

**ПРОГРАММА ПРЕДВАРИТЕЛЬНОГО АНАЛИЗА СЛОЖНОГО
СЕТЕВОГО ГРАФИКА С ЛОКАЛИЗАЦИЕЙ ЛОГИЧЕСКИХ
ОШИБОК**

Ю. Н. ЕФИМОВ

(Представлена научным семинаром вычислительной лаборатории ТПИ)

В настоящее время все большее признание получает автоматизированная система сетевого планирования, известная под названием PERT [1, 2]. Сущность этой системы заключается во всестороннем исследовании технологической зависимости работ друг от друга, составлении сетевого графика в виде логической последовательности работ и дальнейшем анализе графика с целью выделения участков потенциальных затруднений. Одним из преимуществ системы является то, что отдельные части графика составляются непосредственными исполнителями работ. Однако при «сшивании» этих частей сетевого графика в единую сеть из-за несогласованности между отдельными составителями возможно появление логических ошибок в виде замкнутых контуров и обрывов.

Для сетевого графика $G = (I, \Gamma)$ [3], имеющего некоторое подмножество исходных Ei_0 и завершающих Ei_n событий, определим путь L , как последовательность работ, начинающуюся с какого-нибудь события $i_0 \in Ei_0$.

$$L = (i_0, i_1, i_2, \dots, i_k, \dots).$$

Такие пути в общем случае могут быть конечными и бесконечными. Очевидно, что для последнего события i_n конечного пути $\Gamma i_n = \emptyset$. Конечный путь назовем простым, если в нем:

- а) последнее событие $i_n \in Ei_n$;
- б) никакие события не встречаются дважды.

Логически непротиворечивые сетевые графики имеют только простые пути. В бесконечных и конечных путях с повторяющимися событиями всегда можно выделить контур μ — последовательность, у которой начальное событие совпадает с конечным.

$$\mu = (i_k, i_{k+1}, \dots, i_m), \quad \text{где } i_m \equiv i_k.$$

Конечный путь, для которого $i_n \in Ei_n$, имеет обрыв первого рода. Если найдется хотя бы одно событие $i \in I$, для которого $\Gamma i^{-1} = \emptyset$, но $i \in Ei_0$, то в сетевом графике имеется обрыв второго рода.

Противоречивость исходной информации затрудняет анализ сетевого графика и ставит под сомнение правильность результатов вычислений. Некоторые особенности самих методов анализа иногда позволяют обнаружить логические ошибки в исходной информации. Так, при расчете временных параметров методом итераций [4] или при нахожде-

нии порядковой функции [5] замкнутые контуры в сетевом графике могут быть обнаружены по несходимости процесса счета. Обрывы же при этом не обнаруживаются, а приводят к неверным результатам вычислений. Наличие замкнутых контуров ведет к «заикливанию» в программах, использующих алгоритм непосредственного анализа (полного перебора) [6]. Поэтому в программы анализа сетевых графиков обычно включают специальные блоки для прерывания счета и локализации обнаруженных ошибок. После необходимых исправлений машинный анализ часто приходится начинать сначала до обнаружения следующей ошибки и т. д. Такая методика расчета намного увеличивает непроизводительные затраты машинного времени. Естественно, возникает необходимость создания специальной программы для локализации всех логических ошибок в сетевом графике. Эта программа должна использоваться для предварительного анализа сетевого графика на стадии его «сшивания». При этом желательно вычислить и длину критического пути (t_{cp}) с тем, чтобы, исправляя логические ошибки, произвести одновременно временную корректировку работ. Опишем принцип работы такой программы, составленной для двухадресной ЭЦВМ «Минск-1».

Программа использует видоизмененный алгоритм непосредственного анализа сетевого графика. Исходная информация (UU) для работы программы задается в виде упорядоченного списка работ и их длин [7]. Каждой работе соответствует элемент UU, занимающий одну ячейку оперативной памяти машины.

i) r a j t_{ij}.

Перед началом работы программы ячейки памяти, соответствующие событиям $i_n \in Ei_n$, отмечаются знаком минус. В противном случае эти события будут выделены как места обрывов первого рода.

Работа программы (см. блок-схему на рис. 1) заключается в упорядоченной последовательности шагов, называемых шагами вправо и влево. Первый шаг делается вправо, начиная с события $i_0 \in Ei_0$. Шаги вправо совершаются до встречи события i , имеющего $\Gamma i = \emptyset$ или метку (знак минус). В этом случае считаем очередной шаг вправо невозможным и делаем шаг влево. Событию, в которое ведет любой шаг влево, присваивается метка и временная оценка $t'n(i)$, вычисляемая по формуле

$$t'n(i) = \max [t'^*n(i); t'n(i) + t_{ij}], \text{ где}$$

$t'n(i)$ — наибольшая длина пути, последующего за событием i ;
 $t'^*n(i)$ — значение $t'n(i)$, полученное к моменту вычисления.

Каждый шаг вправо фиксируется строкой в специальной таблице пути, характеризующей пройденный путь и определяющей шаги влево. Количество строк в этой таблице, равное числу работ, входящих в путь до рассматриваемого события, подсчитывается счетчиком C_{Γ_1} . Содержимое C_{Γ_1} сравнивается с заранее заданной величиной $\rho \geq \varphi_{\max}(i)$, где $\varphi_{\max}(i)$ — максимальное значение порядковой функции анализируемого графика [5]. При $[C_{\Gamma_1}] \geq \rho$ печатается таблица пути до первого повторяющегося события i_e . Событию i_e присваивается метка и от него делается шаг влево. Так выделяются контуры из бесконечных путей.

Если при шаге вправо встречается событие i_k , для которого $\Gamma i_k = \emptyset$, то печатается таблица пути, т. е. локализуется обрыв первого рода. Событие i_k получает метку, и делается шаг влево.

Если при шаге вправо встречается меченное событие i_m , то по таблице пути проверяется, первый ли раз в рассматриваемый путь

входит это событие. Затем делается шаг влево с печатью таблицы пути, если i_m встречается повторно, и без печати в противном случае. Так выделяются замкнутые контуры из конечных путей.

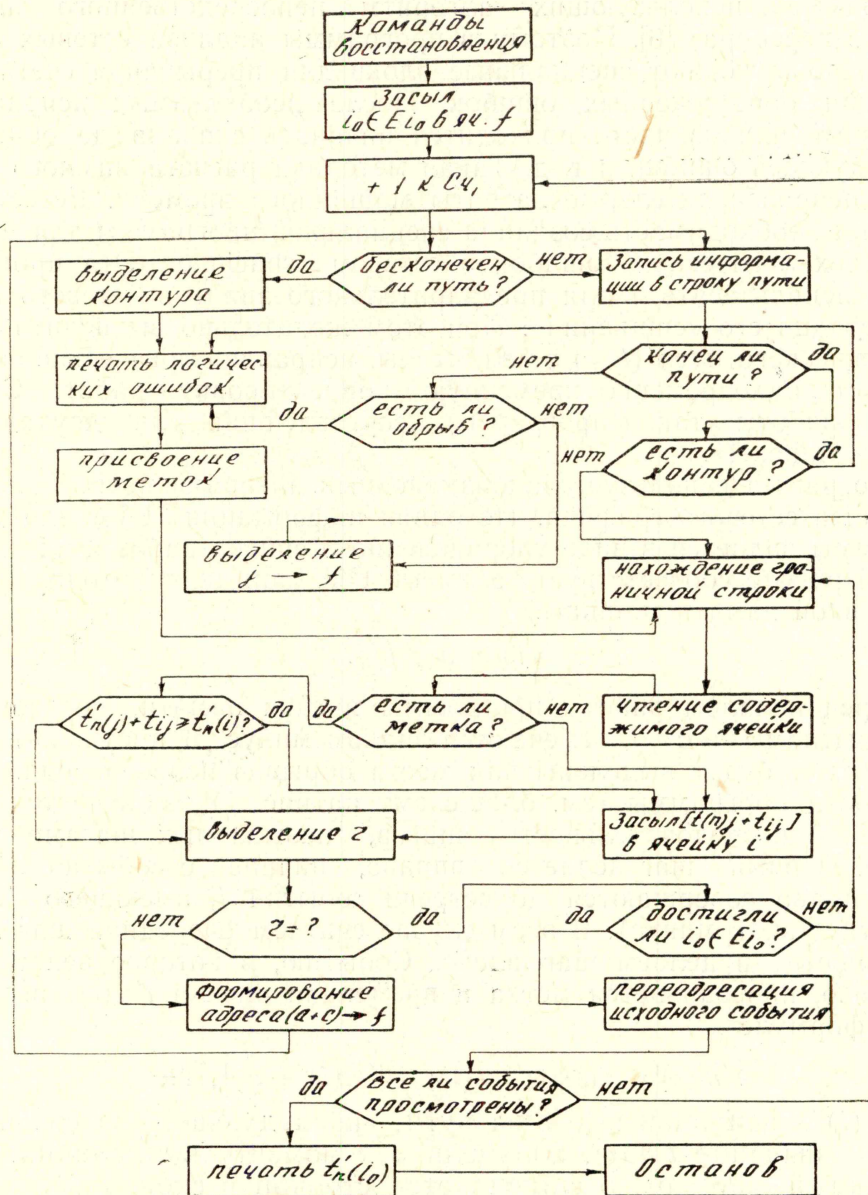


Рис. 1. Блок-схема программы предварительного анализа сложного сетевого графика с локализацией логических ошибок

На k -м шаге влево производится вычисление величины $t'_n(i)$ для события i_0 , из которого делается первый шаг вправо. $t'_n(i_0) = t_{кр}(i_0)$, где $t_{кр}(i_0)$ — максимальная длина пути, начинающегося в i_0 . Описанный процесс анализа повторяется для каждого $i_0 \in E i_0$.

После исправления обнаруженных логических ошибок весь расчет надо повторить с целью проверки внесенных исправлений и вычисления верных значений $t'_n(i)$. Для локализации обрывов второго рода необходимо повторить анализ для сетевого графика с обратной ориентацией работ.

ЛИТЕРАТУРА

1. Г. С. Поспелов, А. И. Тейман. Автоматизация процессов управления разработками больших систем или сложных комплексов. Изв. АН СССР, сер. техн. киберн., 1963, № 4.
2. Ю. А. Авдеев, А. П. Николаева. Управление сложными разработками по методу критического пути. Сб. Вычислительные системы, вып. II, Новосибирск, Изд-во ИМ СО АН СССР, 1964.
3. К. Берж. Теория графов и ее применения. М., изд-во ИЛ., 1962.
4. А. Т. Петрова, Н. Н. Карнаухова. Об одном алгоритме нахождения критического пути сетевого графика. Сб. Вычислительные системы, вып. II, Новосибирск, Изд-во ИМ СО АН СССР, 1964.
5. Ю. Н. Ефимов. Алгоритм нахождения порядковой функции сетевого графика, (данный сборник).
6. Г. М. Адельсон-Вельский, Ф. М. Филлер. Программа вычисления сетевых графиков. Ж. вычисл. мат. и мат. физики, т. 5, № 1, 1965.
7. Ю. Н. Ефимов, В. И. Кизев, В. М. Разин. Об одном алгоритме автоматизации расчетов для процессов управления разработками сложных систем. Изв. ТПИ, т. 154, изд. ТГУ, 1967.