнению с другими слоями. Например, он содержит небольшое количество вершин, так как эти вершины соответствуют или товарным изделиям в АСУ, или стокам в транспортных сетях, или конечным целям разработок в сетевых графиках систем СПУ. Поэтому введем дополнительную характеристику сети $|\Gamma\varphi_0|$ — количество вершин нулевого слоя, которую будем задавать аналогично ранее названным.

Учитывая вышеизложенное, предлагается следующий алгоритм генерирования графов на ЭВМ:

1. Розыгрыш числа слоев $|\varphi|$; $C_1:=|\varphi|$.

2. Розыгрыш $|J\varphi_0|$; i, $R:=|J\varphi_0|$; $E_{1O}:=|J\varphi_0|+1$; $E:=1$.
3. Розыгрыш $|J\varphi|$; $C_2:=|J\varphi|+1$; $C_3=1$.
4. $C_3:=C_3-1$; $C_3=0$?. Да. Переход к *п.* 9. Нет. Переход к *п.* 5.
5. $Ei_0=0$?. Да. Переход к *п.* 13. Нет. Переход к *п.* 6.
6. $E_{1O}:=Ei_O-1$; $j:=Ei_0$.
7. Розыгрыш параметра p_{ij}.
8. Формирование строки j, p_{ij} в i-й записи. Переход к *п.* 4.
9. $C:=2$; $C_2:=C_2-1$; $C_2=0$? Да. переход к *п.* 12. Нет. Переход к *п.* 10.
10. $i:=i+1$; розыгрыш ω_i, $C_3:=\omega_1+1$.
11. Формирование «шапки» (i, ω_i) i-й. Переход к *п.* 4.
12. $E:=R+1$; $R:=i$; $C_1:=C_1-1$; $C_1=0$? Да. Останов машины, граф получен. Нет. Переход к *п.* 3.
13. $C:=C-1$; $C-0$? Да. Переход к *п.* 14. Нет. Переход к *п.* 15.
14. $C:=1$; розыгрыш j при $j_{нг}=1$; $j_{вг}=R$; Переход к *п* 7.
15. Розыгрыш j при $j_{нг}=E$, $j_{вг}=R$; переход к *п.* 7.

Описанный алгоритм реализован на ЭВМ «Урал-11Б». В программе был использован ДСЧ с равномерным распределением. По мере накопления информация о графе переписывалась на магнитную ленту. Время генерирования и записи на МЛ сети, содержащей 4000 вершин, 10 слоев и ω_i в интервале $2 \div 10$ составляет 10 минут. Программа использовалась при отладке некоторых алгоритмов, разработанных для АСУП-Томск [4]. Опыт ее использования доказывает целесообразность разработки программ подобного рода.

Видимо, целесообразны дальнейшие исследования в области создания генераторов графов определенной структуры (например, деревьев, графов с замкнутыми контурами, плоских графов и т. д.), а также графов с характеристиками, выдаваемыми разными ДСЧ, отличающимися законами распределения.

ЛИТЕРАТУРА

1. К. Берж. Теория графов и ее применения. ИЛ., М., 1962.
2. Ю. Н. Ефимов. Некоторые вопросы статистического исследования сетевых графиков. «Известия ТПИ», т. 187, Томск (в печати).
3. Ю. Н. Ефимов. Алгоритм нахождения порядковой функции сетевого графика. «Известия ТПИ», т. 168, Томск, 1969.
4. Ю. Н. Ефимов, В. И. Кизев, В. И. Невраев, Ф: И. Перегубов, П. А. Седельников. Основные принципы построения и функционирования. АСУП-Томск. (данный сборник).