

НЕКОТОРЫЕ АЛГОРИТМЫ ОТОБРАЖЕНИЯ ЛОГИЧЕСКИХ СЕТЕЙ В ВЫЧИСЛИТЕЛЬНЫЕ ЕП-СРЕДЫ

А.И. Макаров, В.А. Скоробогатов
(Новосибирск)

1. Рассмотрим n -мерное ($n \geq 2$) евклидово пространство E^n , плотно и без наложений заполненное центрально-симметрическими n -многогранниками (ц.с. n -многогранниками) [1, 2]. Прямую, проходящую через центры симметрии противоположащих $(n-1)$ -граней ц.с. n -многогранника, назовем осью, а множество K_1 ц.с. n -многогранников, центры симметрии которых принадлежат некоторой оси I , назовем колонной вдоль этой оси.

1.1.0. Счетное множество ц.с. n -многогранников $\mathcal{K} \subseteq E^n$ назовем вычислительной средой, лежащей в пространстве E^n (обозначение BC^n), если каждому ц.с. n -многограннику (элементу BC^n) принадлежащему \mathcal{K} можно поставить в соответствие функцию ϕ из автоматно и соединительно полного набора (базиса) функций $B_{nc} = \{F, T, S\}$ а его $(n-1)$ -граням — переменные из областей определения и значений этой функции, причем F — логически полное множество функций, — T — множество временных задержек, S — соединительно полное множество функций [3].

1.1.1. 1-окрестность элемента $m_1 \in BC^n$ назовем множеством $O_1(m_1) \subseteq BC^n$, состоящее из самого элемента m_1 и тех элементов $m_2 \in BC^n$, которые имеют одну общую $(n-1)$ -грань с элементом m_1 .

O - окрестность элемента $m_1 \in BC^n$ будем считать множество, состоящее из самого элемента m_1 .

r - окрестность элемента $m_1 \in BC^n$ назовем множество $O_r(m_1) = \bigcup O_r(m_j)$, где $m_j \in O_{r-1}(m_1)$.

r - зона элемента $m_1 \in BC^n$ назовем множество

$$z_r(m_1) = O_r(m_1) \setminus O_{r-1}(m_1).$$

O - зона элемента $m_1 \in BC^n$ будем считать множество, состоящее из самого элемента m_1 .

r - окрестность множества $M \subseteq BC^n$ назовем множество $O_r(M) = \bigcup O_r(m_1)$, где $m_1 \in M$.

Цепью, связывающей элементы m_1 и m_j множества BC^n , назовем множество $C(m_1, m_j) \subseteq BC^n$ состоящее из элементов m_k , где $k = 0, 1, 2, \dots, l$, таких, что $m_0 = m_1$, $m_l = m_j$ и $m_{k+1} \in O_1(m_k)$. (Натуральное число l назовем длиной цепи $C(m_1, m_j)$).

I.1.2. Пусть b_{ij} - $(n-1)$ -грань элемента m_1 являющаяся общей для элементов m_1 и m_j , где $m_j \in O_1(m_1)$ ($i \neq j$), т.е. для фиксированных i и j : b_{ij} совпадает с b_{ji} . Множество элементов $m \in BC^n$ назовем программой, если каждому элементу $m_1 \in M$ поставлена в соответствие некоторая определенная функция $\varphi_{m_1} \in B_{BC}$ и переменная этой функции, соответствующая $(n-1)$ -грани b_{ij} , отождествлена с переменной функции φ_{m_j} , соответствующей $(n-1)$ -грани b_{ji} элемента $m_j \in O_1(m_1)$ ($i \neq j$).

Пусть $D^n \in S$ есть функция, которая отождествляет переменные, соответствующие противоположным $(n-1)$ -граням элемента $m_1 \in BC^n$, а $R^n \in S$ - функция, отождествляющая переменные, соответствующие всем $(n-1)$ -граням элемента $m_1 \in BC^n$.

Для каждого элемента $m_1 \in BC^n$ зададим некоторую ориентацию t_{m_1} , т.е. определенную нумерацию его $(n-1)$ -граней, что естественным образом задает нумерацию элементов $m_j \in O_1(m_1)$, ($i \neq j$). Множество $\Theta = \{t_{m_1}\}$ для всех $m_1 \in BC^n$ назовем ориентацией BC^n . Будем говорить, что задан код BC^n , если для BC^n задана ориентация и для каждого элемента $m_1 \in BC^n$ указаны элементы $m_j \in O_1(m_1)$, ($i \neq j$).

I.1.3. Элемент $m_1 \in BC^n$ назовем смежным элементом $m_j \in M$ вдоль оси I , если $I) m_1 \in K_I$, $m_j \in K_I$,

2) все элементы $m_k \in K_I$, лежащие между элементами m_1 и m_j , являются элементами программы M и $\varphi_{m_k} = D^n \in S$.

Элемент $m \in BC^n$ назовем условно смежным элементу $m_j \in BC^n$ вдоль оси I , если $I) m_1 \in K_I, m_j \in K_I$.

2) для каждого элемента $m \in K_I$, лежащего между элементами m_1 и m_j , либо $m_k \in BC^n \setminus M$, либо $m_k \in M$ и $\varphi_{m_k} = D^n \in S$.

Число $s_{m_1}(N)$ осей, вдоль которых элемент $m_1 \in BC^n$ условно смежен элементам множества $N \subseteq BC^n$, назовем степенью условной смежности элемента m_1 относительно множества N . Очевидно, что

$$0 \leq s_{m_1}(N) \leq \frac{d_{BC}}{2},$$

где d_{BC} - число $(n-1)$ - граней элемента m_1 , называемое в дальнейшем степенью BC^n .

I.I.4. Множество $N \subseteq BC^n$ назовем связным, если для любых элементов m_1 и m_j , принадлежащих N , существует связывающая их цепь $C^i(m_1, m_j)$, все элементы m_k которой также принадлежат множеству N .

Компонентой элемента m_1 во множестве $N \subseteq BC^n$ назовем соединение всех связных подмножеств множества N , содержащих элемент m_1 .

Пусть M_D - множество всех элементов программы M , которым поставлена в соответствие функция $D^n \in S$.

Программу M^* назовем расширенной, если M^* есть соединение программы M и множества $M' \subseteq BC^n \setminus M$ элементов m_1 , для которых $s_{m_1}(M \setminus M_D) \geq 2$ и каждому элементу $m_1 \in M'$ поставлена в соответствие функция $D^n \in S$. Компоненту \tilde{M}^i элемента $m_1 \in M$ в расширенной программе M^* назовем правильной программой.

Свободной границей программы M назовем множество $\Gamma_{CB}(M) = O_1(M) \setminus M$. Свободной колонной $K_{I_{CB}}(M)$ вдоль оси I программы M назовем колонну K_I , для которой справедливы равенства: $K_I \cap \Gamma_{CB}(M) \neq \emptyset$ и $K_I \cap (M \setminus M_D) = \emptyset$.

I.I.5. Применим к некоторой конечной связной программе M , лежащей в бесконечной BC^n , код которой задан, следующий алгоритм \tilde{A} (алгоритмы приведения программы к правильной):

п. I. Определяем свободную границу $\Gamma_{CB}(M')$ исходной программы M' (в начале работы алгоритма \tilde{A} считаем $M' = M$).

п. 2. Для каждого элемента $m_i \in \Gamma_{CB}(M')$ определяем степень условной смежности $a_{m_i}(M' \setminus M_D)$ элемента m_i относительно множества $M' \setminus M_D$.

пп.2а. Определяем δ -окрестность элемента m_i : $O_\delta(m_i)$;

пп.2б. Определяем пересечение $C = O_\delta(m_i) \cap M' \neq \emptyset$;

пп.2в. Для каждой колонны K_I содержащей элемент m_i и элемент $m_k \in C$, определяем существует ли элемент $m_r \in M' \setminus M_D$ условно смежный элементу m_i , для чего необходимо перебрать элементы множества $C' = O_{|M'|}(m_i) \cap K_I$, где $|M'|$ - мощность множества элементов программы M' .

пп.2г. Число колонн K_I , для которых существуют элементы m_r , равно $a_{m_i}(M' \setminus M_D)$.

п. 3. Если существует элемент $m_i \in \Gamma_{CB}(M')$, для которого $a_{m_i}(M' \setminus M_D) \geq 2$, то элементу m_i ставим в соответствие функцию $D^* \in S$. Полученную программу считаем исходной. Переход к п. 1.

п. 4. Если для любого $m_i \in \Gamma_{CB}(M')$ $a_{m_i}(M' \setminus M_D) < 2$, то конец.

Результатом работы алгоритма \tilde{A} является конечная правильная программа \tilde{M} .

1.1.6. Пусть дана конечная связная программа M , лежащая в бесконечной BC^n .

Тогда для правильной программы \tilde{M} легко доказать следующее:

Теорема 1. Существует множество $M^0 \subset BC^n$ такое, что каждый элемент $m_i \in M^0$ имеет $a_{m_i}(\tilde{M} \setminus \tilde{M}_D) = 0$. Действительно, т.к. для любого элемента $m_j \in \Gamma_{CB}(\tilde{M})$ $a_{m_j}(\tilde{M} \setminus \tilde{M}_D) \leq 1$, а число осей ц.с. n - многогранника $\frac{dM}{2} \geq 2$, то существует $K_{ICB}(\tilde{M})$, содержащая элемент m_j . В силу конечности программы \tilde{M} колонна $K_{ICB}(\tilde{M})$ содержит элементы m_i для которых

$$a_{m_i}(\tilde{M} \setminus \tilde{M}_D) = 0.$$

Теорема 2. Для двумерного евклидова пространства ($n = 2$), множество M^0 несвязно, т.е. $M^0 = \bigcup M_i^0$ и $M_i^0 \cap M_j^0 = \emptyset$ при $i \neq j$.

Теорема 3. Для n -мерного евклидова пространства ($n \geq 3$) множество M^0 связно.

1.1.7. Пусть дано связное множество $N \subset BC^n$ и элементы m_i и m_j , принадлежащие N . Известно, что кратчайшую цепь

$C_{min}(m_1, m_2)$, все элементы которой принадлежат множеству M , можно получить с помощью "волнового" алгоритма Ля [4], который в несколько измененном виде сводится к следующему алгоритму A_n :

- п. 1. Определяем k -зону $Z_k(m_1)$ элемента m_1 .
- п. 2. Определяем l -зону $Z_l(m_2)$ элемента m_2 .
(В начале работы алгоритма A_n $k=0, l=0$).
- п. 3. Если $Z_k(m_1) \cap Z_l(m_2) = \emptyset$, то переход к п.9.
- п. 4. Определяем $Z_{k+1}(m_1)$.
- п. 5. Если $Z_{k+1}(m_1) \cap Z_l(m_2) = \emptyset$, то переход к п.9.
- п. 6. Определяем $Z_{l+1}(m_2)$.
- п. 7. Если $Z_{k+1}(m_1) \cap Z_{l+1}(m_2) = \emptyset$, то переход к п. 9.
- п. 8. В п. 1, п.2, п. 3, п. 4, п.5, п.6, п.7 индексы " k " и " l " заменяем на " $k+1$ ", " $l+1$ ".
- п. 9. Выбираем некоторый элемент $m_{k1} \in C$.
- п.10. Выбираем некоторый элемент $m_{k-1}^1 \in O_1(m_{k1}) \cap Z_{k-1}(m_1)$.
- п.11. Выбираем некоторый элемент $m_{l-1}^2 \in O_1(m_{k1}) \cap Z_{l-1}(m_2)$.
- п.12. В п.10, п.11 заменяем одиночные индексы " k " на " $k-1$ ", " l " на " $l-1$ "; в п.10 элемент m_{k1} на m_{k-1}^1 ; в п. 11 элемент m_{k1} на m_{l-1}^2 .
- п.13. Если выбраны элементы $m_0^1 = m_1$ и $m_0^2 = m_2$, то конец.

Если нет, то переход к п. 10.

Результатом работы алгоритма A_n является кратчайшая цепь $C_{min}(m_1, m_2) = (m_0^1, m_1^1, \dots, m_{k-1}^1, m_{k1}, m_{l-1}^2, \dots, m_l^2, m_0^2)$

длина которой равна $(k+1)$. Алгоритм A_n можно применить для построения цепи, связывающей элементы $m_1 \in \Gamma_{CB}(\tilde{M})$ и

$$m_2 \in \Gamma_{CB}(\tilde{M}),$$

где \tilde{M} - правильная программа в BC^n при $n \geq 3$, поскольку для элементов m_1 и m_2 найдутся условно смежные им элементы m_0^1 и m_0^2 , принадлежащие связанному множеству M^0 .

I.I.8. Пусть в BC^n при $n=2$ задана конечная правильная программа \tilde{M} (в этом случае возможны два вида ц.с. n -многогранников: четырехугольники ($d_{BC} = 4$) и шестиугольники ($d_{BC} = 6$)). Тогда возможны следующие случаи построения цепей

$C(m_s, m_r)$, связывающих элементы $m_s \in \Gamma_{I^c}(\tilde{M})$ и $m_r \in \Gamma_{J^c}(\tilde{M})$ таким образом, что каждый элемент m_i цепи $C(m_s, m_r)$ не принадлежит множеству $\tilde{M} \setminus \tilde{M}_D$ и если m_i смежен элементу m_{i-1} вдоль оси I , а элементу m_{i+1} вдоль оси J , ($I \neq J$), то $m_i \in M^0$;

1. $m_s \in K_{I^c}(\tilde{M})$, $m_r \in K_{I^c}(\tilde{M})$, т.е. элемент m_s условно смежен элементу m_r . Тогда цепь $C(m_s, m_r)$ состоит из элемента m_s , элементов $m_i \in K_{I^c}(\tilde{M})$, лежащих между элементами m_s и m_r , и элемента m_r .

2. $m_s \in K_{I^c}(\tilde{M})$, $m_r \in K_{J^c}(\tilde{M})$ и $K_{I^c}(\tilde{M}) \cap K_{J^c}(\tilde{M}) = \emptyset$. Тогда цепь $C(m_s, m_r)$ состоит из элемента m_s , элементов $m_i \in K_{I^c}(\tilde{M})$, лежащих между элементами m_s и m_c , элемента m_c , элементов $m_j \in K_{J^c}(\tilde{M})$, лежащих между m_c и m_r , и элемента m_r .

3. $m_s \in K_{I^c}(\tilde{M})$, $m_r \in K_{J^c}(\tilde{M})$ и $K_{I^c}(\tilde{M}) \cap K_{J^c}(\tilde{M}) = \emptyset$. Тогда в силу конечности программы \tilde{M} существует колонна K такая, что $K_{I^c}(\tilde{M}) \cap K = m_c \in M^0$, $K_{J^c}(\tilde{M}) \cap K = m_d \in M^0$ и элемент m_c условно смежен элементу m_d . В этом случае цепь $C(m_s, m_r)$ состоит из элемента m_s , элементов $m_i \in K_{I^c}(\tilde{M})$, лежащих между элементами m_s и m_c , элемента m_c , элементов $m \in K$, лежащих между элементами m_c и m_d , элемента m_d , элементов $m_j \in K_{J^c}(\tilde{M})$, лежащих между элементами m_d и m_r , и элемента m_r .

1.1.9. Логическая сеть.

Пусть дана логическая сеть L (л.с. L) [5,6]. Заменяя элементы л.с. L , узлы отождествления входных полюсов элементов, входные и выходные полюса л.с. L вершинами, а связи между элементами л.с. L дугами, получим G_L -граф, соответствующий логической сети L . Граф G_L можно задать, перенумеровав его вершины, начиная с некоторой, подряд, исходя из матрицы смежности G_L , а любую дугу задать парой номеров вершин, ей инцидентных, причем порядком номеров вершин в паре можно задать направление дуги (в дальнейшем рассматривается такое задание графа G_L , в котором вершины, соответствующие входным и выходным полюсам л.с. L , не нумеруются.)

Будем говорить, что задан код логической сети L , если задан граф G_L и для каждой вершины графа указана функция, выполняемая соответствующим элементом л.с. L .

Базисом B_L логической сети L назовем набор различных функций, выполняемых элементами л.с. L , в том числе и "фиктивными" элементами, заменившими узлы отождествления входных полюсов элементов л.с. L .

Введем следующие ограничения:

1) Степень графа л.с. G_L $d_{G_L} \leq d_{BC}$.

2) Базис л.с. $B_L \subseteq B_{BC}$.

1.2.0. Алгоритм A^n размещения логической сети L в BC^n .

Исходными данными для алгоритма A^n являются:

1. Код BC^n .

2. Код логической сети L , удовлетворяющей ограничениям: $d_{G_L} \leq d_{BC}$, $B_L \subseteq B_{BC}$.

3. Элемент $m_1 \in BC^n$.

Элемент BC^n , которому поставлена в соответствие функция φ_k , выполняемая элементом л.с. L , соответствующим " k "-той вершине графа G_L , будем обозначать m_k .

п. 0. Начало: функцию φ_1 ставим в соответствие элементу $m_1 \in BC^n$ и считаем его исходной программой M' .

п. 1. Присоединение вершины.

пп.1а. По коду л.с. L находим еще не размещенные " s "-тые вершины G_L , смежные исходной вершине " k ". ($k < q \leq s \leq q + d_{G_L} - 2 \leq m$, где m - число вершин графа G_L а q - натуральное число).

Если таких вершин нет, то переход к п. 3 (в начале работы алгоритма A^n : $k = 1, 2 \leq s \leq d_{G_L} + 1$).

пп.1б. По коду BC^n находим элементы $m'_s \in \Gamma_{CD}(M')$, смежные элементу m_k исходной программы M' .

пп.1в. Согласно ориентации BC^n элементам m'_s ставим в соответствие функции φ_s , при этом после размещения каждой " s "-той вершины G_L к полученной программе M' применяем алгоритм \tilde{A} , что в результате дает правильную программу \tilde{M}'_1 .

п. 2. Построение пути.

пп.2а. По коду л.с. L находим уже размещенные " r "-тые

вершины G_L , смежные вершине "а" ($r \neq k$). (Если таких вершин нет, то переход к п.3).

пп. 2б. Для элементов m_s и m_r по коду BC^n находим смежные элементы m'_s и m'_r , принадлежащие $\Gamma_{CB}(\tilde{M}'_1)$.

пп. 2в. Если размерность $BC^n: n \geq 3$, то для элементов m'_s и m'_r находим условно смежные им элементы m_s^0 и m_r^0 , принадлежащие множеству M^0 и колоннам $K_{I_{CB}}(\tilde{M}'_1)$ и $K_J(\tilde{M}'_1)$, соответственно. С помощью алгоритма A_n находим цепь $C(m_s^0, m_r^0)$. Элементам m'_s , m'_r и элементам $m_p \in C(m_s^0, m_r^0)$, которые условно смежны элементам m_{p-1} вдоль оси I' , а элементам m_{p+1} - вдоль оси J' , $I' \neq J'$, ставим в соответствие функцию $R^n \in S$. Остальным элементам $m_p \in C(m_s^0, m_r^0)$, элементам $m_1 \in K_{I_{CB}}(\tilde{M}'_1)$, лежащим между элементами m'_s и m_s^0 , и элементам $m_j \in K_{J_{CB}}(\tilde{M}'_1)$, лежащим между элементами m'_r и m_r^0 , ставим в соответствие функцию $D^n \in S$.

Элементам $m_s^0 = m_0 \in C(m_s^0, m_r^0)$ и $m_r^0 = m_1 \in C(m_s^0, m_r^0)$ ставим в соответствие функцию $D^n \in S$, если $m_1 \in C(m_s^0, m_r^0)$ принадлежит колонне $K_{I_{CB}}(\tilde{M}'_1)$ и $m_{1-1} \in C(m_s^0, m_r^0)$ принадлежит колонне $K_{J_{CB}}(\tilde{M}'_1)$. Иначе, элементам m_s^0 и m_r^0 ставится в соответствие функция $R^n \in S$.

пп. 2г. Если размерность $BC^n: n = 2$, то для 1) случая построения цепи $C(m'_s, m'_r)$ элементам m'_s и m'_r ставим в соответствие функцию $R^n \in S$, а элементам $m_1 \in K_{I_{CB}}(\tilde{M}'_1)$, лежащим между элементами m'_s и m'_r , функцию $D^n \in S$.

для 2) случая построения цепи $C(m'_s, m'_r)$ элементам m'_s , m'_r и m_c ставим в соответствие функцию $R^n \in S$, а элементам $m_1 \in K_{I_{CB}}(\tilde{M}'_1)$, лежащим между элементами m'_s и m_c , и элементам $m_j \in K_{J_{CB}}(\tilde{M}'_1)$, лежащим между элементами m'_r и m_c , ставим в соответствие функцию $D^n \in S$.

для 3) случая построения цепи $C(m'_s, m'_r)$ элементам m'_s , m'_r , m_c и m_d ставим в соответствие функцию $R^n \in S$, а элементам $m_1 \in K_{I_{CB}}(\tilde{M}'_1)$, лежащим между элементами m'_s и m_c , элементам $m_e \in K$, лежащим между элементами m_c и m_d , элементам $m_j \in K_{J_{CB}}(\tilde{M}'_1)$, лежащим между элементами m_d и m'_r , ставим в соответствие функцию $D^n \in S$.

пп. 2е. После построения пути к полученной программе M'_2 применяем алгоритм \tilde{A} , что в результате дает правильную программу \tilde{M}'_2 .

п. 3. Проверка числа построенных вершин графа G_L .

пп. 3а. Если $s < k$ то полученную программу считаем исходной программой m' и в п.1 и п.2 заменяем номер "к" на "к+1".

пп. 3б. Если $\alpha = m$, то конец.

Результатом работы алгоритма Λ^n является правильная программа \tilde{M}_L^n , соответствующая логической сети L , размещенной в вычислительной среде BC^n . Естественно, что после получения программы \tilde{M}_L^n элементам, которым поставлена в соответствие функция $D^n \in S$, не участвующим в построении соединений между другими элементами, можно сопоставить "пустые" функции $0 \notin V_{BC}$.

На рис. 2 представлена программа \tilde{M}_L^n , соответствующая логической сети L (рис. 1) в BC^n при $n = 2$ и $d_{BC} = 6$, причем элементы, отмеченные пунктиром, принадлежат расширенной программе M_L^{n+1} , а элементам, лежащим вне обведенной области, можно поставить в соответствие функцию $0 \notin V_{BC}$.

2.0.0. Алгоритм Λ^2 размещения L в BC^2 при $d_{BC} = 4$.

Рассмотрим BC^2 , имеющую $d_{BC} = 4$.

Зададим в BC^2 систему прямоугольных координат (x, y) так, чтобы центры элементов BC имели целочисленные координаты. Тогда оси, проходящие через элемент m с координатами (i, j) , обозначим соответственно I и J , а колонны вдоль этих осей K_I и K_J .

В дальнейшем при рассмотрении особенностей алгоритма Λ^n при $n = 2$ и $d_{BC} = 4$ вместо обозначения " m_k " элемента BC^n применяется обозначение " $m(i_k, j_k)$ ". Результатом применения алгоритма Λ^2 к коду логической сети L является правильная программа \tilde{M}_L , для которой легко доказать

2.0.1. Лемма. Не существует колонны K такой, что пересечение $N = K \cap \tilde{M}_L \neq \emptyset$ и всем элементам $m \in N$ поставлена в соответствие функция $D \in S$.

Лемма доказывается методом индукции по шагам. Для программы, построенной на первом шаге работы алгоритма Λ^2 лемма очевидна. Пусть лемма справедлива для программы, построенной на n -м шаге алгоритма Λ^2 . Тогда на $(n+1)$ -м шаге (п.1 или п.2) алгоритма Λ^2 функция $D \in S$ ставится в соответствие только тем элементам BC , которые принадлежат колонне $K_{CB}(\tilde{M}_2')$ (или $K_{CB}(\tilde{M}_1')$), некоторому элементу которой на данном шаге ставится в соответствие функция $\phi \in V_{BC}$ и $\phi \neq D \in S$. Таким образом для программы, построенной на $(n+1)$ -м шаге, лемма также справедлива.

2.0.2. Теорема 4. Программа \tilde{M}_L является прямоугольной.

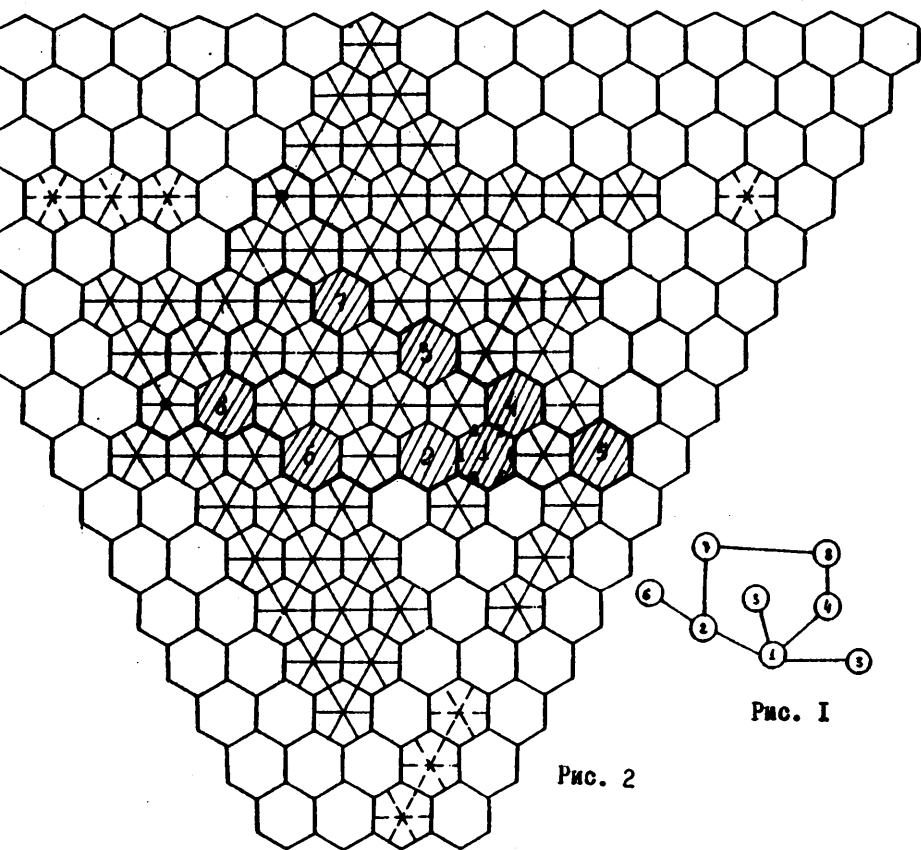


Рис. 2

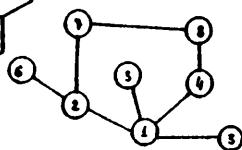


Рис. 1

Доказательство теоремы 4 опирается на лемму и аналогично доказательству леммы.

2.0.3. Теорема 4 даёт возможность упростить алгоритм \tilde{A} . В этом случае алгоритм \tilde{A} приведения программы M , полученной в результате некоторого шага работы алгоритма A^2 , к правильной сводится к следующему алгоритму \tilde{A}_0 . Если элементу $m(i_a, j_a) \in K_{I_a, j_a}(M)$ ставится в соответствие функция $\varphi \in B_{BC}$ и $\varphi \neq D \in S$, то всем элементам колонны $K_{J_a, b}(M)$ с координатами (i, j) , где $i = i_a$, $j_{\min} \leq j \leq j_{\max}$, $j \neq j_a$, ставится в соответствие функция $D \in S$, причем j_{\min}, j_{\max} — предельные значения координат элементов программы M . Для элемента $m(i_a, j_a) \in K_{J_a, b}(M)$ приведение программы к правильной производится аналогично при замене i на j и наоборот.

2.1.0. Алгоритм A^2 можно улучшить в смысле уменьшения числа элементов программы, например, следующим образом:

а) после пп.1а введем

пп.1а'. По коду л.с. L находим уже размещённые " r "-тые вершины G_L , смежные вершине " a " ($r \neq k$) (если таких вершин нет, то переход к пп.1б).

пп.1а". Для элементов $m(i_k, j_k)$ и $m(i_r, j_r)$ по коду S находим смежные элементы m'_k и m'_r , принадлежащие $\Gamma_{CB}(M')$.

пп.1а". В программе M' определяем элемент $m_c = K^k \cap K^r$, где K^k — колонна, содержащая элементы $m(i_k, j_k)$ и m'_k ; K^r — колонна, содержащая элементы $m(i_r, j_r)$ и m'_r . Если элемент m_c лежит между элементами $m(i_r, j_r)$ и m'_r и между элементами $m(i_k, j_k)$ и m'_k , то элементу m_c вместо ранее поставленной в соответствие функции $D \in S$ ставим в соответствие функцию φ_a (если такого элемента m_c не существует, то переход к пп.1б).

б) после пп.2б введем

пп.2б'. В программе \tilde{M}' определяем элемент $m_c = K^a \cap K^r$ где K^a — колонна, содержащая элементы $m(i_a, j_a)$ и m'_a ,

K^r — колонна, содержащая элементы $m(i_r, j_r)$ и m'_r .

Если элемент m_c лежит между элементами $m(i_a, j_a)$ и m'_a и между элементами $m(i_r, j_r)$ и m'_r , то элементу m_c вместо ранее поставленной в соответствие функции $D \in S$ ставим в соответствие функцию $\varphi \in S$ (если такого элемента m_c нет, то переход к пп.2г).

Этот улучшенный алгоритм A_0^2 в качестве результата работы дает также правильную программу \tilde{M}^0 . На рис. 4 показана программа \tilde{M}_L , а на рис. 5 - \tilde{M}_L^0 , соответствующие логической сети L (рис. 3).

Очевидно, что алгоритм A^2 можно улучшать и далее, например, учитывая степень каждой вершины графа G_L .

3.0.0 Алгоритм A_1 соединения правильных программ.

Пусть данная логическая сеть L разбита на множество подсетей $\{L_1, \dots, L_p, \dots, L_m\}$ и для каждой логической сети L_p построена с помощью алгоритма A_0^2 правильная программа $\tilde{M}_{L_p}^0$, причем в каждой программе $\tilde{M}_{L_p}^0$ отмечены элементы, которые необходимо соединить с элементами других программ $\tilde{M}_{L_q}^0$, $p \neq q$.

Пусть $N(\tilde{M}_{L_p}^0, \tilde{M}_{L_q}^0)$ - список всех пар отмеченных элементов программы $\tilde{M}_{L_p}^0$ и $\tilde{M}_{L_q}^0$, которые необходимо соединить между собой.

В качестве начальной исходной программы \tilde{M}_{L_p}' возьмем программу $\tilde{M}_{L_p}^0$. Предельные значения координат элементов некоторой программы \tilde{M} будем обозначать следующим образом: $\max i(\tilde{M})$, $\min i(\tilde{M})$, $\max j(\tilde{M})$, $\min j(\tilde{M})$.

п.1. Выбираем k -тый элемент $m_k \in \tilde{M}_{L_p}'$ списка $N(\tilde{M}_{L_p}', \tilde{M}_{L_{p+1}}^0)$.

пп.1а. Проверяем существует ли элемент $m_r^1 \in K_{\max i(\tilde{M}_{L_p}')} смежный элементу m_k ; если m_r^1 существует, то элементу m_r^1 присваиваем отметку элемента m_k и заносим в список $N_1(\tilde{M}_{L_p}', \tilde{M}_{L_{p+1}}^0)$.$

пп.1б. Проверяем существование элемента $m_r^2 \in K_{\min j(\tilde{M}_{L_p}')} смежного элементу m_k .$

Если m_r^2 существует, то ему присваиваем отметку элемента m_k и заносим в список $N_2(\tilde{M}_{L_p}', \tilde{M}_{L_{p+1}}^0)$.

пп.1в. Если элементу m_k смежен элемент m_s , принадлежащий колонне $K_{\text{сб} \max j(\tilde{M}_{L_p}')} (или $K_{\text{сб} \min i(\tilde{M}_{L_p}')})$, то строим$

путь (см. алгоритм A_0^2 пп.2г) между элементом m_s и элементом



$$m = K_{св} \max J(\tilde{M}'_{L_p}) \cap K_{\max} I(\tilde{M}'_{L_p})$$

(или

$$m = K_{св} \min I(\tilde{M}'_{L_p}) \cap K_{\min} J(\tilde{M}'_{L_p}).$$

Элементу m ставим в соответствие функцию $D \in S$, если m не совпадает с m_a , и функцию $P \in S$, если m совпадает с m_a ; присваиваем отметку элемента m_k и заносим в список $N_1(\tilde{M}'_{L_p}, \tilde{M}^0_{L_{p+1}})$ (или $N_2(\tilde{M}'_{L_p}, \tilde{M}^0_{L_{p+1}})$).

пп.1г. Полученную программу приводим к правильной (алгоритм \tilde{A}_0) и считаем её исходной программой \tilde{M}'_{L_p} . Этот пункт выполняется для всех k от 1 до n' , где n' - число пар списка $N(\tilde{M}'_{L_p}, \tilde{M}^0_{L_{p+1}})$.

п.2. Для каждого элемента m_k списка $N_1(\tilde{M}'_{L_p}, \tilde{M}^0_{L_{p+1}})$ определяем соответствующий элемент m_c программы $\tilde{M}'_{L_{p+1}}$ (в начале работы п.2 программа $\tilde{M}'_{L_{p+1}}$ совпадает с $\tilde{M}_{L_{p+1}}$).

пп.2а. Если элемент m , смежный элементу m_c , принадлежит колонне $K_{\max} J(\tilde{M}'_{L_{p+1}})$, то ему присваиваем отметку элемента m_c ; если такого элемента нет, то между элементом m_a , смежным элементу m_c и элементом

$$m = K_{св} \max I(\tilde{M}'_{L_{p+1}}) \cap K_{\max} J(\tilde{M}'_{L_{p+1}})$$

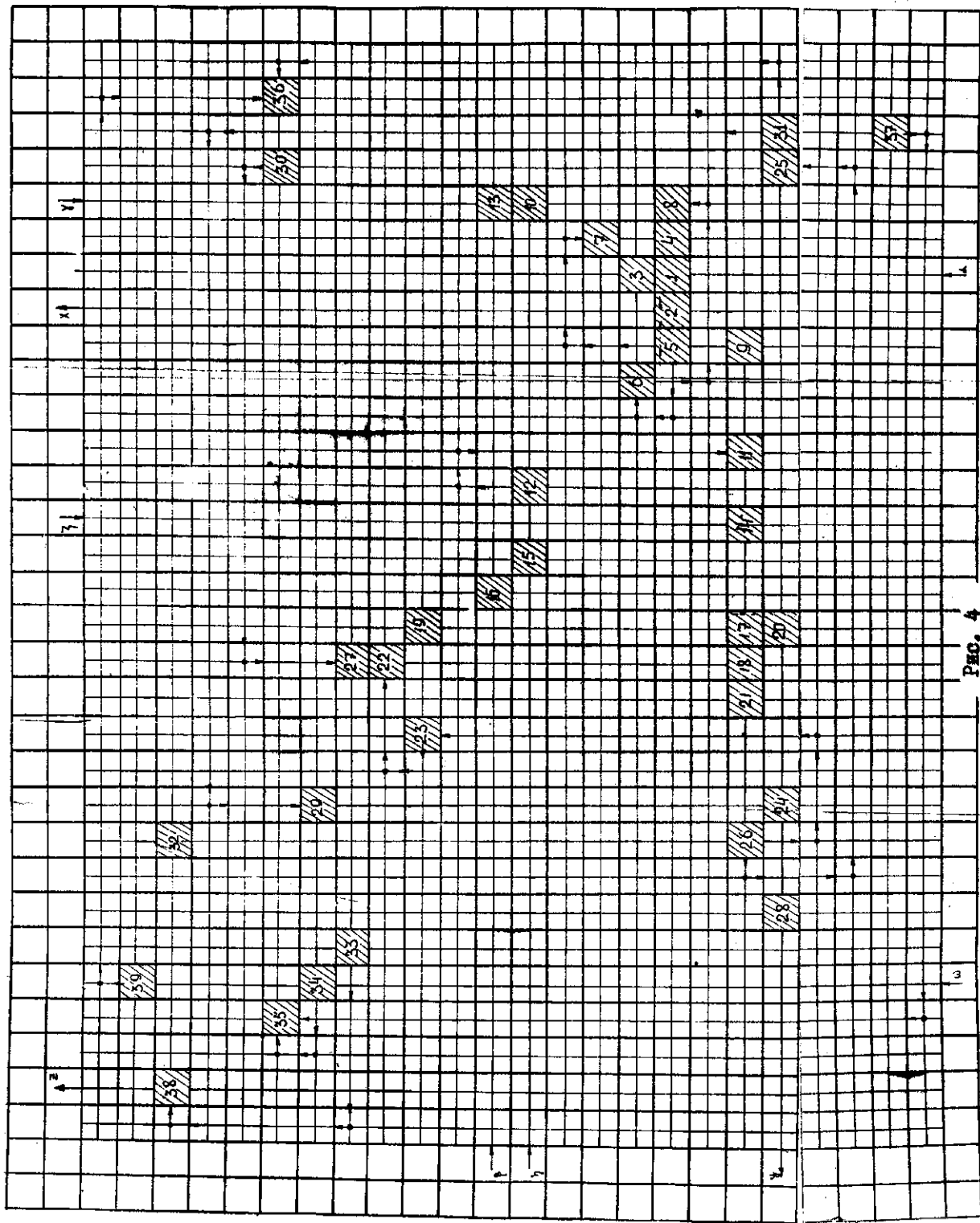
(или элементом

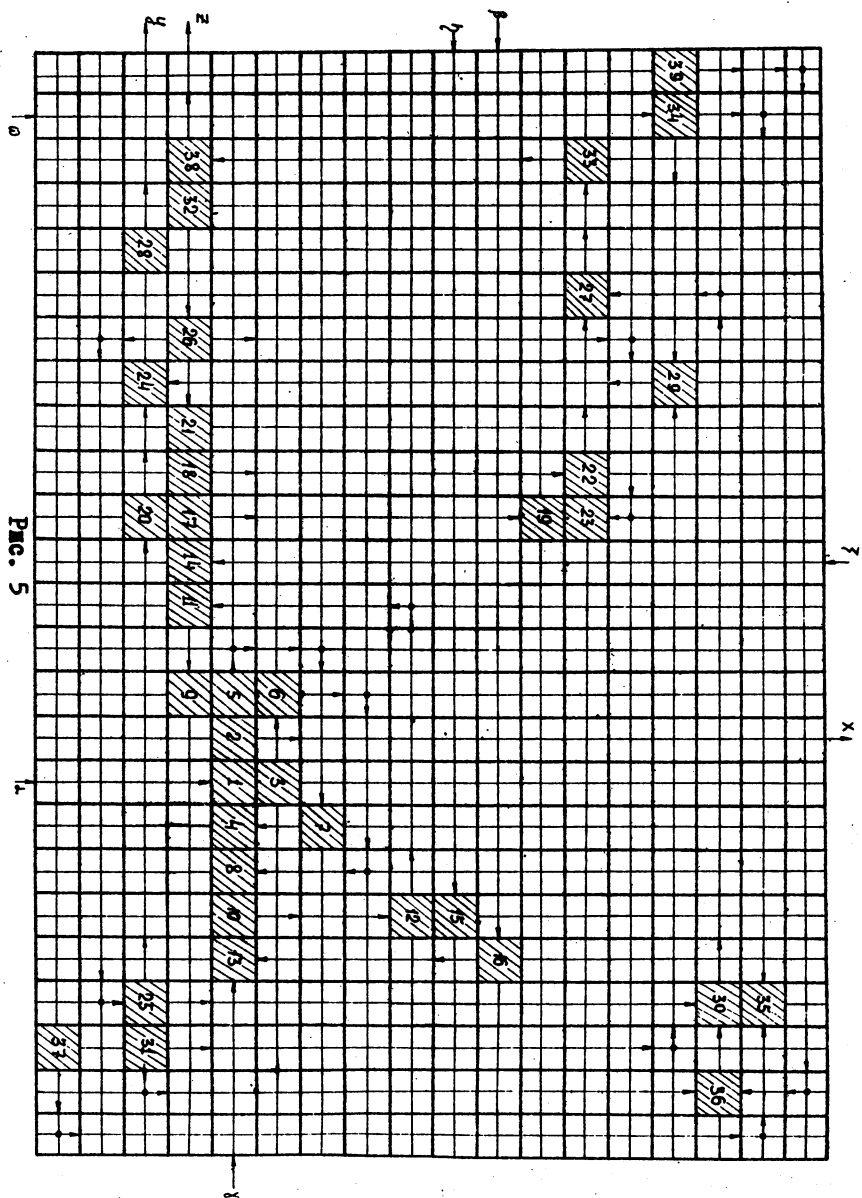
$$m = K_{св} \min I(\tilde{M}'_{L_{p+1}}) \cap K_{\max} J(\tilde{M}'_{L_{p+1}})$$

в соответствии с требованием минимальной длины пути) строим путь (см. алгоритм \tilde{A}_0 пп.2г). Элементу m ставим в соответствие функцию $D \in S$, если m не совпадает с m_a и функцию $P \in S$ если m совпадает с m_a ; присваиваем отметку элемента m_c и заносим в список $N_1(\tilde{M}'_{L_{p+1}}, \tilde{M}'_{L_p})$. Полученную программу приводим к правильной.

пп.2б. Аналогичная процедура применяется для каждого элемента списка $N_2(\tilde{M}'_{L_p}, \tilde{M}^0_{L_{p+1}})$ с заменой I на J , \min на \max и наоборот, и занесением элемента m в список $N_2(\tilde{M}^0_{L_{p+1}}, \tilde{M}'_{L_p})$.

п.3. Увеличиваем координаты (i, j) элементов програм-





мы $\tilde{M}'_{L'_{p+1}}$ на величины:

$$\Delta i = \max i(\tilde{M}'_{L'_p}) - \min i(\tilde{M}'_{L'_{p+1}}) + 1,$$

$$\Delta j = \min j(\tilde{M}'_{L'_p}) - \max j(\tilde{M}'_{L'_{p+1}}) - 1,$$

соответственно. Приводим программу $M = \tilde{M}'_{L'_p} \cup \tilde{M}'_{L'_{p+1}}$ к правильной: всем элементам BC^2 , имеющим координаты

$$\min i(\tilde{M}'_{L'_{p+1}}) \leq i \leq \max i(\tilde{M}'_{L'_{p+1}})$$

и

$$\min j(\tilde{M}'_{L'_p}) \leq j \leq \max j(\tilde{M}'_{L'_p})$$

и всем элементам, имеющим координаты

$$\min i(\tilde{M}'_{L'_p}) \leq i \leq \max i(\tilde{M}'_{L'_p})$$

и

$$\min j(\tilde{M}'_{L'_{p+1}}) \leq j \leq \max j(\tilde{M}'_{L'_{p+1}})$$

ставим в соответствие функцию $D \in S$.

п.4. Для каждого элемента $m(i_1, j_1)$ списка $N_1(\tilde{M}'_{L'_p}, \tilde{M}'_{L'_{p+1}})$ находим соответствующий ему элемент $m(i_2, j_2)$ списка $N_1(\tilde{M}'_{L'_{p+1}}, \tilde{M}'_{L'_p})$. Элементу $m(i_2, j_1)$ ставим в соответствие функцию $P \in S$. Для каждого элемента $m(i_1, j_1)$ списка $N_2(\tilde{M}'_{L'_p}, \tilde{M}'_{L'_{p+1}})$ находим соответствующий ему элемент $m(i_2, j_2)$ списка $N_2(\tilde{M}'_{L'_{p+1}}, \tilde{M}'_{L'_p})$. Элементу $m(i_1, j_2)$ ставим в соответствие функцию $P \in S$. Полученную программу, которая, очевидно, имеет вид прямоугольника, считаем исходной, заменяя индекс "p" на "p + 1".

Если $p + 1 < m$, то переход к п.1.

Если $p + 1 = m$, то конец.

Результатом работы алгоритма A_1 является правильная программа $\tilde{M}'_{L'_1}$, соответствующая логической сети L . Программа $\tilde{M}'_{L'_1}$, соответствующая логической сети L (рис.3), разбитой на подсети L_1, L_2, L_3 (рис.6), показана на рис. 7.

Для логической сети L , граф G_L которой имеет $n = 39$ вершин, число N элементов программы $\tilde{M}'_{L'_1}, \tilde{M}'_{L'_2}, \tilde{M}'_{L'_3}$ соответственно равно: $N_{\tilde{M}'_{L'_1}} = 744$, $N_{\tilde{M}'_{L'_2}} = 450$, $N_{\tilde{M}'_{L'_3}} = 588$.

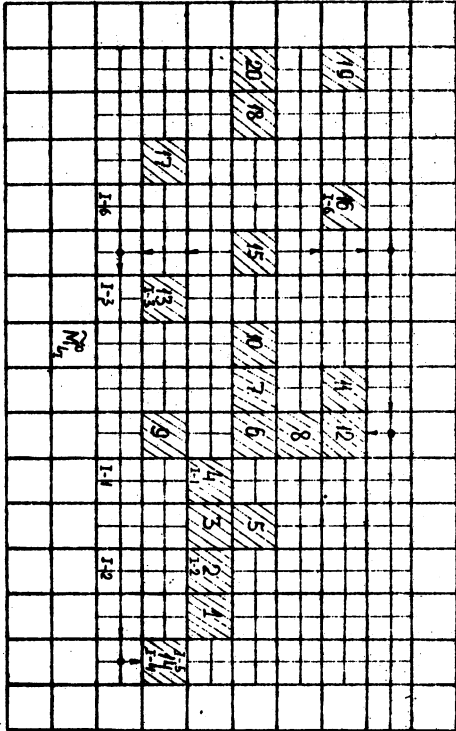
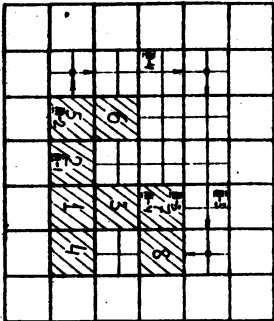
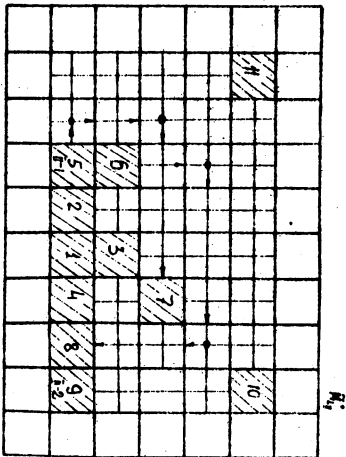


FIG. 6



4.0.0. Оценка сложности правильных программ в BC^2

при $d_{BC} = 4$

Назовем сложностью $N(M)$ программы M — мощность множества её элементов.

Пусть дана л.с. L . Построим граф G_L логической сети. Пусть число вершин графа G_L равно n .

Оценка сложности программ \tilde{M}_L , \tilde{M}_L^0 затруднительна, поэтому сделаем оценку сложности правильной программы \tilde{M}_L' , которая может быть получена с помощью алгоритма A' , имеющего в сокращенной записи следующий вид:

п.1. Согласно кодам л.с. L и BC размещаем все n вершин графа G_L , приводя на каждом шаге программу к правильной.

п.2. Строим все оставшиеся дуги графа G_L , приводя на каждом шаге программу к правильной.

Легко показать, что программа \tilde{M}_L' также имеет вид прямоугольника.

4.1.0. Очевидно, что максимальное число дуг $m = 2n$ имеет однородный граф G_L^0 степени 4, имеющий n вершин, поэтому оценка максимальной сложности программы \tilde{M}_L' может быть получена для л.с. L^0 , которой соответствует граф G_L^0 .

Оценим сложность $N_1(\tilde{M}'_{1L^0})$ программы \tilde{M}'_{1L^0} , полученной после выполнения п.1 алгоритма A' .

4.1.1. Пусть после некоторого шага выполнения п.1 алгоритма A' размещено в BC 1 вершин G_L^0 в прямоугольнике со сторонами a_1 и b_1 , т.е. число элементов BC N_1 , необходимых для размещения 1 вершин G_L^0 : $N_1 = a_1 \cdot b_1$.

Нетрудно показать, что на следующем шаге выполнения п.1 либо $N_{1+1} = (a_1 + 1)(b_1 + 2)$, либо $N_{1+1} = (a_1 + 2)(b_1 + 1)$.

Рассмотрим таблицу I:

1	a_1	b_1
1	1	1
5	3	3
8	5	4
II	6	6
14	8	7
17	9	9
...

Из таблицы следует, что при n — четном

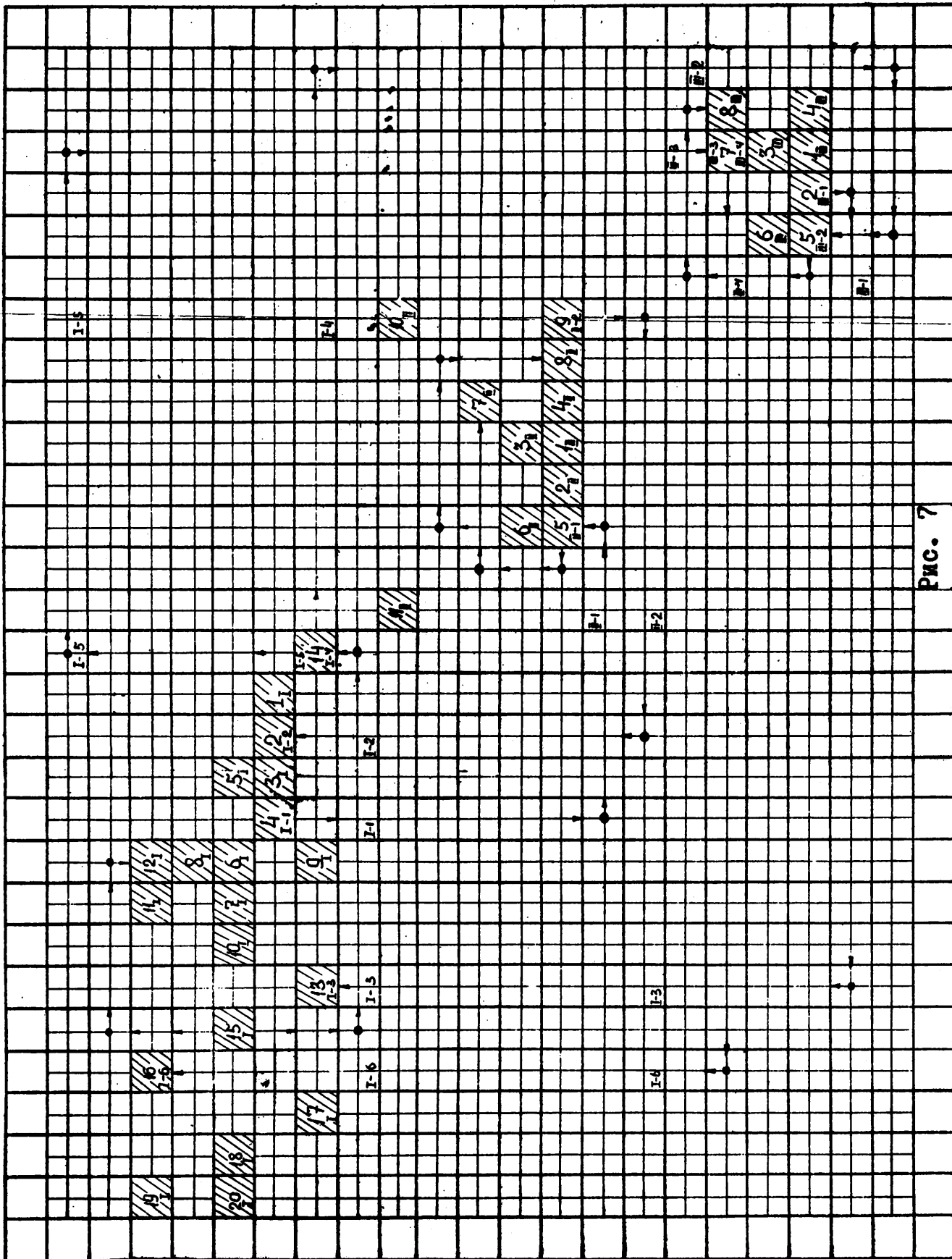
$$a = b + 1, b_n = \frac{n}{2},$$

при n — нечетном

$$a_n = b_n = \frac{n+1}{2}.$$

Таким образом

$$N_1(\tilde{M}'_{1L^0}) = \begin{cases} \frac{n(n+2)}{4}, & n - \text{четное}, \\ \frac{(n+1)^2}{2}, & n - \text{нечетное}. \end{cases}$$



4.1.2. После выполнения п. I необходимо построить еще $(n + 1)$ дугу графа G_{L^0} . Рассмотрим два случая (в предположении, что все пути имеют максимальную длину, т.е. соединяют элементы, принадлежащие противоположным сторонам программы).

а) n - нечетное, т.е.

$$a_n = b_n = \frac{n + 1}{2}.$$

Нетрудно показать, что после построения k -той дуги G_{L^0} число элементов ВС, занятых программой, $N_k = (a_n + c_k)(b_n + d_k)$, причем c_k и d_k можно определить из таблицы П

Т а б л и ц а П

k	c_k	d_k
1	1	2
2	3	3
3	4	5
4	6	6
5	7	8
6	9	9
7	10	11
\vdots	\vdots	\vdots

Поскольку n - нечетное, то для $k = n + 1$ четного из таблицы П следует, что

$$c_{n+1} = d_{n+1} = \frac{3(n + 1)}{2}.$$

Таким образом для нечетного n

$$N(\tilde{M}'_{L^0}) = (a_n + c_{n+1})(b_n + d_{n+1}) = 4(n + 1)^2.$$

б) n - четное, т.е.

$$a_n = \frac{n + 2}{2}, \quad b_n = \frac{n}{2}.$$

В этом случае также $N_k = (a_n + c_k)(b_n + d_k)$.

Из таблицы П следует, что для нечетного $k = (n + 1)$

$$c_{n+1} = \frac{3n}{2} + 1, \quad d_{n+1} = \frac{3n}{2} + 2.$$

Отсюда для четного n

$$N(\tilde{M}'_{L^0}) = (a_n + c_{n+1})(b_n + d_{n+1}) = 4(n + 1)^2.$$

Таким образом максимальная сложность программы \tilde{M}'_{L_0} , полученной алгоритмом A' , $N(\tilde{M}'_{L_0}) \rightarrow c \cdot n^2$ при $n \rightarrow \infty$ где $c = 4$.

4.1.3. Зависимость константы c от параметров ВС (например, от базиса B_{BC} ориентации θ и т.д.) в данной работе не рассматривается. Очевидно, что минимальная сложность $N(\tilde{M}'_{L_0}) = n$, если G_L содержит n вершин и $(n - 1)$ дугу.

Таким образом

$$n \leq N(\tilde{M}'_{L_0}) \leq c \cdot n^2.$$

Л и т е р а т у р а

1. Б.А. Розенфельд. Многомерные пространства, М., "Наука", 1966.
2. Б. Делоне, Н. Падуров, А. Александров. Математические основы структурного анализа кристаллов, ОНТИ, 1934.
3. В.В. Евреинов, Ю.Г. Косарев. Однородные универсальные системы высокой производительности. Наука, Новосибирск, 1967.
4. C.Y.Lee. An algorithm for path connections and its applications. - IRE Transaction, vol.BC-10, N 3.
5. Н.Е. Кобринский, Б.А. Трахтенброт. Введение в теорию конечных автоматов. Ф.М., 1962.
6. В.М. Глушков. Синтез цифровых автоматов, 1962.