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

Том 187 1974

ОПТИМИЗАЦИЯ ПЛАНА РАСПРЕДЕЛЕНИЯ РАБОТ В СИСТЕМЕ СПУ С ОГРАНИЧЕННЫМИ РЕСУРСАМИ

Ю. Н. ЕФИМОВ

(Представлена научным семинаром вычислительной лаборатории)

При создании уникальных образцов новой техники, больших систем и сложных комплексов наиболее удобным инструментом планирования и управления является метод, основанный на анализе сетевых графиков [1]. Сетевые методы планирования и управления (СПУ) позволяют выявить «узкие» места в проекте еще на стадии планирования. На втором этапе — этапе управления — система СПУ помогает сконцентрировать внимание руководителей на «критических» работах, отсрочка в выполнении которых влечет за собой соответствующий сдвиг даты окончания всего комплекса. Используя быстродействующие электронные вычислительные машины (ЭВМ), система СПУ становится оперативной системой управления, быстро реагирующей на всякое изменение первоначально принятых параметров (графической модели и временных характеристик работ).

В настоящее время разработаны алгоритмы временного [2, 3] и логического [4] анализа сетевых графиков, накоплен некоторый опыт применения систем СПУ в промышленности и строительстве. Однако при анализе сетевых графиков обычно не учитывают имеющиеся в наличии ресурсы (рабочая сила, оборудование и т. д.), то есть ресурсы считаются неограниченными. Но так как практически ресурсы всегда ограничены, то такой анализ приводит к занижению общего расчетного времени выполнения всех работ комплекса ($t_{кр}$) и дает неправильные даты выполнения отдельных работ. В литературе имеются попытки учета ресурсных ограничений за счет включения в понятие критического пути критичности не только по времени, но и по ресурсам [5]. Практически ограниченные ресурсы учитываются еще при составлении сетевых графиков самой топологией сети (распараллеливание работ, введение фиктивных работ и т. д.). Естественно, что такой учет производится далеко не наилучшим способом. Таким образом, возникает задача оптимального распределения ресурсов по работам, т. е. составления такого расписания работ с учетом их технологических связей, которое при заданных ограниченных ресурсах обеспечивало бы выполнение проекта в минимальное время. Часто представляет интерес решение обратной задачи, то есть нахождение оптимального потребного ресурса для выполнения проекта в заданный срок. До настоящего времени не найдено полного решения этих задач, однако разрабатываются эвристические алгоритмы [6], приводящие к достаточно хорошим,

практически удовлетворительным решениям. Опишем один из таких алгоритмов для решения обратной задачи, а также программу, реализующую алгоритм на ЭВМ «Минск-1».

Пусть сетевой график без контуров $G = (I, U)$ задан множеством работ (U), их длин $t_{ij} \geqslant 0$ и интенсивностей потребления ресурса на каждой работе $\nu_{ij} \geqslant 0 (ij \in U)$. Будем считать, что все работы потребляют ресурс одного вида, например рабочую силу. Причем, интенсивность потребления ресурса на каждой работе остается постоянной на все время выполнения работы, т. е. $\nu_{ij} = \underset{t(0,t_{ij})}{\text{const}}$. Пусть график движения рабочей силы (график загрузки оборудования) представляет собой ступенчатую функцию $R(t)$, имеющую хотя бы один максимум в промежутке времени $(0, t_{кр})$. Задача состоит в таком размещении работ, чтобы величина $\max\limits_{t \in (0, t_{кр})} R(t)$ была минимальной. Другими словами, требуется определить минимальное количество рабочих (машин), необходимых для выполнения всего комплекса работ не позже $t_{кр}$; где $t_{кр}$ — величина, рассчитанная без учета ресурсных ограничений.

Оптимальное размещение работ в пределах времени $(0, t_{кр})$ может быть достигнуто за счет задержки времени начала некоторых работ, не лежащих на критическом пути. Такие работы, называемые «некритическими», обладают некоторым резервом времени для своего выполнения. Как известно [7], можно выделить четыре рода резервов некритических работ: полный, свободный и два вида частных. Полный резерв времени $P_п(ij)$ работы — это максимально возможное время отдаления окончания данной работы за счет задержки ее начала, при котором срок завершающего события комплекса операций не отдаляется, т. е. продолжительность критического пути не увеличивается.

$$P_п(ij) = t_п(j) - t_р(i) - t_{ij},$$

где

$t_п(j)$ — наиболее поздний из допустимых сроков свершения конечного события (j) работы (ij),

$t_р(i)$ — наиболее ранний из возможных сроков свершения начального события (i) работы (ij),

t_{ij} — продолжительность (длина) работы (ij).

Полный резерв времени работы (ij) является резервом времени максимального из путей, проходящих через эту работу. Использование части полного резерва времени работы для задержки начала ее выполнения изменяет полные резервы времени всех последующих за ней работ и требует их пересчета. Кроме того, может случиться, что весь резерв времени будет израсходован на начальных работах, а конечные работы уже не будут иметь возможности быть задержанными, что значительно ухудшает план распределения. Следовательно, использование $P_п(ij)$ для оптимизации размещения работ при заданном времени выполнения проекта $(t_{кр})$ связано с вычислительными трудностями и не всегда приводит к удовлетворительным результатам. Для целей оптимизации желательно использовать такие резервы времени, расходование которых на начальных работах никак не отражается на величине резервов последующих работ. Таким резервом может явиться свободный резерв времени работы — максимальное время, на которое можно задержать начало выполнения работы без изменения сроков свершения ее начального (i) и конечного (j) событий.

$$P_с(ij) = \max\{0, [t_р(j) - t_п(i) - t_{ij}]\}.$$

Однако, как показал опыт расчетов, подавляющее большинство работ реальных сетевых графиков имеют $P\ (i\,j) = 0$. Сдвиги начала работ только в пределах их $P_{\text{с}}$ дают в большинстве случаев незначительное уменьшение величины $\max\limits_{t\in(0,t_{\text{кр}})} R(t)$. Поэтому нам представляется целесообразным использовать для оптимизации размещения работ при заданном времени выполнения проекта частный резерв второго вида. Частный резерв второго вида $[P_{\text{п}}\ (ij)]$ образуется у работ, непосредственно предшествующих событию, в котором пересекаются пути разной длины. Этот резерв показывает, какая часть полного резерва времени работы может быть использована для сдвига начала этой и предшествующей ей работ, принадлежащих отрезку пути, лежащему между двумя ближайшими точками его пересечения с путями большей длины, при условии, что этот сдвиг не вызовет сокращения резерва ни у одной из последующих работ.

$$P_{\text{п}}''(ij)=t_{\text{р}}(j)-t_{\text{р}}(i)-t_{ij}=P_{\text{п}}(ij)-p(j), \tag{1}$$

$P(j)$ — резерв времени события j, определяемый по формуле

$$p(j)=t_{\text{п}}(j)-t_{\text{р}}(j)$$

Как видно из (1), использование $P_{\text{п}}''(i\,j)$ изменяет только величины $P_{\text{п}}(i\,\text{j})$ у предшествующих работ, величины же $P_{\text{п}}''(i\,j)$ всех работ не изменяются, поэтому не требуется пересчета резервов времени.

Исходная информация о каждой работе, видимо, должна включать в себя:

1) $t_{\text{рн}}$ — время раннего начала работы,
2) $P_{\text{п}}''$ — резерв времени,
3) N — обозначение работы (двухиндексное или одноиндексное),
4) t_{ij} — продолжительность работы,
5) υ_{ij} — интенсивность потребления ресурса.

Следовательно, исходная информация представляет собой матрицу M размерами $(n\times 5)$, где n — число работ. Тогда решаемую задачу оптимизации можно сформулировать следующим образом.

Увеличивая $t_{\text{рн}}$ некоторых работ $(i\,j)\in U$ на величину $\delta\leqslant P_{\text{п}}''(i\,j)$, найти план размещения работ во времени $(\text{о},t_{\text{кр}})$, минимизирующий $\max\limits_{\in(0,t_{\text{кр}})} R(t)$. Очевидно, что искомый $\min\max R(t)\geqslant R_{\text{ср}}$, где $R_{\text{ср}}=\dfrac{\sum\limits_{i\text{j}\in U}\upsilon_{ij}t_{ij}}{t_{\text{кр}}}$ среднее количество ресурса, потребляемое в единицу времени.

Рассмотрим следующие подмножества множества U.

$$A(t)=\{(ij)\in U/t_{\text{рн}}(ij)=t\}, \tag{2}$$

$$C(t)=\{(ij)\in U/t_{\text{рн}}(ij)+t_{ij}=t\}, \tag{3}$$

$$B(t)=\{ij)\in U/t_{\text{рн}}(ij)<t<t_{\text{рн}}(ij)+t_{ij}\}. \tag{4}$$

Эти подмножества можно рассматривать как одноместные функции независимого переменного t (времени). Из (2—4) следует, что $A(t)$ включает в себя работы, начинающиеся в момент времени t, $C(t)$ — работы, оканчивающиеся в момент t, $B(t)$ — работы, начатые ранее и продолжающиеся в момент времени t. Назовем фронтом работ $F(t)$ подмножество работ, выполняющих в момент времени t.

$$F(t)=\{(ij)\in U/t_{\text{рн}}(ij)+t_{ij}>t\}. \tag{5}$$

Очевидно, что $F(t) = A(t) \cup B(t)$.
При $t = 0$ $B(0) = \varnothing$; $F(0) = A(0)$.
При

$$t = t_{кр} \quad B(t_{кр}) = \varnothing; \; A(t_{кр}) = \varnothing; \; F(t_{кр}) = \varnothing.$$

Функцию $R(t)$ можно определить как сумму интенсивностей потребляемых ресурсов всех работ фронта $F(t)$.

$$R(t) \sum_{(ij) \in F(t)} \nu_{ij}. \tag{6}$$

Поэтому для нахождения $R(t)$ достаточно сформировать фронт работ для каждого t в интервале $(0, t_{кр})$ и просуммировать ν_{ij} работ, принадлежащих $F(t)$. Тогда алгоритм минимизации сводится к следующему.

1. Находится функция $R_0(t)$ и значение $\max R_0(t)$, т. е. строится график движения рабочей силы без оптимизации.

Пусть для какого-то времени уже сформирован фронт работ $F(t_к)$, тогда функция $R_0(t$ строится следующим образом:

а) Суммируется интенсивности потребления ресурса на этих работах и находится точка $R_0(t_к)$.

$$R_0(t_к) = \sum_{(ij) \in F(t_к)} \nu_{ij}.$$

б) Сравниваются величины $R(t_к)$ и $\max R_0(t_к)$, большая из них принимается за $\max R_0(t_к)$.

в) Длительности всех работ $(ij) \in F(t_к)$ уменьшаются на единицу.

г) Из работ, длительность которых становится равной нулю, формируется подмножество, $C(t_{к+1})$, из остальных — подмножество $B(t_{к+1})$.

$$B(t_{к+1}) = F(t_к) / C(t_{к+1}).$$

д) Из множества работ U выбираются работы, начинающиеся в момент времени $t_{к+1}$, и формируется подмножество $A(t_{к+1})$.

е) Находится новый фронт работ $F(t_{к+1})$.

$$F(t_{к+1}) = A(t_{к+1}) \cup B(t_{к+1}).$$

ж) При $t < t_{кр}$ перейти к а), при $t = t_{кр}$ функция $R_0(t)$ построена и найдено значение $\max R_0(t)$.

2. Делается попытка построить функцию $R_1(t)$, задав предел $\max R_1(t) \leqslant \max R_0(t) - 1$.

Пусть для какого-то времени $t_к$ уже сформирован фронт работ $F(t_к)$, тогда построение $R_1(t)$ осуществляется следующим образом:

а) Работы фронта упорядочиваются по возрастанию величины $P_п''(ij)$ (т. е. приоритет получают работы с меньшим резервом времени).

б) Находится $R(t_к) = \sum_{(ij) \in F(t_к)} \nu_{ij}$, причем суммирование производится до тех пор, пока получаемая сумма не превзойдет величину $[\max R_0(t) - 1]$. Работа, интенсивность потребления ресурса которой делает сумму больше величины $[\max R_0(t) - 1]$, сдвигается вправо, т. е. исключается из $F(t_к)$ и переносится в $F(t_{к+1})$. При этом время ее начала увеличивается, а резерв времени уменьшается на единицу. Однако сдвиг вправо возможен лишь в случае, если резерв сдвигаемой работы боль-

ше нуля. В противном случае дальнейшее улучшение плана невозможно и следует перейти к 3. После сдвига работы продолжается суммирование интенсивностей потребления ресурса остальных работ фронта.

в) Из длительностей всех несдвинутых работ фронта вычитается по единице и всем этим работам присваивается значение $P_{п}^{и}=0$, так как считается, что работы не допускают перерыва.

г) Остальные шаги построения $R_1(t)$ аналогичны построению $R_0(t)$. После построения $R_1(t)$ делается попытка построения $R_2(t)$ с учетом ограничения

$$\max R_2(t) \leqslant \max R_1(t) - 1$$

и т. д., пока после построения $R_п(t)$ дальнейшее улучшение плана невозможно.

3. Находится коэффициент улучшения

$$K_{ул} = \frac{\max R_п(t)}{\max R_0(t)}.$$

Этот коэффициент дает количественную характеристику степени улучшения первоначального плана распределения работ. Очевидно, что $K_{ул} \leqslant 1$.

Ниже приводится формальное описание алгоритма в терминах языка АЛГОЛ-60 [8].

```
begin integer r, Ct 1, CtF, n, no, max S, max 1 S;
    procedure пересылка (u, d, A, B);
      integer u, d; integer array A, B;
        begin integer j; for j:=1 step 1 until 5 do
         B[d, j]:=A[u, j) end;
ввод    (n, no); r:=0; max S:=0;
   begin integer Aн, f; integer array N [no: no+n, 1:5];
ввод (N);
      L1 : Ct1:=0; CtF:=1; Aн :=no; f:=0;
      L2: begin integer i, Ct4; integer array F [1:CtF, 1:5];
            Ct4:=0; fori:=1 step 1 whil Ct4=CtF do
            begin go to if F [i, 3]>0 then M2 else M1
              M1:CtF:=CtF-1; if F[CtF, 3]>0 then
    пересылка (CtF, i, F, F) else go to M1;
              M2:Ct4:=Ct4+1 end
          end;
        go to if f=0 then формирование F else
        if CtF>0 then L3 else L5;
формирование F: begin integer i;
    for i:= Aн step 1 while N[i, 1]>Ct1 V Aн >no+n do
    if N[i, 3]>0 then begin CtF:=CtF+1;
пересылка (i, CtF, N, F) end; Aн :=Aн+1;
    if Aн =no+n then f:=f+1 end;
I.3: begin integer i, ro; integer array F [1; CtF, 1:5],
          r[1:5];
          if CtF=1 then begin CtF:=CtF+1; печать <0>;
          go to L4 end;
```

```
        M1: ro:=0; for i:=1 step 1 until CtF−1 do
          begin if F[1, 2]>F[i+1, 2] then begin
пересылка   (i, 1, F, r); пересылка  (i+1, i, F, F);
пересылка (i, i+1, r, F); ro:=ro+1 end
           end;
          go to if ro>0 then M1 else M2;
        M2: end;
    begin integer S, i; S:=0; i:=0;
        MO: i:=i+1; ifi>CtF then go to M4;
          if F[i, 5]=0 then go to M3; S:=S+F[i, 5];
          if S≤max S then go to M2 else if r=0
          then go to M1; S:=S−F[i, 5]; if F[i, 2]>0
          then begin F[i, 2]:=F[i, 2]−1; F[i, 1]:=
          F[i, 1]+1; go to MO end else go to L8;
        M1: max S:=S;
        M2: F[i, 2]:=0; F[i, 4]:=F[i, 4]−1;
            go to if F[i, 4]=0 then M3 else MO;
        M3: F[i, 3]:=0−F[i, 3]; печать  (F[i, 3], F[ i, 1]);
            go to MO;
        M4: end; печать  (S);
    L5: интервал;     go to if r=0 then L6 else L7;
    L6: max 1S:=max S;
    L7: max S:=max S−1; go to L1
    end;
    L8: begin real Kул ; Kул :=max S/max 1S;
печать  (Kул ) end;
```

Описанный алгоритм был апробирован на ЭЦВМ «Минск-1». Исходная информация об одной работе записывалась в одну ячейку оперативной памяти машины в виде

$$(m)\ t_{\text{п}}\ P\ N\ t\ \upsilon, \tag{7}$$

где

m — номер ячейки памяти,

N — одноиндексный номер работы.

Программа занимает 233 «8» ячейки оперативной памяти машины и может оптимизировать сетевые графики, содержащие до 770 работ. Причем информация о работах может задаваться с пропусками, тогда в пропущенной ячейке должно стоять стандартное слово — нули во всех разрядах.

Программа выдает на печать по желанию оператора:

1) функцию $R_0\ (t)$,
2) функцию $R_{\text{п}}(t)$ и $K_{\text{ул}}$,
3) все промежуточные функции $R_{\text{к}}(t)$.

При этом во всех трех случаях может быть отпечатано время начала для каждой работы или только для сдвинутых работ. Программа позволяет улучшать план распределения работ крупных сетевых графиков многотемных разработок, потребляющих несколько видов ресурсов. Для этого из всех работ рассчитанного по времени сетевого графика выбираются работы, потребляющие только один вид ресурса. Эти работы представляют исходную информацию о каком-то подграфе $G_1 \subset G(I, U)$, к которому применяется описанный алгоритм. Аналогичные расчеты надо применить для каждого вида ресурсов. Производить оптимизацию следует после полного временного анализа сетевого графика, ибо тогда достаточно просто программным путем сформировать исходную информацию в форме (7).

ЛИТЕРАТУРА

1. Г. С. Поспелов, А. И. Тейман. Автоматизация процессов управления разработками больших систем или сложных комплексов. Изв. АН СССР, сер. техническая кибернетика, № 4, 1963.

2. Г. М. Адельсон-Вельский, Ф. М. Филлер. Программа вычисления сетевых графиков. Ж. вычисл. мат. и мат. физики, том 5, № 1, 1965.

3. А. Г. Петрова, Н. Н. Карнаухова. Об одном алгоритме нахождения критического пути сетевого графика. Сб. «Вычислительные системы», вып. 11, Новосибирск, изд-во ИМ СО АН СССР, 1964.

4. Л. Я. Лейфман, А. Т. Петрова. Некоторые алгоритмы для анализа ориентировочных графов. Сб. «Вычислительные системы», вып. 11, Новосибирск, изд-во ИМ СО АН СССР, 1964.

5. I. D. Wiest. Operations Research, 12, 1964.

6. С. И. Зуховицкий, И. А. Радчик. Математические методы сетевого планирования, М., 1965.

7. Основные положения по разработке и применению СПУ. М., «Экономика», 1965.

8. М. И. Агеев. Основы алгоритмического языка АЛГОЛ-60. ВЦ АН СССР, М., 1965.