Глава 4

ПРОЕКТИРОВАНИЕ СПЕЦПРОЦЕССОРОВ С ИСПОЛЬЗОВАНИЕМ РАСЩЕПЛЕНИЯ И РАЗДЕЛЬНОГО ТАЙМИРОВАНИЯ МАКРООПЕРАЦИЙ АЛГОРИТМА

4.1. РАСЩЕПЛЕНИЕ МАКРООПЕРАЦИЙ

В этой главе для удобства дальнейшего изложения изменим обозначения в алгоритме (3.1): вершину v заменим на $\overline{v}$, векторы $\varphi^{(i,j)}$ – на $\overline{\varphi}^{(i,j)}$, множества Φ, V_λ и V – соответственно на $\overline{\Phi}$, $\overline{V}_\lambda$, и $\overline{V}$. Таким образом, будем рассматривать алгоритмы вида

$$\begin{gathered} x^{(k)}(\overline{v}) = F_\lambda^{(k)}(x^{(1)}(\overline{v} - \overline{\varphi}^{(1,k)}), \ldots, x^{(K)}(\overline{v} - \overline{\varphi}^{(K,k)})), \\ 1 \le k \le K, \quad 1 \le \lambda \le \Lambda, \quad K, \Lambda \in \mathbf{N}, \quad \overline{v} \in \overline{V}_\lambda \subset \mathbf{Z}^n. \end{gathered} \tag{4.1}$$

Каждой операции $F_\lambda^{(k)}$ вычисления переменной $x^{(k)}$ в точке $\overline{v} \in \overline{V}_\lambda \subset \mathbf{Z}^n$ поставим во взаимно однозначное соответствие точку $v = (\overline{v}, \widehat{v}^{(k)}) = (i_1, \ldots, i_n, i_{n+1}, \ldots, i_{n+p}) \in \mathbf{Z}^{n+p}$ – вершину графа $G = (V, E)$, размещенного в пространстве $\mathbf{Z}^{n+p}$. Первые n целочисленных координат точки $v \in V$ совпадают с координатами точки $\overline{v} = (i_1, \ldots, i_n) \in \overline{V}_\lambda$, а последние p координат $\widehat{v}^{(k)} = (i_{n+1}, \ldots, i_{n+p})$ являются индивидуальными для каждой переменной $x^{(k)}$. Множество из $K \ge 2$ векторов $\widehat{v}^{(k)} \in \mathbf{Z}^p$ будем обозначать $\widehat{V}$. Для определенности будем полагать $p = \lceil \log_2 K \rceil$, а в качестве координат вектора $\widehat{v}^{(k)}$ будем брать коэффициенты p-разрядного двоичного представления числа k. Полученное таким образом множество $V \in \mathbf{Z}^{n+p}$ вершин графа $G = (V, E)$ алгоритма (4.1), размещенного в пространстве $\mathbf{Z}^{n+p}$, имеет вид $V = \{ (\overline{v}, \widehat{v}^{(k)}) \in \mathbf{Z}^{n+p} \mid \overline{v} \in \overline{V},\ \widehat{v}^{(k)} \in \widehat{V},\ 1 \le k \le K \}$. Обозначим $V_\lambda^{(k)} = \{ (\overline{v}, \widehat{v}^{(k)}) \in V \mid \overline{v} \in \overline{V}_\lambda,\ \widehat{v}^{(k)} \in \widehat{V} \}$, $1 \le \lambda \le \Lambda$, $1 \le k \le K$, $V^{(k)} = \bigcup\limits_{\lambda=1}^{\Lambda} V_\lambda^{(k)}$. Тогда $V = \bigcup\limits_{k=1}^{K} V^{(k)}$, $|V| = K\ |\overline{V}|$, где $|V|$ – число точек множества V.

Расщепление макроопераций приводит к разделению множества дуг

E графа $G = (V, E)$ алгоритма (4.1) на два подмножества, определяемые соответственно двумя подмножествами векторов зависимостей. Векторы множества $\Phi' = \{\, (\overline{\varphi}^{(k,k)}, 0) \in \mathbf{Z}^{n+p} \mid \overline{\varphi}^{(k,k)} \in \overline{\Phi},\ 1 \le k \le K \,\}$ определяют дуги, связывающие две информационно зависимые вершины (операции), принадлежащие одному множеству $V^{(k)}$, а векторы множества $\Phi'' = \{(\overline{\varphi}^{(k_1,k_2)}, \widehat{\varphi}^{(k_1,k_2)}) \in \mathbf{Z}^{n+p} :\ \overline{\varphi}^{(k_1,k_2)} \in \overline{\Phi},\ \widehat{\varphi}^{(k_1,k_2)} = \widehat{v}^{(k_2)} - \widehat{v}^{(k_1)},\ \widehat{v}^{(k_1)}, \widehat{v}^{(k_2)} \in \widehat{V},\ 1 \le k_1, k_2 \le K,\ k_1 \ne k_2\}$ определяют дуги, связывающие две информационно зависимые вершины (операции), принадлежащие двум разным множествам $V^{(k_1)}$ и $V^{(k_2)}$, $k_1 \ne k_2$.

Обозначим через $\Phi = \Phi' \cup \Phi''$ множество векторов зависимостей, определяющих дуги графа $G = (V, E)$. Тогда $|\Phi| = |\Phi'| + |\Phi''| = |\overline{\Phi}|$. Таким образом, при переходе к мелкозернистой графовой модели алгоритма (4.1) число вершин графа алгоритма увеличивается в K раз, а число дуг (векторов зависимостей) остается неизменным.

Минимизация высоты параллельной формы

Пусть граф G алгоритма (4.1) строго направлен: $K(G) = \{\tau \in \in \mathbf{Z}^{n+p} \mid \tau \cdot \varphi^{(i,j)} > 0,\ \varphi^{(i,j)} \in \Phi\} \ne \emptyset$. Тогда функция $t(v) = \tau \cdot v + t_0$, где $t_0 \in \mathbf{Z},\ \tau \in K(G)$, – направляющий вектор графа G (мелкозернистой модели алгоритма), определяет параллельную форму алгоритма.

Получим условия, при выполнении которых переход к мелкозернистой модели позволяет уменьшить высоту параллельной формы алгоритма в случае, когда V является $(n+p)$-мерным параллелепипедом $V = \{v(i_1, \ldots, i_{n+p}) \in \mathbf{Z}^{n+p} \mid \alpha_k \le i_k \le \beta_k,\ 1 \le k \le n+p\}$.

Пусть $\overline{G}$ – граф зависимостей алгоритма (4.1), размещенный в пространстве $\mathbf{Z}^n$, $K(\overline{G})$ – конус допустимых направлений графа $\overline{G}$:

$$K(\overline{G}) = \{\overline{\tau}(\overline{\tau}_1, \ldots \overline{\tau}_n) \in \mathbf{Z}^n \mid \sum_{k=1}^{n} \overline{\tau}_k \cdot \overline{\varphi}_k^{(i,j)} > 0, \quad \forall \overline{\varphi}^{(i,j)} \in \overline{\Phi}\} \ne \emptyset.$$

Обозначим через $\overline{\tau}^{\min} = (\overline{\tau}_1^{\min}, \ldots, \overline{\tau}_n^{\min}),\ \overline{\tau}^{\min} \cdot \overline{\varphi}^{(i,j)} > 0,\ \overline{\varphi}^{(i,j)} \in \overline{\Phi}$, вектор таймирования $t(\overline{v}) = \overline{\tau}^{\min} \cdot \overline{v} + t_0^{\min},\ \overline{v} \in \overline{V}$, определяющий параллельную форму алгоритма (4.1) минимальной высоты $T_0 = \min\limits_{\overline{\tau} \in K(\overline{G})} \max\limits_{\overline{v} \in \overline{V} \ominus \overline{V}} \overline{\tau} \cdot \overline{v} + 1$, которую можно получить, не переходя к мелкозернистой модели алгоритма.

Если область V – $(n+p)$-мерный параллелепипед, то $\overline{V} \ominus \overline{V} = \{\overline{v}(i_1,\ldots,i_n) \in \mathbf{Z}^n \mid |i_k| \le \beta_k - \alpha_k,\ 1 \le k \le n\}$ и $T_0 = \sum\limits_{k=1}^{n} |\overline{\tau}_k^{\min}|(\beta_k - \alpha_k) + 1$.

Предположим, что существуют такие $k_0 \in \{1,2,\ldots,n\}$ и $k_1 \in \{n+1, n+2, \ldots, n+p\}$, и такое целое число $\tau_{k_1} \neq 0$, которые удовлетворяют системе неравенств:

$$\begin{gathered} \sum_{k_0 \neq k=1}^{n} \overline{\tau}_k^{\min} \varphi_k^{(i,j)} > -\tau_{k_1} \varphi_{k_1}^{(i,j)} \\ \text{для всех } \varphi^{(i,j)} = (\varphi_1^{(i,j)}, \ldots, \varphi_{n+p}^{(i,j)}) \in \Phi, \\ |\overline{\tau}_{k_0}^{\min}|(\beta_{k_0} - \alpha_{k_0}) > |\tau_{k_1}|(\beta_{k_1} - \alpha_{k_1}). \end{gathered} \tag{4.2}$$

Тогда вектор $\tau = (\overline{\tau}_1^{\min}, \ldots, \overline{\tau}_{k_0-1}^{\min}, 0, \overline{\tau}_{k_0+1}^{\min}, \ldots, \overline{\tau}_n^{\min}, 0, \ldots, 0, \tau_{k_1}, 0, \ldots, 0\} \in K(G)$ определяет распараллеливание с таймирующей функцией $t(v) = \tau \cdot v + t_0,\ t_0 \in \mathbf{Z},\ v \in V$, причем высота ПФ $|T|$ при таком распараллеливании удовлетворяет неравенству $|T| < |T_0|$.

Действительно, в силу неравенства (4.2) имеем $\tau \in K(G)$. Кроме того, снова в силу (4.2), $|T| = \sum\limits_{k_0 \neq k=1}^{n} |\overline{\tau}_k^{\min}|(\beta_k - \alpha_k) + |\tau_{k_1}|(\beta_{k_1} - \alpha_{k_1}) + 1 < |T_0|$.

Пример 4.1. Рассмотрим алгоритм, заданный следующей системой рекуррентных уравнений:

$$\begin{aligned} &1 \le i,j \le N, \\ &a(i,j) = a(i-1,j) + b(i,j-1), \\ &b(i,j) = a(i-1,j) + b(i-1,j+1). \end{aligned}$$

В данном случае $n=2,\ K=2,\ \overline{V} = \{(i,j) \in \mathbf{Z}^2 \mid 1 \le i,j \le N\},\ \Phi = \{\overline{\varphi}^{(1,1)} = (1,0),\ \overline{\varphi}^{(2,1)} = (0,1),\ \overline{\varphi}^{(1,2)} = (1,0),\ \overline{\varphi}^{(2,2)} = (1,-1)\},\ \overline{\tau}^{\min} = (2,1),\ |T_0| = 3N-2$. Полагая $p=1,\ \widehat{v}^{(1)} = 0,\ \widehat{v}^{(2)} = 1$, получаем $V = \{(i,j,k) \in \mathbf{Z}^3 \mid 1 \le i,j \le N,\ 0 \le k \le 1\},\ \varphi^{(1,1)} = (1,0,0),\ \varphi^{(2,1)} = (0,1,-1),\ \varphi^{(1,2)} = (1,0,1),\ \varphi^{(2,2)} = (1,-1,0)$. При $k_0 = 2,\ k_1 = 3$ система неравенств (4.2) имеет вид $2 > -\tau_3 \cdot 0,\ 0 > -\tau_3 \cdot (-1),\ 2 > -\tau_3,\ 2 > -\tau_3 \cdot 0$. В силу целочисленности τ_3 получим $\tau_3 = -1$. Таким образом, $\tau = (2,0,-1),\ |T| = 2N$. Высота параллельной формы алгоритма уменьшилась с $3N-2$ до $2N$.

Получение независимых разбиений

Пусть r – число линейно независимых векторов множества Φ, причем $r < n+p$. Тогда существует $n+p-r$ линейно независимых векторов $\psi^{(1)}, ..., \psi^{(n+p-r)} \in \mathbf{Z}^{n+p}$ со взаимно простыми координатами, ортогональных всем векторам зависимостей: $\psi^{(q)} \cdot \varphi = 0,\ \varphi \in \Phi,\ 1 \le q \le n+p-r$. Эти векторы определяют разбиение всех операций алгоритма на независимые части (расщепление области V гиперплоскостями с нормальными векторами $\psi^{(1)}, ..., \psi^{(n+p-r)}$):

$$V = \bigcup_{y \in Y} V_y,$$

$$Y = \{y_1, ..., y_{n+p-r} \in \mathbf{Z}^{n+p-r} \mid \min_{v \in V} \psi^{(q)} \cdot v \le y_q \le \max_{v \in V} \psi^{(q)} \cdot v,$$

$$1 \le q \le n+p-r\},$$

$$V_y = \{v \in V \mid \psi^{(q)} \cdot v = y_q,\ \ 1 \le q \le n+p-r\},\ y \in Y.$$

Переход к мелкозернистой модели не может увеличить число $\overline{r}$ линейно независимых векторов в $\overline{\Phi}$ более чем на p, т. е. $r \le \overline{r} + p$; следовательно, $n+p-r \ge n-\overline{r}$. Это означает, что если существовало расщепление области $\overline{V}$ до перехода к мелкозернистой модели, то оно будет существовать и после, но разбиение операций алгоритма на независимые части может оказаться возможным при переходе к мелкозернистой модели и в том случае, когда в исходной модели независимых разбиений не существовало.

Пример 4.2. Рассмотрим следующий алгоритм:

$$1 \le i, j \le N,$$
$$a(i,j) = b(i,j-1),$$
$$b(i,j) = a(i-1,j) + b(i-1,j-1).$$

В данном случае $n = 2,\ K = 2,\ \overline{V} = \{(i,j) \in \mathbf{Z}^2 \mid 1 \le i,j \le N\},\ \overline{\varphi}^{(2,1)} = (0,1),\ \overline{\varphi}^{(1,2)} = (1,0),\ \overline{\varphi}^{(2,2)} = (1,1)$. Так как $r = n = 2$, то алгоритм не разбивается на независимые части.

Перейдем к мелкозернистой модели, положив $p = 1,\ \widehat{v}^{(1)} = 0,\ \widehat{v}^{(2)} = 1$. Получим $V = \{(i,j,k) \in \mathbf{Z}^3 \mid 1 \le i,j \le N,\ 0 \le k \le 1\},\ \Phi = \{(0,1,-1),\ (1,0,1),\ (1,1,0)\}$. Таким образом, $r = 2$ и $r < n+p$, поэтому алгоритм допускает разбиение на независимые части. Нормальный вектор семейства параллельных плоскостей, осуществляющих расщепление расширенной области вычислений V, имеет вид $\psi^{(1)} = (-1,1,1)$. Он

ортогонален всем векторам зависимостей: $\psi^{(1)} \cdot \varphi = 0,\ \varphi \in \Phi$. Определив $Y = \{(y_1) \in \mathbf{Z}^1 \mid \min\limits_{\substack{1 \le i,j \le N \\ 0 \le k \le 1}} (-i+j+k) \le y_1 \le \max\limits_{\substack{1 \le i,j \le N \\ 0 \le k \le 1}} (-i+j+k)\} = \{(y_1) \in \mathbf{Z}^1 \mid -N+1 \le y_1 \le N\}$, получим разбиение $2N^2$ операций алгоритма на $2N$ независимых частей: $V = \bigcup_{y_1=-N+1}^{N} V_{y_1}$, $V_{y_1} = \{v(i,j,k) \in V \mid -i+j+k = y_1\}$.

На рис. 4.1 и 4.2 изображены графы зависимостей рассматриваемого алгоритма соответственно до и после расщепления.

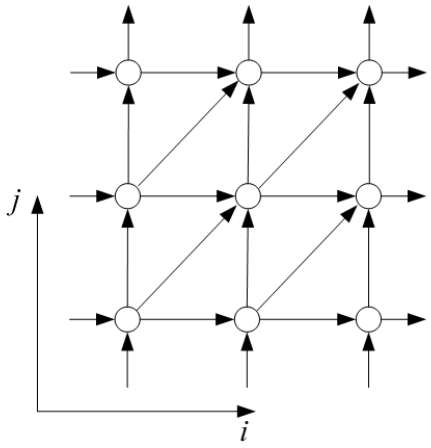

Рис. 4.1. Граф зависимостей алгоритма до расщепления

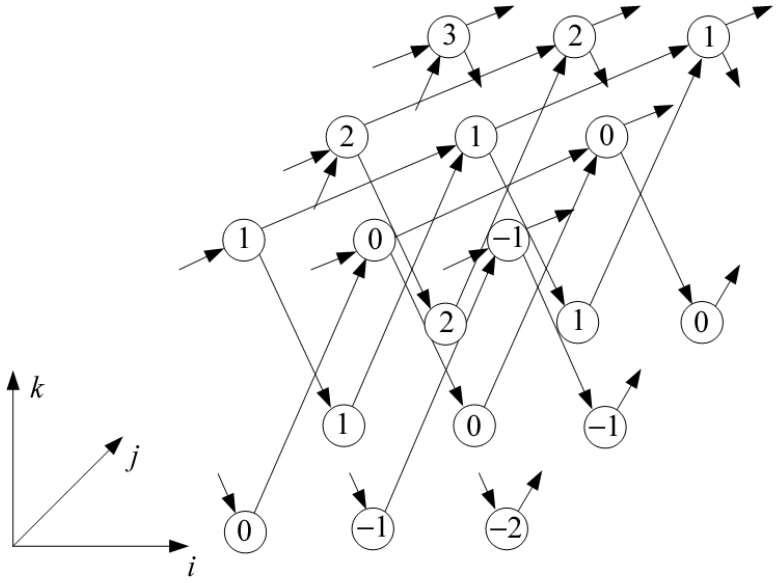

Рис. 4.2. Граф зависимостей алгоритма после расщепления

4.2. РАЗДЕЛЬНОЕ ТАЙМИРОВАНИЕ

Поставим в соответствие каждой операции $F_\lambda^{(k)}$ по вычислению переменной $x^{(k)},\ 1 \le k \le K$, алгоритма (4.1) в точке $\overline{v} \in \overline{V}_\lambda$ целое число – значение таймирующей функции:

$$t^{(k)}(v) = \tau^{(k)} \cdot v, \quad v = (\overline{v}, \widehat{v}^{(k)}) \in V_\lambda^{(k)}, \\ 1 \le \lambda \le \Lambda, \quad \tau^{(k)} \in \mathbf{Z}^{n+p}. \tag{4.3}$$

Тем самым каждой переменной алгоритма (4.1) соответствует своя таймирующая функция (раздельное таймирование). Значение $t^{(k)}(v) \in \mathbf{Z}$ можно интерпретировать как номер такта, на котором вычисляется значение $x^{(k)}(\overline{v})$. За величину такта примем промежуток времени, необходимый для выполнения самой длительной операции $F_\lambda^{(k)},\ 1 \le \lambda \le \le \Lambda,\ 1 \le k \le K$. Это накладывает следующие ограничения на таймирующие функции (выбор векторов $\tau^{(k)} \in \mathbf{Z}^{n+p},\ 1 \le k \le K$):

$$t^{(j)}(v + \varphi^{(i,j)}) > t^{(i)}(v), \quad v + \varphi^{(i,j)} \in V^{(j)}, \quad v \in V^{(i)}, \quad \varphi^{(i,j)} \in \Phi, \tag{4.4}$$

в частности, при $i = j$

$$\tau^{(i)} \cdot \varphi^{(i,i)} > 0, \quad \varphi^{(i,i)} \in \Phi', \quad 1 \le i \le K. \tag{4.5}$$

Далее будем считать, что векторы $\tau^{(k)},\ 1 \le k \le K$, имеют взаимно простые координаты, а множество значений, принимаемых таймирующими функциями, заполняет целочисленный промежуток $[m, M]$ без пропусков, где $m = \min\limits_{1 \le k \le K} (\min\limits_{v \in V^{(k)}} t^{(k)}(v)),\ M = \max\limits_{1 \le k \le K} (\max\limits_{v \in V^{(k)}} t^{(k)}(v))$. Тогда при выполнении условий (4.4), (4.5) таймирующие функции (4.3) распараллеливают все вершины графа $G = (V, E)$ алгоритма (4.1), распределяя их по ярусам параллельной формы. Значение таймирующей функции $t^{(k)}(v)$ означает номер яруса параллельной формы, которому принадлежит вершина $v \in V^{(k)}$. Высота параллельной формы определяется как

$$T = \max_{1 \le k \le K} (\max_{v \in V^{(k)}} t^{(k)}(v)) - \min_{1 \le k \le K} (\min_{v \in V^{(k)}} t^{(k)}(v)) + 1. \tag{4.6}$$

Заметим, что таймирующие функции $t^{(k)}(v) = \tau^{(k)} \cdot v = \overline{\tau}^{(k)} \cdot \overline{v} + \widehat{\tau}^{(k)} \cdot \widehat{v}^{(k)},\ v \in V^{(k)},\ \overline{\tau}^{(k)} = (\tau_1^{(k)}, \ldots, \tau_n^{(k)}) \in \mathbf{Z}^n,\ \widehat{\tau}^{(k)} = (\tau_{n+1}^{(k)}, \ldots, \tau_{n+p}^{(k)}) \in \mathbf{Z}^p$,

определены на непересекающихся множествах $V^{(k)},\ 1 \le k \le K$. Однако их значения совпадают со значениями таймирующих функций

$$\overline{t}^{(k)}(\overline{v}) = \overline{\tau}^{(k)} \cdot \overline{v} + t_0^{(k)}, \quad \overline{v} \in \overline{V}, \ \ t_0^{(k)} = \widehat{\tau}^{(k)} \cdot \widehat{v}^{(k)}, \tag{4.7}$$

определенных на одном и том же индексном множестве $\overline{V}$ для всех $1 \le k \le K$:

$$t^{(k)}(v) = \overline{t}^{(k)}(\overline{v}), \quad v = (\overline{v}, \widehat{v}^{(k)}) \in V^{(k)}, \ \ \overline{v} \in \overline{V}. \tag{4.8}$$

Ограничения (4.4) примут вид

$$\begin{gathered} \overline{\tau}^{(j)} \cdot \overline{\varphi}^{(i,j)} + (\overline{\tau}^{(j)} - \overline{\tau}^{(i)}) \cdot \overline{v} + t_0^{(j)} - t_0^{(i)} > 0, \\ \overline{\varphi}^{(i,j)} \in \overline{\Phi}, \ \ \overline{v} \in \overline{V}^{(i,j)}, \end{gathered} \tag{4.9}$$

где $\overline{V}^{(i,j)} = \{\overline{v} \in \overline{V} \mid \overline{v} + \overline{\varphi}^{(i,j)} \in \overline{V}\}$.

Приведем условия, при выполнении которых высота параллельной формы алгоритма (4.1) при раздельном таймировании меньше минимальной высоты параллельной формы алгоритма (4.1), которая может быть достигнута при едином таймировании, когда $\overline{V}$ является n-мерным параллелепипедом: $\overline{V} = \{(i_1, \ldots, i_n) \in \mathbf{Z}^n : \quad \alpha_k \le i_k \le \beta_k, \ 1 \le k \le n\}$. В этом случае $\overline{V}^{(i,j)} = \{(i_1, \ldots, i_n) \in \mathbf{Z}^n \mid \alpha_k^{(i,j)} \le i_k \le \beta_k^{(i,j)}, \ 1 \le k \le n\}$, где $\alpha_k^{(i,j)} = \max(\alpha_k, \alpha_k - \overline{\varphi}_k^{(i,j)}), \quad \beta_k^{(i,j)} = \min(\beta_k, \beta_k - \overline{\varphi}_k^{(i,j)})$.

Введем в рассмотрение множества

$$I_j = \{i \in \mathbf{N} \mid \overline{\varphi}^{(i,j)} \in \overline{\Phi}\}, \ I = \bigcup_{j=1}^{K} I_j;$$

$$J_i = \{j \in \mathbf{N} \mid \overline{\varphi}^{(i,j)} \in \overline{\Phi}\}, \ J = \bigcup_{i=1}^{K} J_i = \{1, \ldots, K\}.$$

Обозначим через $\overline{\tau}^{\min} = (\overline{\tau}_1^{\min}, \ldots, \overline{\tau}_n^{\min}), \ \overline{\tau}^{\min} \cdot \overline{\varphi}^{(i,j)} \ge 1, \ \overline{\varphi}^{(i,j)} \in \overline{\Phi}$, вектор таймирования $t(\overline{v}) = \overline{\tau}^{\min} \cdot \overline{v} + t_0^{\min}, \ \overline{v} \in \overline{V}$, определяющий параллельную форму алгоритма (4.1) минимальной высоты $|T_0| = \sum\limits_{k=1}^{n} |\overline{\tau}_k^{\min}|(\beta_k - \alpha_k) + 1$ при едином таймировании.

Теорема 4.1. *Если при каждом $j \in J$ выполняются неравенства*

$$1 + \max_{i \in I_j}\Big\{ t_0^{(i)} + \frac{1}{2}\sum_{k=1}^{n}(|\overline{\tau}_k^{(j)} - \overline{\tau}_k^{(i)}|(\beta_k^{(i,j)} - \alpha_k^{(i,j)}) - \\ -(\overline{\tau}_k^{(j)} - \overline{\tau}_k^{(i)})(\beta_k^{(i,j)} + \alpha_k^{(i,j)})) - \overline{\tau}^{(j)} \cdot \overline{\varphi}^{(i,j)}\Big\} \le t_0^{(j)} < \\ < \min_{i \in I}\Big\{ t_0^{(i)} + \frac{1}{2}\sum_{k=1}^{n}((2|\overline{\tau}_k^{\min}| - |\overline{\tau}_k^{(j)}| - |\overline{\tau}_k^{(i)}|)(\beta_k - \alpha_k) - \\ -(\overline{\tau}_k^{(j)} - \overline{\tau}_k^{(i)})(\beta_k + \alpha_k))\Big\}, \tag{4.10}$$

то высота $|T|$ параллельной формы алгоритма при раздельном таймировании операций алгоритма (4.1) удовлетворяет неравенству $|T| < |T_0|$.

Доказательство. Ограничения (4.9) будут выполняться, если выполняются неравенства $\overline{\tau}^{(j)} \cdot \overline{\varphi}^{(i,j)} + \min\limits_{\overline{v} \in \overline{V}^{(i,j)}}(\overline{\tau}^{(j)} - \overline{\tau}^{(i)})\overline{v} + t_0^{(j)} - t_0^{(i)} \ge 1$ или, так как $\overline{V}^{(i,j)}$ — n-мерный параллелепипед, неравенства $\overline{\tau}^{(j)} \cdot \overline{\varphi}^{(i,j)} - 2^{-1}\sum\limits_{k=1}^{n}(|\overline{\tau}_k^{(j)} - \overline{\tau}_k^{(i)}|(\beta_k^{(i,j)} - \alpha_k^{(i,j)}) - (\overline{\tau}_k^{(j)} - \overline{\tau}_k^{(i)})(\beta_k^{(i,j)} + \alpha_k^{(i,j)})) + t_0^{(j)} - t_0^{(i)} \ge 1$, $\overline{\varphi}^{(i,j)} \in \overline{\Phi}$. Поскольку при фиксированном $j \in J$ эти неравенства должны выполняться для всех $i \in I_j$, то получим ограничение $t_0^{(j)} \ge 1 + \max\limits_{i \in I_j}\{t_0^{(i)} + 2^{-1}\sum\limits_{k=1}^{n}(|\overline{\tau}_k^{(j)} - \overline{\tau}_k^{(i)}|(\beta_k^{(i,j)} - \alpha_k^{(i,j)}) - (\overline{\tau}_k^{(j)} - \overline{\tau}_k^{(i)})(\beta_k^{(i,j)} + \alpha_k^{(i,j)})) - \overline{\tau}^{(j)} \cdot \overline{\varphi}^{(i,j)}\}$, что соответствует левой части неравенства (4.10).

По определению высоты параллельной формы (4.6) с учетом равенства (4.8) получим

$$|T| = \max_{j \in J}(\max_{\overline{v} \in \overline{V}} \overline{\tau}^{(j)} \cdot \overline{v} + t_0^{(j)}) - \min_{i \in I}(\min_{\overline{v} \in \overline{V}} \overline{\tau}^{(i)} \cdot \overline{v} + t_0^{(i)}) + 1 = \\ = \max_{i \in I,\ j \in J}(\max_{\overline{v} \in \overline{V}} \overline{\tau}^{(j)} \cdot \overline{v} - \min_{\overline{v} \in \overline{V}} \overline{\tau}^{(i)} \cdot \overline{v} + t_0^{(j)} - t_0^{(i)}) + 1 = \max_{i \in I,\ j \in J}\{2^{-1}\sum_{k=1}^{n}((|\overline{\tau}_k^{(j)}| + \\ + |\overline{\tau}_k^{(i)}|)(\beta_k - \alpha_k) + (\overline{\tau}_k^{(j)}\,\overline{\tau}_k^{(i)})(\beta_k + \alpha_k)) + t_0^{(j)} - t_0^{(i)}\} + 1 = \sum_{k=1}^{n}|\overline{\tau}_k^{\min}|(\beta_k - \\ - \alpha_k) + 1 + \max_{i \in I,\ j \in J}\{2^{-1}\sum_{k=1}^{n}((|\overline{\tau}_k^{(j)}| + |\overline{\tau}_k^{(i)}| - 2|\overline{\tau}_k^{\min}|)(\beta_k - \alpha_k) + (\overline{\tau}_k^{(j)} - \\ - \overline{\tau}_k^{(i)})(\beta_k + \alpha_k)) + t_0^{(j)} - t_0^{(i)}\} = |T_0| - \min_{i \in I,\ j \in J}\{2^{-1}\sum_{k=1}^{n}((2|\overline{\tau}_k^{\min}| - |\overline{\tau}_k^{(j)}| - \\ - |\overline{\tau}_k^{(i)}|)(\beta_k - \alpha_k) - (\overline{\tau}_k^{(j)} - \overline{\tau}_k^{(i)})(\beta_k + \alpha_k)) + t_0^{(i)} - t_0^{(j)}\}.$$

По условию (4.10)

$$\min_{j\in J}\{\min_{i\in I}\{t_0^{(i)} + 2^{-1}\sum_{k=1}^{n}((2|\overline{\tau}_k^{\min}| - |\overline{\tau}_k^{(j)}| - |\overline{\tau}_k^{(i)}|)(\beta_k - \alpha_k) - \\ -(\overline{\tau}_k^{(j)} - \overline{\tau}_k^{(i)})(\beta_k + \alpha_k))\} - t_0^{(j)}\} > 0,$$

откуда следует утверждение теоремы.

Пример 4.3. Рассмотрим алгоритм перемножения двух квадратных матриц размером $N \times N$, заданный следующей системой рекуррентных уравнений вида (3.1):

$$x^{(1)}(i_1, i_2, i_3) = x^{(1)}(i_1 - 1, i_2, i_3),\ 1 \le i_1, i_2, i_3 \le N;$$
$$x^{(2)}(i_1, i_2, i_3) = x^{(2)}(i_1, i_2 - 1, i_3),\ 1 \le i_1, i_2, i_3 \le N;$$
$$x^{(3)}(i_1, i_2, i_3) = \begin{cases} x^{(1)}(i_1 - 1, i_2, i_3) \cdot x^{(2)}(i_1, i_2 - 1, i_3), \\ \qquad 1 \le i_1, i_2 \le N,\ i_3 = 1; \\ x^{(1)}(i_1 - 1, i_2, i_3) \cdot x^{(2)}(i_1, i_2 - 1, i_3) + \\ + x^{(3)}(i_1, i_2, i_3 - 1),\ 1 \le i_1, i_2 \le N,\ 2 \le i_3 \le N. \end{cases}$$

В данном случае $n = 3,\ K = 3,\ \Lambda = 2$;
$\overline{V} = \{\overline{v} = (i_1, i_2, i_3) \in \mathbf{Z}^3 \mid 1 \le i_1, i_2, i_3 \le N\}$;
$\overline{V}_1 = \{\overline{v} = (i_1, i_2, i_3) \in \mathbf{Z}^3 \mid 1 \le i_1, i_2 \le N,\ i_3 = 1\}$;
$\overline{V}_2 = \{\overline{v} = (i_1, i_2, i_3) \in \mathbf{Z}^3 \mid 1 \le i_1, i_2 \le N,\ 2 \le i_3 \le N\}$;
$\overline{\Phi} = \{\overline{\varphi}^{(1,1)} = \overline{\varphi}^{(1,3)} = (1, 0, 0),\ \overline{\varphi}^{(2,2)} = \overline{\varphi}^{(2,3)} = (0, 1, 0),\ \overline{\varphi}^{(3,3)} = (0, 0, 1)\}$.

Перейдем к мелкозернистой модели, полагая $p = \lceil \log_2 3 \rceil = 2,\ \widehat{v}^{(1)} = (0, 1),\ \widehat{v}^{(2)} = (1, 0),\ \widehat{v}^{(3)} = (1, 1)$. Получим

$$V = \{v = (i_1, i_2, i_3, i_4, i_5) \in \mathbf{Z}^5 \mid \\ 1 \le i_1, i_2, i_3 \le N,\ 0 \le i_4, i_5 \le 1,\ i_4 + i_5 \ne 0\};$$
$$V^{(1)} = \{v = (i_1, i_2, i_3, i_4, i_5) \in \mathbf{Z}^5 \mid 1 \le i_1, i_2, i_3 \le N,\ i_4 = 0,\ i_5 = 1\};$$
$$V^{(2)} = \{v = (i_1, i_2, i_3, i_4, i_5) \in \mathbf{Z}^5 \mid 1 \le i_1, i_2, i_3 \le N,\ i_4 = 1,\ i_5 = 0\};$$
$$V^{(3)} = \{v = (i_1, i_2, i_3, i_4, i_5) \in \mathbf{Z}^5 \mid 1 \le i_1, i_2, i_3 \le N,\ i_4 = i_5 = 1\};$$
$$V_1^{(3)} = \{v = (i_1, i_2, i_3, i_4, i_5) \in \mathbf{Z}^5 \mid 1 \le i_1, i_2 \le N,\ i_3 = i_4 = i_5 = 1\};$$
$$V_2^{(3)} = \{v = (i_1, i_2, i_3, i_4, i_5) \in \mathbf{Z}^5 \mid 1 \le i_1, i_2 \le N, \\ 2 \le i_3 \le N,\ i_4 = i_5 = 1\};$$

$$\Phi' = \{\varphi^{(1,1)} = (1,0,0,0,0) = e_1,\ \varphi^{(2,2)} = (0,1,0,0,0) = e_2,$$
$$\varphi^{(3,3)} = (0,0,1,0,0) = e_3\},;$$
$$\Phi'' = \{\varphi^{(1,3)} = (1,0,0,1,0) = e_1 + e_4,\ \varphi^{(2,3)} = (0,1,0,0,1) = e_2 + e_5\};$$
$$\Phi = \Phi' \cup \Phi'' = \{e_1,\ e_2,\ e_3,\ e_1 + e_4,\ e_2 + e_5\}.$$

Для рассматриваемого алгоритма единая таймирующая функция, которая определяет параллельную форму алгоритма минимальной высоты $|T_0| = 3N - 2$, имеет вид $t(\overline{v}) = i_1 + i_2 + i_3 - 2,\ \overline{v}(i_1, i_2, i_3) \in \overline{V}$, т. е. $\overline{\tau}^{\min} = (1,1,1)$.

Для раздельного таймирования поставим в соответствие переменным алгоритма таймирующие функции вида (4.3): $t^{(1)}(v) = \tau^{(1)} \cdot v,\ v \in$ $\in V^{(1)},\ t^{(2)}(v) = \tau^{(2)} \cdot v,\ v \in V^{(2)},\ t^{(3)}(v) = \tau^{(3)} \cdot v,\ v \in V^{(3)};\ \tau^{(1)} =$ $= (1,0,1,0,-1),\ \tau^{(2)} = (0,1,1,-1,0),\ \tau^{(3)} = (0,0,1,N-1,0)$. Их значения совпадают со значениями таймирующих функций (4.7): $\overline{t}^{(1)}(\overline{v}) =$ $= i_1 + i_3 - 1,\ \overline{t}^{(2)}(\overline{v}) = i_2 + i_3 - 1,\ \overline{t}^{(3)}(\overline{v}) = i_3 + N - 1,\ 1 \le i_1, i_2, i_3 \le$ $\le N$, т. е. $\overline{\tau}^{(1)} = (1,0,1),\ t_0^{(1)} = -1,\ \overline{\tau}^{(2)} = (0,1,1),\ t_0^{(2)} = -1,\ \overline{\tau}^{(3)} =$ $= (0,0,1),\ t_0^{(3)} = N - 1$, для которых выполняются условия (4.10).

Таким образом, согласно теореме 4.1, соответствующая высота параллельной формы алгоритма будет меньше минимальной высоты параллельной формы алгоритма при едином таймировании, т. е. $|T| <$ $< |T_0| = 3N - 2$. Высота параллельной формы алгоритма при таком выборе таймирующих функций равна $|T| = 2N - 1$.

4.3. БАЗОВЫЙ ВЫЧИСЛИТЕЛЬНЫЙ ГРАФ

Каждая вершина $v = (\overline{v}, \widehat{v}^{(k)}) \in V$ графа $G = (V, E)$ алгоритма (4.1), размещенного в пространстве $\mathbf{Z}^{n+p}$, помечена операцией вычисления значения переменной $x^{(k)}$ в точке $\overline{v} \in \overline{V}$ и значением таймирующей функции $t^{(k)}(v) = \overline{t}^{(k)}(\overline{v})$, которое можно трактовать как номер такта, на котором по входящим в вершину дугам, соответствующим направлениям $\varphi^{(i,k)} \in \Phi,\ i \in I_k$, поступают значения переменных алгоритма (4.1) $x^{(i)}(\overline{v} - \overline{\varphi}^{(i,k)}),\ i \in I_k$, и осуществляется вычисление $x^{(k)}(\overline{v})$.

В связи с тем, что при раздельном таймировании каждой переменной алгоритма соответствует своя таймирующая функция, число тактов, на которые необходимо задержать вычисленное значение $x^{(k)}(v)$

в вершине v прежде, чем передать его по выходящим из вершины v дугам для дальнейшего использования, не является постоянным, а зависит от координат вершины v. *Задержкой в вершине* $v \in V^{(k)}$ *по выходящей дуге* $(v, v+\varphi^{(k,j)}) \in E,\ v+\varphi^{(k,j)} \in V^{(j)}$, определяемой вектором $\varphi^{(k,j)} \in \Phi,\ 1 \le k \le K,\ j \in J_k$, будем называть целое положительное число

$$h_{\varphi^{(k,j)}}(v) = t^{(j)}(v+\varphi^{(k,j)}) - t^{(k)}(v), \quad v \in V^{(k,j)}, \tag{4.11}$$

где $V^{(k,j)} = \{v \in V \mid v \in V^{(k)},\ v+\varphi^{(k,j)} \in V^{(j)}\},\ (v, v+\varphi^{(k,j)}) \in E$, $\varphi^{(k,j)} \in \Phi$.

Заметим, что для $k = j$ задержка в вершине $v \in V^{(k)}$ по выходящей дуге, определяемой вектором $\varphi^{(k,k)} \in \Phi$, не зависит от точки $v \in V^{(k)}$ и определяется как $h_{\varphi^{(k,k)}}(v) = \tau^{(k)} \cdot \varphi^{(k,k)}$.

Задержку в граничной по направлению вектора $\varphi^{(k,j)} \in \Phi$ вершине $v \in V^{(k)}$ по выходящей дуге, определяемой вектором $\varphi^{(k,j)} \in \Phi$, так же, как и при едином таймировании, будем полагать равной единице, т. е. $h_{\varphi^{(k,j)}}(v) = 1,\ v \in V^{(k)} \backslash V^{(k,j)}$, если не оговорено противное.

Поскольку в мелкозернистой модели алгоритма каждая переменная $x^{(k)}$ вычисляется в соответствующей ей области $V^{(k)}$, пометим каждую вершину $v \in V^{(k)}$ *вектором задержек* $H^{(k)}(v) = (h_{\varphi^{(k,1)}}(v), \ldots, h_{\varphi^{(k,K)}}(v))$, причем в отличие от случая единого таймирования вектор задержек зависит от координат вершины $v \in V^{(k)}$. Вектор задержек является частично неопределенным. Если не существует дуги, выходящей из вершины $v \in V^{(k)}$, характеризующейся вектором $\varphi^{(k,j_0)}$, т. е. $j_0 \notin J_k$, то в векторе $H^{(k)}$ на месте координаты j_0 будем ставить прочерк. Множество векторов задержек, соответствующих всем вершинам $v \in V^{(k)}$, обозначим $H^{(k)} = \{H^{(k)}(v),\ v \in V^{(k)}\}$.

Множество значений таймирующей функции $t^{(k)}(v)$, соответствующих всем вершинам $v \in V^{(k)}$, обозначим $t^{(k)} = \{t^{(k)}(v),\ v \in V^{(k)}\}$.

Таким образом, получаем базовый вычислительный граф алгоритма (4.1), $G = (V, E, T, H)$, размещенный в пространстве $\mathbf{Z}^{n+p}$, где V, E — множество вершин и множество дуг графа алгоритма (4.1), вершины графа помечены выполняемыми операциями алгоритма, значениями таймирующих функций из множества $T = \{t^{(k)},\ 1 \le k \le K\}$ и векторами задержек из множества $H = \{H^{(k)},\ 1 \le k \le K\}$.

Будем рассматривать базовый вычислительный граф как граф семейства специализированных виртуальных вычислительных систем, реализующих алгоритм (4.1). Опишем структуру такой системы и функционирование ее памяти.

Каждый ПЭ(v), $v = (\overline{v}, \widehat{v}^{(k)}) \in V_{\lambda}^{(k)}$, $1 \le \lambda \le \Lambda$, $1 \le k \le K$, имеет набор входов $\{I_{\varphi^{(i,k)}}^{(i,k)},\ i \in I_k\}$ и набор выходов $\{O_{\varphi^{(k,j)}}^{(k,j)},\ j \in J_k\}$. Число входов равно числу операндов операции $F_{\lambda}^{(k)}$ по вычислению $x^{(k)}(\overline{v})$. Число выходов равно числу операций, использующих результат операции $F_{\lambda}^{(k)}$ в качестве операнда. Каждый ПЭ(v) БВГ выполняет только одно вычисление $x^{(k)}(\overline{v})$ на такте $t^{(k)}(v)$. Для этого в ПЭ(v) имеется соответствующее устройство, позволяющее выполнить эту операцию над входными данными, поступившими в ПЭ(v) БВГ. Каждый ПЭ(v), $v \in V^{(k)}$, БВГ имеет один блок линейно связанных ячеек локальной памяти длины $\max\limits_{j \in J_k} h_{\varphi^{(k,j)}}(v)$: $\{R_{\alpha}^{(k)}(t),\quad 1 \le \alpha \le$ $\le \max\limits_{j \in J_k}\{h_{\varphi^{(k,j)}}(v)\},\ v \in V^{(k)}\}$, меняющих свое содержимое в зависимости от номера такта t. В первую ячейку памяти в момент времени $t = t^{(k)}(v)$ записывается результат выполнения операции $F_{\lambda}^{(k)}$: $R_1^{(k)}(t) =$ $= F_{\lambda}^{(k)}(I_{\varphi^{(1,k)}}^{(1,k)}, \ldots, I_{\varphi^{(K,k)}}^{(K,k)})$. На каждом следующем такте происходит перезапись содержимого каждой ячейки памяти с номером $\alpha - 1$ в ячейку памяти с номером α, $2 \le \alpha \le \max\limits_{j \in J_k} h_{\varphi^{(k,j)}}(v)$: $R_{\alpha}^{(k)}(t + \alpha - 1) = R_{\alpha-1}^{(k)}(t +$ $+ \alpha - 2)$. На каждый выход $O_{\varphi^{(k,j)}}^{(k,j)}$, $j \in J_k$, в момент времени $t^{(k)}(v) +$ $+ h_{\varphi^{(k,j)}}(v)$ подается содержимое ячейки памяти с номером $h_{\varphi^{(k,j)}}(v)$: $O_{\varphi^{(k,j)}}^{(k,j)} = R_{h_{\varphi^{(k,j)}}(v)}^{(k)}(t^{(k)}(v) + h_{\varphi^{(k,j)}}(v) - 1)$.

Если $\max\limits_{j \in J_k} h_{\varphi^{(k,j)}}(v) = 1$, то содержимое единственной ячейки памяти $R_1^{(k)}(t)$ в момент времени $t + 1$ подается на все выходы $O_{\varphi^{(k,j)}}^{(k,j)}$, $j \in J_k$, $t = t^{(k)}(v)$.

Процессорные элементы стыкуются друг с другом по входным и выходным каналам таким образом, что выход $O_{\varphi^{(k,j)}}^{(k,j)}$ ПЭ(v), $v \in V^{(k)}$, совмещается со входом $I_{\varphi^{(k,j)}}^{(k,j)}$ ПЭ$(v + \varphi^{(k,j)})$, $v + \varphi^{(k,j)} \in V^{(j)}$, $\varphi^{(k,j)} \in \Phi$. Во времени такая стыковка означает следующее: если в момент времени $t = t^{(k)}(v) + h_{\varphi^{(k,j)}}(v)$ на выходе $O_{\varphi^{(k,j)}}^{(k,j)}$ ПЭ(v) появляется значение переменной $x^{(k)}(\overline{v})$, то в этот же момент времени t оно подается на вход $I_{\varphi^{(k,j)}}^{(k,j)}$ ПЭ$(v + \varphi^{(k,j)})$.

Пример 4.3 (продолжение). Вычислим задержки в вершинах графа алгоритма $G = (V, E)$ по выходящим дугам, определяемым векторами зависимостей $\varphi^{(i,j)} \in \Phi$:
$h_{\varphi(1,1)}(v) = \tau^{(1)} \cdot \varphi^{(1,1)} = 1,\ v \in V^{(1,1)} = \{v = (i_1, i_2, i_3, i_4, i_5) \in \mathbf{Z}^5 \mid 1 \leq$
$\leq i_1 \leq N-1,\ 1 \leq i_2, i_3 \leq N,\ i_4 = 0,\ i_5 = 1\}$;
$h_{\varphi(2,2)}(v) = \tau^{(2)} \cdot \varphi^{(2,2)} = 1,\ v \in V^{(2,2)} = \{(i_1, i_2, i_3, i_4, i_5) \in \mathbf{Z}^5 \mid 1 \leq i_2 \leq$
$\leq N-1,\ 1 \leq i_1, i_3 \leq N,\ i_4 = 1,\ i_5 = 0\}$;
$h_{\varphi(1,3)}(v) = t^{(3)}(v + \varphi^{(1,3)}) - t^{(1)}(v) = \overline{t}^{(3)}(\overline{v} + \overline{\varphi}^{(1,3)}) - \overline{t}^{(1)}(\overline{v}) = N -$
$- i_1,\ v \in V^{(1,3)} = V^{(1,1)}$;
$h_{\varphi(2,3)}(v) = t^{(3)}(v + \varphi^{(2,3)}) - t^{(2)}(v) = \overline{t}^{(3)}(\overline{v} + \overline{\varphi}^{(2,3)}) - \overline{t}^{(2)}(\overline{v}) = N -$
$- i_2,\ v \in V^{(2,3)} = V^{(2,2)}$;
$h_{\varphi(3,3)}(v) = \tau^{(3)} \cdot \varphi^{(3,3)} = 1,\ v \in V^{(3)}$.

Вершины $v \in V^{(1)} \backslash V^{(1,1)}$ являются граничными по направлениям векторов $\varphi^{(1,1)},\ \varphi^{(1,3)} \in \Phi$, следовательно, $h_{\varphi^{(1,1)}}(v) = h_{\varphi^{(1,3)}}(v) = 1,\ v \in V^{(1)} \backslash V^{(1,1)}$.

Вершины $v \in V^{(2)} \backslash V^{(2,2)}$ являются граничными по направлениям векторов $\varphi^{(2,2)},\ \varphi^{(2,3)} \in \Phi$, следовательно, $h_{\varphi^{(2,2)}}(v) = h_{\varphi^{(2,3)}}(v) = 1,\ v \in V^{(2)} \backslash V^{(2,2)}$.

Векторы задержек можно записать следующим образом:

$$H^{(1)}(v) = (1, -, N - i_1),\ v \in V^{(1,1)},$$

$$H^{(1)}(v) = (1, -, 1),\ v \in V^{(1)} \backslash V^{(1,1)};$$

$$H^{(2)}(v) = (-, 1, N - i_2),\ v \in V^{(2,2)},$$

$$H^{(2)}(v) = (-, 1, 1),\ v \in V^{(2)} \backslash V^{(2,2)};$$

$$H^{(3)}(v) = (-, -, 1),\ v \in V^{(3)}.$$

Множества векторов задержек:

$$H^{(1)} = \{(1, -, N - i_1) \in \mathbf{Z}^3,\ 1 \leq i_1 \leq N - 1\};$$

$$H^{(2)} = \{(-, 1, N - i_2) \in \mathbf{Z}^3,\ 1 \leq i_2 \leq N - 1\};$$

$$H^{(3)} = \{(-, -, 1) \in \mathbf{Z}^3\}.$$

$$t^{(1)} = \{i_1 + i_3 - 1 \in \mathbf{Z},\ 1 \leq i_1, i_3 \leq N\},$$

$$t^{(2)} = \{i_2 + i_3 - 1 \in \mathbf{Z},\ 1 \leq i_2, i_3 \leq N\},$$

$$t^{(3)} = \{i_3 + N - 1 \in \mathbf{Z},\ 1 \leq i_3 \leq N\}$$

Так как координаты векторов зависимостей являются взаимно-простыми числами, то множества значений таймирующих функций заполняют целочисленный промежуток $[1, 2N-1]$ без пропусков.

Таким образом, получили БВГ алгоритма $G = (V, E, T, H)$, где V, E — множество вершин и множество дуг графа алгоритма в пространстве $\mathbf{Z}^5$, $T = \{t^{(1)}, t^{(2)}, t^{(3)}\}$, $H = \{H^{(1)}, H^{(2)}, H^{(3)}\}$.

Опишем функционирование памяти процессорных элементов виртуальной вычислительной системы, соответствующей БВГ. Каждый процессорный элемент, размещенный в точке $v(i_1, i_2, i_3, 0, 1) \in V^{(1)}$, имеет блок ячеек локальной памяти $\{R_\alpha^{(1)}(t),\ 1 \le \alpha \le N - i_1\}$.

В момент времени $t = t^{(1)}(v) = i_1 + i_3 - 1$ происходит запись в первую ячейку памяти данного, поступившего в этот момент времени на вход $I_{e_1}^{(1,1)}$: $R_1^{(1)}(t) = I_{e_1}^{(1,1)}$. На каждом следующем такте происходит перезапись этого данного в следующую ячейку памяти: $R_\alpha^{(1)}(t + \alpha - 1) = R_{\alpha-1}^{(1)}(t + \alpha - 2),\ 2 \le \alpha \le N - i_1$.

В момент времени $t+1$ содержимое ячейки памяти $R_1^{(1)}(t)$ подается на выход $O_{e_1}^{(1,1)}$ процессорного элемента: $O_{e_1}^{(1,1)} = R_1^{(1)}(t)$.

В момент времени $t + N - i_1$ содержимое ячейки памяти $R_{N-i_1+1}^{(1)}(t + N - i_1)$ подается на выход $O_{e_1+e_4}^{(1,3)}$ процессорного элемента: $O_{e_1+e_4}^{(1,3)} = R_{N-i_1}^{(1)}(t + N - i_1 - 1)$.

Аналогично опишем функционирование памяти процессорных элементов, размещенных в точках $v \in V^{(2)}$ и $v \in V^{(3)}$:

$v(i_1, i_2, i_3, 1, 0) \in V^{(2)}$:
$R_1^{(2)}(t) = I_{e_2}^{(2,2)}$,
$R_\alpha^{(2)}(t + \alpha - 1) = R_{\alpha-1}^{(2)}(t + \alpha - 2),\ 2 \le \alpha \le N - i_2$,
$O_{e_2}^{(2,2)} = R_1^{(2)}(t),\ O_{e_2+e_5}^{(2,3)} = R_{N-i_2}^{(2)}(t + N - i_2 - 1)$,
$t = t^{(2)}(v) = i_2 + i_3 - 1$;

$v(i_1, i_2, 1, 1, 1) \in V_1^{(3)}$:
$R_1^{(3)}(t) = I_{e_1+e_4}^{(1,3)} \cdot I_{e_2+e_5}^{(2,3)},\ O_{e_3}^{(3,3)} = R_1^{(3)}(t)$,
$t = t^{(3)}(v) = 1 + N - 1$;

$v(i_1, i_2, i_3, 1, 1) \in V_2^{(3)}$:
$R_1^{(3)}(t) = I_{e_1+e_4}^{(1,3)} \cdot I_{e_2+e_5}^{(2,3)} + I_{e_3}^{(3,3)},\ O_{e_3}^{(3,3)} = R_1^{(3)}(t)$,
$t = t^{(3)}(v) = i_3 + N - 1$.

4.4. ОТОБРАЖЕНИЕ В ПРОСТРАНСТВА МЕНЬШИХ РАЗМЕРНОСТЕЙ

Пусть алгоритм (4.1) описан БВГ $G = (V, E, T, H)$, $V \in \mathbf{Z}^{n+p}$. С помощью линейной функции размещения $\pi\colon \mathbf{Z}^{n+p} \to \mathbf{Z}^r$ отобразим БВГ в «физически реализуемые» пространства, получая тем самым вычислительные графы в качестве архитектурных моделей спецпроцессоров: $\mathbf{Z}^{n+p} \xrightarrow{\pi} \mathbf{Z}^1$ (линейные архитектуры), $\mathbf{Z}^{n+p} \xrightarrow{\pi} \mathbf{Z}^2$ (плоские архитектуры), $\mathbf{Z}^{n+p} \xrightarrow{\pi} \mathbf{Z}^3$ (трехмерные архитектуры).

Для локальности соединений между процессорными элементами проектируемого спецпроцессора требуется, чтобы координаты образов векторов зависимостей не превосходили по абсолютной величине единицы:

$$\|\pi(\varphi^{(i,j)})\|_\infty \le 1, \quad \varphi^{(i,j)} \in \Phi. \tag{4.12}$$

Условие совместимости таймирующей функции $t^{(k)}$ и функции размещения π выглядит аналогично формуле (2.8):

$$\begin{gathered} t^{(k)}(v') \ne t^{(k)}(v''), \text{ если } \pi(v') = \pi(v''), \\ v', v'' \in V^{(k)},\ v' \ne v'',\ 1 \le k \le K. \end{gathered} \tag{4.13}$$

Критерий выполнения этого условия дает следующее утверждение: *таймирующие функции (4.3) совместимы с функцией размещения* π *тогда и только тогда, когда*

$$\tau^{(k)} \cdot s \ne 0, \quad s \in S^{(k)} = \{s \in V^{(k)} \ominus V^{(k)} \mid \pi(s) = 0,\ s \ne 0\}, \tag{4.14}$$

где $V^{(k)} \ominus V^{(k)} = \{s \in \mathbf{Z}^{n+p} \mid s = v' - v'',\ v', v'' \in V^{(k)}\}$.

Обозначения образов компонент БВГ при отображении π: $V^\pi = \{\widetilde{v} \in \mathbf{Z}^r \mid \widetilde{v} = \pi(v),\ v \in V\}$ – множество вершин ВГ, размещенных в пространстве $\mathbf{Z}^r$, в частности $V_\lambda^{(k)\pi} = \{\widetilde{v} \in \mathbf{Z}^r \mid \widetilde{v} = \pi(v),\ v \in V_\lambda^{(k)}\}$, $\bigcup\limits_{\lambda=1}^{\Lambda} V_\lambda^{(k)\pi} = V^{(k)\pi}$, $\bigcup\limits_{k=1}^{K} V^{(k)\pi} = V^\pi$; $E^\pi = \{\widetilde{v}_1, \widetilde{v}_2) \in \mathbf{Z}^r \times \mathbf{Z}^r \mid \widetilde{v}_1 = \pi(v_1),\ \widetilde{v}_2 = \pi(v_2),\ (v_1, v_2) \in E\}$ – множество дуг ВГ; $t^{(k)\pi}(\widetilde{v}) = \{t^{(k)}(v) \in \mathbf{Z}^1 \mid v \in V^{(k)},\ \widetilde{v} = \pi(v)\}$, $\widetilde{v} \in V^{(k)\pi}$, $1 \le k \le K$, – многозначные таймирующие функции. Значения $t = t^{(k)\pi}(\widetilde{v})$ определяют номера тактов, на которых ПЭ$(\widetilde{v})$, соответствующий вершине $\widetilde{v} \in V^\pi$ вычислительного графа, выполняет операции, приписанные прообразам этой вершины в БВГ.

Описание функционирования памяти ПЭ($\widetilde{v}$), $\widetilde{v} \in V^{\pi}$, можно получить аналогично тому, как его получали в п. . Если при выбранном операторе π в процессорный элемент ПЭ($\widetilde{v}$), $\widetilde{v} \in V^{\pi}$, отображаются вершины множеств $V_{\lambda}^{(k_\delta)}$, $1 \le \delta \le \Delta$, то для выполнения соответствующих операций по вычислению значений переменных $x^{(k_\delta)}$, их хранению и передаче результатов в соседние процессорные элементы в ПЭ($\widetilde{v}$) включаются все устройства, имеющиеся у прообразов, в частности Δ блоков локальной памяти $\{R_{\alpha}^{(k_\delta)}(t),\ 1 \le \alpha \le \max\limits_{j \in J_{k_\delta}} h_{\varphi^{(k_\delta,j)}}(v),\ v \in V^{(k_\delta)},\ \pi(v) = \widetilde{v}\}$, $1 \le \delta \le \Delta$, изменяющих свое содержание в зависимости от номера такта $t = t^{(k_\delta)\pi}(\widetilde{v})$, $\widetilde{v} = \pi(v) \in V_{\lambda}^{(k_\delta)\pi}$, $1 \le \delta \le \Delta$:

$$R_1^{(k_\delta)}(t) = F_{\lambda}^{(k_\delta)}(I_{\pi(\varphi^{(1,k_\delta)})}^{(1,k_\delta)}, \ldots, I_{\pi(\varphi^{(K,k_\delta)})}^{(K,k_\delta)}), \tag{4.15}$$

$$R_{\alpha}^{(k_\delta)}(t+\alpha-1) = R_{\alpha-1}^{(k_\delta)}(t+\alpha-2),\ 1 \le \alpha \le \max_{j \in J_{k_\delta}} h_{\varphi^{(k_\delta,j)}}(v);$$

$$O_{\pi(\varphi^{(k_\delta,j)})}^{(k_\delta,j)} = R_{h_{\varphi^{(k_\delta,j)}}(v)}^{(k_\delta)}(t + h_{\varphi^{(k_\delta,j)}}(v) - 1),\ j \in J_{k_\delta}. \tag{4.16}$$

Если для некоторого $i_0 \in I_{k_\delta}$ вектор $\varphi^{(i_0,k_\delta)}$ принадлежит ядру оператора π, то в равенстве (4.15) следует заменить $I_{\pi(\varphi^{(i_0,k_\delta)})}^{(i_0,k_\delta)}$ на $R_{h_{\varphi^{(i_0,k_\delta)}}(v)}^{(i_0)}(t-1)$. Если же для некоторого $j_0 \in J_{k_\delta}$ вектор $\varphi^{(k_\delta,j_0)}$ принадлежит ядру оператора π, то соответствующее равенство (4.16) следует опустить.

4.5. УПРАВЛЕНИЕ

При выбранном операторе отображения π в одну и ту же вершину $\widetilde{v} = \pi(v) \in V^{\pi}$ вычислительного графа могут отображаться вершины БВГ разных типов, т. е. в ПЭ($\widetilde{v}$) вычисление значений переменных может проводиться по разным правилам, а вывод результатов вычислений может проводиться с разными задержками. При проектировании спецпроцессора, реализующего алгоритм (4.1), необходимо осуществить разметку вершин разных типов соответствующими управляющими метками.

Разобьем область вычислений переменной $x^{(k)}$, $V^{(k)} = \bigcup\limits_{\lambda=1}^{\Lambda} V_{\lambda}^{(k)}$, на подмножества с одинаковыми задержками по выходящим дугам. Со-

ответственно разобьем каждое множество $H^{(k)}$, $1 \le k \le K$, на непересекающиеся подмножества $H_\gamma^{(k)}$, $1 \le \gamma \le \Gamma_k$, в каждом из которых содержатся равные между собой векторы задержек. Обозначим множество вершин $v \in V_\lambda^{(k)}$, которым соответствуют векторы задержек из множества $H_\gamma^{(k)}$, через $V_{\lambda\gamma}^{(k)}$, $1 \le \lambda \le \Lambda$, $1 \le \gamma \le$ $\le \Gamma_k$, $1 \le k \le K$. Будем называть вершины $v \in V_{\lambda\gamma}^{(k)}$ вершинами $k\lambda\gamma$ типа.

Обозначим $V_{\lambda\gamma}^{(k)\pi} = \{\widetilde{v} \in \mathbf{Z}^r \mid \widetilde{v} = \pi(v),\ v \in V_{\lambda\gamma}^{(k)}\}$, $\bigcup\limits_{\gamma=1}^{\Gamma_k} V_{\lambda\gamma}^{(k)\pi} =$ $= V_\lambda^{(k)\pi}$. Таким образом, для введения управляющих меток необходимо выполнить хотя бы одно из условий:

$$V_{\lambda_1\gamma}^{(k)\pi} \cap V_{\lambda_2\gamma}^{(k)\pi} \ne \emptyset, \quad \lambda_1 \ne \lambda_2, \quad 1 \le \lambda_1, \lambda_2 \le \Lambda; \tag{4.17}$$

$$V_{\lambda\gamma_1}^{(k)\pi} \cap V_{\lambda\gamma_2}^{(k)\pi} \ne \emptyset, \quad \gamma_1 \ne \gamma_2, \quad 1 \le \gamma_1, \gamma_2 \le \Gamma_k. \tag{4.18}$$

Обозначим множество индексов k, для которых выполняется хотя бы одно из условий (4.17), (4.18), через K_m. Каждая вершина $v \in V_{\lambda\gamma}^{(k)}$, $k \in K_m$, $1 \le \lambda \le \Lambda$, $1 \le \gamma \le \Gamma_k$, должна быть помечена своей управляющей меткой $m_{\lambda\gamma}^{(k)}$. Пусть граница $\partial V_{\lambda\gamma}^{(k)}$ области $V_{\lambda\gamma}^{(k)}$ отображается на границу $\pi(\partial V)$ области $\pi(V)$: $\pi(\partial V_{\lambda\gamma}^{(k)}) \subset \pi(\partial V)$. Тогда во многих случаях разметку вершин $V_{\lambda\gamma}^{(k)}$ можно осуществить, используя алгоритм транспортировки. Для этого алгоритм (4.1) дополняется управляющими переменными $m^{(k)}$, $k \in K_m$, которые осуществляют транспортировку метки $m_{\lambda\gamma}^{(k)}$ от границы $\partial V_{\lambda\gamma}^{(k)}$ ко всем внутренним точкам области $V_{\lambda\gamma}^{(k)}$ по магистралям, определяемым вектором зависимости $\varphi^{(mk)}$:

$$\begin{gathered} m^{(k)}(v) = m^{(k)}(v - \varphi^{(mk)}), \quad v \in V^{(k)}; \\ m^{(k)}(v - \varphi^{(mk)}) = m_{\lambda\gamma}^{(k)}, \quad v \in \partial V_{\lambda\gamma}^{(k)}, \quad v - \varphi^{(mk)} \notin V_{\lambda\gamma}^{(k)}. \end{gathered} \tag{4.19}$$

Для транспортировки управляющих меток следует брать векторы $\varphi^{(mk)}$, не попадающие в ядро оператора отображения π: $\pi(\varphi^{(mk)}) \ne 0$.

Рассмотрим влияние управления на временные характеристики вычислительной системы. Поставим в соответствие переменным $m^{(k)}$ функции таймирования $t^{(mk)}\colon \mathbf{Z}^{n+p} \to \mathbf{Z}$ вида

$$t^{(mk)}(v) = \tau^{(mk)} \cdot v, \quad v \in V^{(k)}, \quad \tau^{(mk)} \in \mathbf{Z}^{n+p}. \tag{4.20}$$

Значения таймирующих функций вида (4.20) совпадают со значениями таймирующих функций $\overline{t}^{(mk)}(\overline{v}) = \overline{\tau}^{(mk)} \cdot \overline{v} + t_0^{(mk)}$, $\overline{v} \in \overline{V}$, $\overline{\tau}^{(mk)} \in \mathbf{Z}^n$, $t_0^{(mk)} \in \mathbf{Z}$, где $\overline{\tau}^{(mk)} = (\tau_1^{(mk)}, \ldots, \tau_n^{(mk)}) \in \mathbf{Z}^n$, $t_0^{(mk)} = \widehat{\tau}^{(mk)} \cdot \widehat{v}^{(k)}$, $\widehat{\tau}^{(mk)} = (\tau_{n+1}^{(mk)}, \ldots, \tau_{n+p}^{(mk)}) \in \mathbf{Z}^p$.

Поскольку векторы $\varphi^{(mk)}$ определяют дуги между вершинами $v \in V^{(k)}$ и $v + \varphi^{(mk)} \in V^{(k)}$, то существуют задержки в вершинах $v \in V^{(k)}$, $k \in K_m$, которые вычисляются следующим образом: $h_{\varphi^{(mk)}}(v) = \tau^{(mk)} \cdot \varphi^{(mk)}$, $v \in V^{(mk)}$, $k \in K_m$, где $V^{(mk)} = \{v \in V^{(k)} \mid v, v + \varphi^{(mk)} \in V^{(k)}\}$. Если вершина $v \in V^{(k)}$ является граничной по направлению вектора $\varphi^{(mk)}$, т. е. $v + \varphi^{(mk)} \notin V^{(k)}$, то $h_{\varphi^{(mk)}}(v) = 1$.

Так как вычисление переменной $x^{(k)}$ в вершине $v \in V^{(k)}$ зависит от значения управляющей переменной $m^{(k)}$ в этой вершине, то необходимо ввести понятие задержки управления в вершине $v \in V^{(k)}$ при вычислении переменной $x^{(k)}$.

Задержкой управления в вершине $v \in V^{(k)}$, $k \in K_m$, при вычислении переменной $x^{(k)}$ назовем целое положительное число

$$h_{mk}(v) = t^{(k)}(v) - t^{(mk)}(v), \ v \in V^{(k)}. \tag{4.21}$$

Необходимо, чтобы при отображении графа алгоритма оператором π вершины с одинаковым образом имели одинаковые задержки управления (в противном случае увеличивается число различных типов вершин области вычислений, что противоречит смыслу введения управляющих меток), т. е. задержки управления не должны изменяться при переходе от вершины графа алгоритма к вершине по направлению, определяемому вектором $\varphi \in \operatorname{Ker} \pi$. Другими словами, должно выполняться условие

$$\begin{gathered} h_{mk}(v - \varphi) = h_{mk}(v), \quad \varphi = (\varphi_1, \ldots, \varphi_n, 0, \ldots, 0) \in \mathbf{Z}^{n+p}, \\ \varphi \neq 0, \ \ \pi(\varphi) = 0, \ \ v, v - \varphi \in V^{(k)}, \ k \in K_m. \end{gathered} \tag{4.22}$$

Равенство задержек в условии (4.22) можно записать в другом виде, используя определения (4.20) и (4.21): $(\tau^{(k)} - \tau^{(mk)}) \cdot \varphi = 0$.

В итоге получим следующие ограничения для таймирующей функции, соответствующей управляющей переменной $m^{(k)}$:

$$t^{(k)}(v) > t^{(mk)}(v), \quad v \in V^{(k)}, \ \ k \in K_m; \tag{4.23}$$

$$\begin{gathered} (\tau^{(k)} - \tau^{(mk)}) \cdot \varphi = 0, \quad \pi(\varphi) = 0, \\ \varphi = (\varphi_1, \ldots, \varphi_n, 0, \ldots, 0) \in \mathbf{Z}^{n+p}, \ \ \varphi \neq 0, \ \ k \in K_m; \end{gathered} \tag{4.24}$$

$$\tau^{(mk)} \cdot \varphi^{(mk)} > 0, \quad k \in K_m. \tag{4.25}$$

Рассмотрим, как влияет введение управляющей переменной на высоту параллельной формы алгоритма $|T|$, которая определяется формулой (4.6). Очевидно, что $\max\limits_{1\le k\le K}(\max\limits_{v\in V^{(k)}} t^{(k)}(v))$ не увеличится в силу условия (4.23). Для того чтобы $\min\limits_{1\le k\le K}(\min\limits_{v\in V^{(k)}} t^{(k)}(v))$ не уменьшался, необходимо выбрать вектор $\widehat{\tau}^{(mk)}$ из условия (4.23) и неравенств $\min\limits_{v\in V^{(k)}} t^{(mk)}(v) \ge \min\limits_{v\in V^{(i)}} t^{(i)}(v),\ \varphi^{(i,k)} \in \Phi,\ k \in K_m$.

При поступлении в ПЭ($\pi(v)$) управляющей метки $m_{\lambda\gamma}^{(k)}$ этот процессорный элемент должен выполнить операцию $F_\lambda^{(k)}$ и подать на выход результат вычисления с задержками $h_{\varphi^{(k,j)}}(v)$ из множества $H_\gamma^{(k)}$.

Для вычисления переменной $x^{(k)}$ в ПЭ($\widetilde{v}$), $\widetilde{v} \in V_{\lambda\gamma}^{(k)\pi}$, $k \in K_m$, в момент времени $t = t^{(k)\pi}(\widetilde{v})$ необходим анализ значения содержимого ячейки памяти для управляющей метки в предыдущий момент времени $R_{h_{mk}(v)}^{(mk)}(t-1)$. Описание функционирования памяти ПЭ($\widetilde{v}$) дополняется следующими равенствами:

$$R_1^{(mk)}(t) = I_{\pi(\varphi^{(mk)})}^{(mk)},$$
$$R_\alpha^{(mk)}(t+\alpha-1) = R_{\alpha-1}^{(mk)}(t+\alpha-2),\ 2 \le \alpha \le \max(h_{mk}(v), h_{\varphi^{(mk)}}(v)),$$
$$O_{\pi(\varphi^{(mk)})}^{(mk)} = R_{h_{mk}(v)}^{(mk)}(t+h_{\varphi^{(mk)}}(v)-1),\ t = t^{(mk)\pi}(\widetilde{v}),$$

где $t^{(mk)\pi}(\widetilde{v}) = \{t^{(mk)}(v) \in \mathbf{Z} \mid v \in V^{(k)},\ \widetilde{v} = \pi(v)\}$, $\widetilde{v} \in V^{(k)\pi}$, $k \in$ $\in K_m$, – многозначная таймирующая функция.

ПЭ($\widetilde{v}$) ВГ дополняется устройством управления, которое анализирует значения управляющих меток, находящихся в ячейках $R_{h_{mk}(v)}^{(mk)}(t-$ $-1)$, $k \in K_m$, и определяет операции, которые необходимо выполнить в ПЭ($\widetilde{v}$) ВГ в данный момент времени в зависимости от этих значений.

Пример 4.3 (продолжение). Для получения спецпроцессора плоской архитектуры, реализующего алгоритм перемножения двух квадратных матриц, выберем функцию размещения $\pi\colon \mathbf{Z}^5 \to \mathbf{Z}^2$, которая задается матрицей $\Pi = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \end{pmatrix}$. Проектируемый спецпроцессор будет иметь только локальные связи, поскольку выполняются условия (4.12). Проверим выполнение условий (4.14) для выбранных ранее таймирующих функций $t^{(1)}$, $t^{(2)}$, $t^{(3)}$ и функции размещения π:

$V^{(1)} \ominus V^{(1)} = V^{(2)} \ominus V^{(2)} = V^{(3)} \ominus V^{(3)} = \{s = (i_1, i_2, i_3, i_4, i_5) \in \mathbf{Z}^5 \mid 1 - N \le i_1, i_2, i_3 \le N - 1,\ i_4 = i_5 = 0\}$, $S^{(1)} = S^{(2)} = S^{(3)} = \{s = (i_1, i_2, i_3, i_4, i_5) \in \mathbf{Z}^5 \mid 1 - N \le i_3 \le N - 1,\ i_3 \ne 0,\ i_1 = i_2 = i_4 = i_5 = 0\}$, $\tau^{(1)} \cdot s = i_3 \ne 0,\ s \in S^{(1)}$, $\tau^{(2)} \cdot s = i_3 \ne 0,\ s \in S^{(2)}$, $\tau^{(3)} \cdot s = i_3 \ne 0,\ s \in S^{(3)}$. Условие (4.14) выполняется, следовательно, таймирующие функции $t^{(1)},\ t^{(2)},\ t^{(3)}$ совместимы с функцией размещения π.

Образы областей вычисления переменных алгоритма одинаковы: $V^{\pi} = V^{(1)\pi} = V^{(2)\pi} = V^{(3)\pi} = V_1^{(3)\pi} = V_2^{(3)\pi} = \{\widetilde{v} = (i_1, i_2) \in \mathbf{Z}^2 \mid 1 \le i_1, i_2 \le N\}$; образы векторов зависимостей: $\pi(\varphi^{(1,1)}) = \pi(\varphi^{(1,3)}) = (1, 0),\ \pi(\varphi^{(2,2)}) = \pi(\varphi^{(2,3)}) = (0, 1),\ \pi(\varphi^{(3,3)}) = (0, 0)$.

Многозначные таймирующие функции для переменных алгоритма:

$$t^{(1)\pi}(\widetilde{v}) = \{i_1 + i_3 - 1,\ 1 \le i_3 \le N\},\ \widetilde{v} \in V^{(1)\pi};$$
$$t^{(2)\pi}(\widetilde{v}) = \{i_2 + i_3 - 1,\ 1 \le i_3 \le N\},\ \widetilde{v} \in V^{(2)\pi};$$
$$t^{(3)\pi}(\widetilde{v}) = \{i_3 + N,\ 1 \le i_3 \le N\},\ \widetilde{v} \in V^{(3)\pi}.$$

Проверим необходимость введения в БВГ алгоритма управляющих меток. Множество векторов задержек $H^{(1)}$ разобьем на N непересекающихся подмножеств $H_\gamma^{(1)},\ 1 \le \gamma \le N$. Векторы из подмножества $H_\gamma^{(1)},\ 1 \le \gamma \le N$, имеют вид $(1, -, N - \gamma + 1)$. Получим разбиение множества $V^{(1)}$ на подмножества: $V_\gamma^{(1)} = \{v = (i_1, i_2, i_3, i_4, i_5) \in \mathbf{Z}^5 \mid i_1 = \gamma,\ 1 \le i_2, i_3 \le N,\ i_4 = 0,\ i_5 = 1\},\ 1 \le \gamma \le N$, которые оператором π отображаются в следующие подмножества множества $V^{(1)\pi}$: $V_\gamma^{(1)\pi} = \{\widetilde{v} = (i_1, i_2) \in \mathbf{Z}^2 \mid i_1 = \gamma,\ 1 \le i_2 \le N\},\ 1 \le \gamma \le N$. Переменная $x^{(1)}$ в алгоритме вычисляется по одному правилу, поэтому рассматривать условие (4.17) не надо. Условие (4.18) не выполняется: $V_{\gamma_1}^{(1)\pi} \cap V_{\gamma_2}^{(1)\pi} = \emptyset,\ \gamma_1 \ne \gamma_2,\ 1 \le \gamma_1, \gamma_2 \le N$. Следовательно, для вычисления переменной $x^{(1)}$ в плоском вычислителе не надо вводить управление.

Аналогично можно проверить, что для вычисления переменной $x^{(2)}$ не надо вводить управление.

Проверим выполнение условий (4.17) и (4.18) для переменной $x^{(3)}$. Все векторы из множества $H^{(3)}$ одинаковые, поэтому условие (4.18) рассматривать не надо.

Условие (4.17) выполняется: $V_1^{(3)\pi} \cap V_2^{(3)\pi} = \{\widetilde{v} = (i_1, i_2) \in \mathbf{Z}^2 \mid 1 \le i_1, i_2 \le N\} \ne \emptyset$, т. е. для вычисления переменной $x^{(3)}$ в плоском вычислителе надо вводить управление. Пометим вершины $v \in V_1^{(3)}$ и $v \in V_2^{(3)}$ соответственно метками $m_1^{(3)}$ и $m_2^{(3)}$, для их транспортировки

введем управляющую переменную $m^{(3)}(v)$ и запишем алгоритм транспортировки вида (4.19):

$$m^{(3)}(v) = m^{(3)}(v - e_1),\ v \in V^{(3)};$$
$$m^{(3)}(v - e_1) = m_1^{(3)},\ v \in \partial V_1^{(3)} = \{v = (i_1, i_2, i_3, i_4, i_5) \in \mathbf{Z}^5 \mid$$
$$i_1 = i_3 = i_4 = i_5 = 1,\ 1 \le i_2 \le N\};$$
$$m^{(3)}(v - e_1) = m_2^{(3)},\ v \in \partial V_2^{(3)} = \{v = (i_1, i_2, i_3, i_4, i_5) \in \mathbf{Z}^5 \mid$$
$$i_1 = i_4 = i_5 = 1,\ 1 \le i_2 \le N,\ 2 \le i_3 \le N\}.$$

Вектор $\varphi^{(m3)} = e_1$ не попадает в ядро оператора отображения: $\pi(\varphi^{(m3)}) = (1, 0)$.

Если выбрать $t^{(m3)}(v) = \overline{t}^{(m3)}(\overline{v}) = i_1 + i_3 - 1,\ v(i_1, i_2, i_3, 1, 1) \in \in V^{(3)},\ \overline{v}(i_1, i_2, i_3) \in \overline{V}$, то легко проверить, что условия (4.23)–(4.25) выполняются, и такая таймирующая функция для переменной $m^{(3)}$ не увеличит высоты параллельной формы алгоритма, так как $1 = \min\limits_{v \in V^{(3)}} t^{(m3)}(v) = \min\limits_{v \in V^{(1)}} t^{(1)}(v) = 1$ и $\varphi^{(1,3)} \in \Phi$.

Вычислим задержки по дугам, определяемым вектором $\varphi^{(m3)}$: $h_{\varphi^{(m3)}}(v) = \tau^{(m3)} \cdot \varphi^{(m3)} = 1,\ v \in V^{(m3)} = \{v = (i_1, i_2, i_3, i_4, i_5) \in \mathbf{Z}^5 \mid 1 \le \le i_1 \le N-1,\ 1 \le i_2, i_3 \le N,\ i_4 = i_5 = 1\};\ h_{\varphi^{(m3)}}(v) = 1,\ v \in V^{(3)} \backslash V^{(m3)}$, поскольку вершины $v \in V^{(3)} \backslash V^{(m3)}$ являются граничными по направлению вектора $\varphi^{(m3)}$. Задержки управления в вершинах $v \in V^{(3)}$ при вычислении переменной $x^{(3)}$: $h_{m3}(v) = t^{(3)}(v) - t^{(m3)}(v) = \overline{t}^{(3)}(\overline{v}) - - \overline{t}^{(m3)}(\overline{v}) = N - i_1 + 1,\ v \in V^{(3)}$.

При отображении π получаем следующее: $\pi(\varphi^{(m3)}) = (1, 0)$, $t^{(m3)\pi}(\overline{v}) = \{i_1 + i_3 - 1,\ 1 \le i_3 \le N\},\ \overline{v} \in V^{(3)\pi}$.

Опишем функционирование памяти плоского вычислителя. Каждый ПЭ$(\widetilde{v})$, $\widetilde{v}(i_1, i_2) \in V^{\pi}$, имеет три блока ячеек локальной памяти $\{R_\alpha^{(m3)}(t),\ 1 \le \alpha \le N - i_1 + 1\},\ \{R_\alpha^{(1)}(t),\ 1 \le \alpha \le N - i_1 + + 1\},\ \{R_\alpha^{(2)}(t),\ 1 \le \alpha \le N - i_2 + 1\}$ для хранения значений переменных $m^{(3)},\ x^{(1)},\ x^{(2)}$ и одну ячейку памяти $R_1^{(3)}(t)$ для хранения значений переменной $x^{(3)}$. Содержимое этих ячеек меняется и подается на выходы следующим образом:

$R_1^{(m3)}(t) = I_{e_1}^{(m3)}$,
$R_\alpha^{(m3)}(t + \alpha - 1) = R_{\alpha-1}^{(m3)}(t + \alpha - 2),\ 2 \le \alpha \le N - i_1 + 1$,
$O_{e_1}^{(m3)} = R_1^{(m3)}(t)$,
$t = t^{(m3)\pi}(\widetilde{v}) = \{i_1 + i_3 - 1,\ 1 \le i_3 \le N\}$;

$$R_1^{(1)}(t) = I_{e_1}^{(1,1)},$$
$$R_\alpha^{(1)}(t+\alpha-1) = R_{\alpha-1}^{(1)}(t+\alpha-2),\ 2 \le \alpha \le N - i_1 + 1,$$
$$O_{e_1}^{(1,1)} = R_1^{(1)}(t),\ O_{e_1}^{(1,3)} = R_{N-i_1+1}^{(1)}(t+N-i_1),$$
$$t = t^{(1)\pi}(\widetilde{v}) = \{i_1 + i_3 - 1,\ 1 \le i_3 \le N\};$$
$$R_1^{(2)}(t) = I_{e_2}^{(2,2)},$$
$$R_\alpha^{(2)}(t+\alpha-1) = R_{\alpha-1}^{(2)}(t+\alpha-2),\ 2 \le \alpha \le N - i_2 + 1,$$
$$O_{e_2}^{(2,2)} = R_1^{(2)}(t),\ O_{e_2}^{(2,3)} = R_{N-i_2+1}^{(2)}(t+N-i_2),$$
$$t = t^{(2)\pi}(\widetilde{v}) = \{i_2 + i_3 - 1,\ 1 \le i_3 \le N\};$$
$$R_1^{(3)}(t) = \begin{cases} I_{e_1}^{(1,3)} \cdot I_{e_2}^{(2,3)}, \text{ если } R_{N-i_1+1}^{(m3)}(t-1) = m_1^{(3)}, \\ I_{e_1}^{(1,3)} \cdot I_{e_2}^{(2,3)} + R_1^{(3)}(t-1), \text{ если } R_{N-i_1+1}^{(m3)}(t-1) = m_2^{(3)}, \end{cases}$$
$$t = t^{(3)\pi}(\widetilde{v}) = \{i_3 + N,\ 1 \le i_3 \le N\}.$$

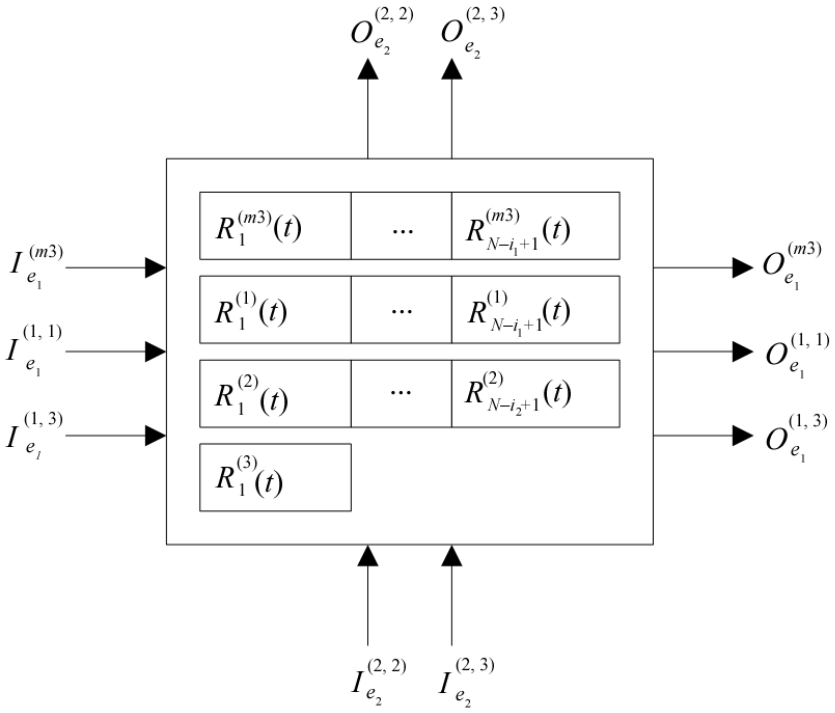

Рис. 4.3. Структура процессорного элемента ПЭ $(\widetilde{v})$, $\widetilde{v}(i_1, i_2) \in V^\pi$

Процессорный элемент имеет пять входов: $I_{e_1}^{(m3)}$, $I_{e_1}^{(1,1)}$, $I_{e_1}^{(1,3)}$, $I_{e_2}^{(2,2)}$, $I_{e_2}^{(2,3)}$ и пять выходов: $O_{e_1}^{(m3)}$, $O_{e_1}^{(1,1)}$, $O_{e_1}^{(1,3)}$, $O_{e_2}^{(2,2)}$, $O_{e_2}^{(2,3)}$.

Структура ПЭ полученной вычислительной системы изображена на рис. 4.3, а на рис. 4.4 – структура полученного процессора.

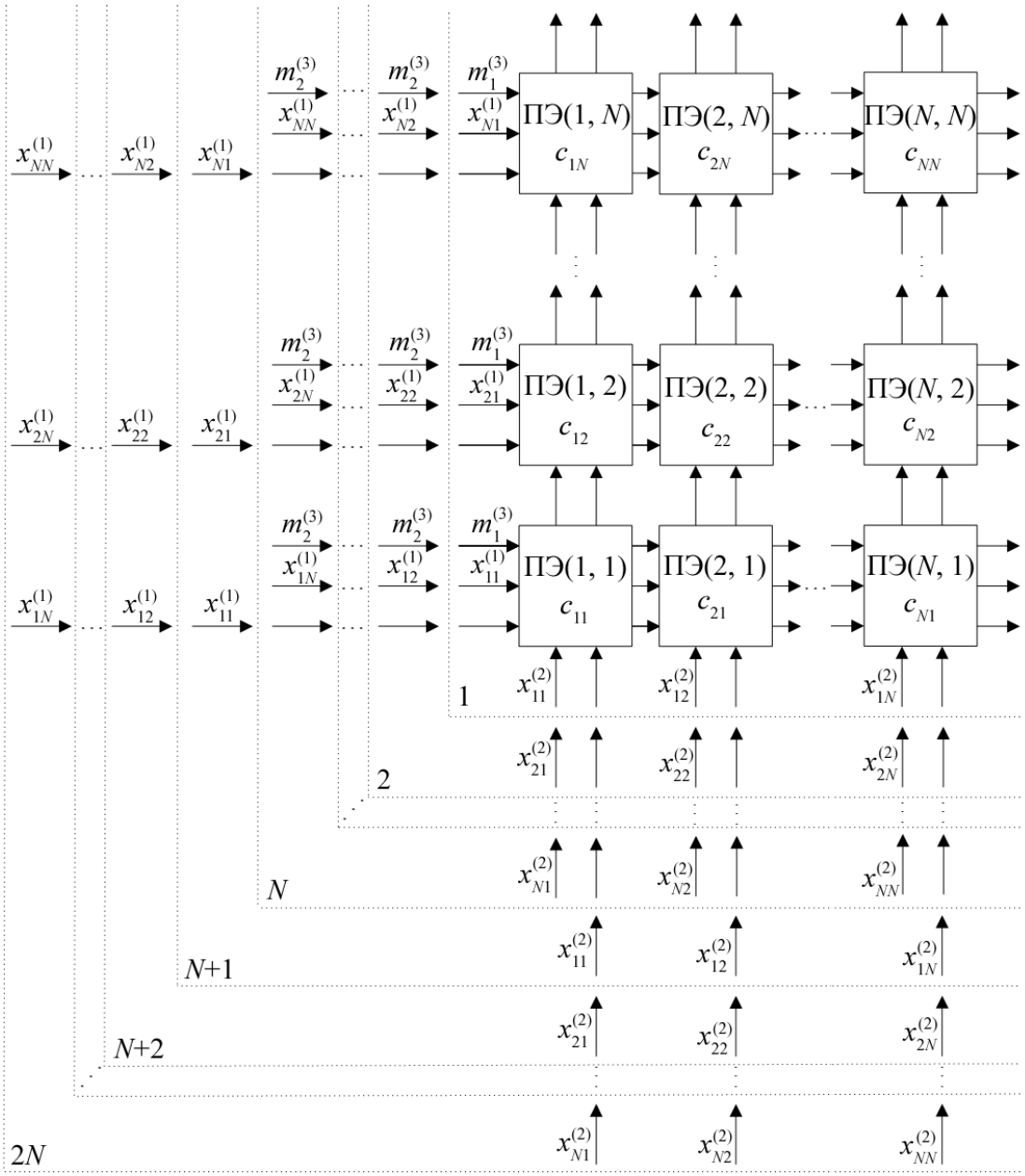

Рис. 4.4. Структура процессора для перемножения двух квадратных матриц порядка N

Таким образом, получен проект плоского вычислителя для перемножения двух квадратных матриц размером $N \times N$. Вычислитель состоит из N^2 процессорных элементов, состав и описание функционирования памяти каждого ПЭ приведены выше. Время выполнения задачи – $2N$ тактов, что на $N - 2$ такта быстрее, чем систолического вычислителя, спроектированного на основе наилучшего единого линейного таймирования и соответствующего отображения трехмерного графа зависимостей на плоскость.

ЛИТЕРАТУРА

Основная

Воеводин, В. В. Параллельные вычисления / В. В. Воеводин, Вл. В. Воеводин. СПб. : БХВ-Петербург, 2002. 608 с.

Кун, С. Матричные процессоры на СБИС / С. Кун. М. : Мир, 1991. 672 с.

Дополнительная

Адуцкевич, Е. В. Проектирование матричных процессоров с использованием расщепления и раздельного таймирования макроопераций алгоритма / Е. В. Адуцкевич, П. И. Соболевский // Тр. Ин-та математики НАН Беларуси. 2002. Т. 11. С. 22–33.

Баханович, С. В. Пространственно-временное отображение алгоритмов с однородными зависимостями на параллельные архитектуры : раздельное таймирование / С. В. Баханович, Н. А. Лиходед, П. И. Соболевский // Тр. Ин-та математики НАН Беларуси. 1999. Т. 3. С. 134–140.

Лиходед, Н. А. Один метод синтеза систолических структур, реализующих итерационные алгоритмы / Н. А. Лиходед, П. И. Соболевский, А. А. Тиунчик // Весці АН Беларусі. Сер. фіз.-мат. навук. 1992. № 3,4. С. 109–113.

Лиходед, Н. А. Построение параллельных форм заданной ширины итерационных алгоритмов для реализации на матричных процессорах / Н. А. Лиходед, П. И. Соболевский, А. А. Тиунчик // Докл. НАН Беларуси. 1995. Т. 39, № 5. С. 17–20.

Лиходед, Н. А. Распараллеливание алгоритмов на основе расщепления макроопераций / Н. А. Лиходед, П. И. Соболевский, А. А. Тиунчик // Докл. НАН Беларуси. 2001. Т. 45, № 4. С. 42–45.

Лиходед, Н. А. Стыковка базовых систолических вычислителей / Н. А. Лиходед, П. И. Соболевский // Докл. АН БССР. 1989. Т. 33, № 5. С. 389–392.

ОГЛАВЛЕНИЕ

Учебное издание

Соболевский Павел Иосифович
Адуцкевич Евгения Владимировна

ПРОЕКТИРОВАНИЕ СПЕЦПРОЦЕССОРОВ НА СБИС (ОТ АЛГОРИТМА К АРХИТЕКТУРЕ)

Пособие для студентов механико-математического факультета

Редактор *О. Н. Кохно*
Художник обложки *Т. Ю. Таран*
Технический редактор *Т. К. Раманович*
Корректор *Л. В. Рутковская*
Компьютерная верстка *Е. В. Адуцкевич*

Подписано в печать ??.??.2008. Формат $60\times84\,{}^{1}\!/_{16}$. Бумага офсетная.
Гарнитура Таймс. Печать офсетная. Усл. печ. л. ??,??.
Уч.-изд. л. 7,06. Тираж 300 экз. Зак. .

Белорусский государственный университет.
ЛИ №02330/0056804 от 02.03.2004.
200030, Минск, проспект Независимости, 4.

Отпечатано с оригинала-макета заказчика.
Республиканское унитарное предприятие
«Издательский центр Белорусского государственного университета».
ЛП №02330/0056850 от 30.04.2004.
200030, Минск, ул. Красноармейская, 6.