Глава 2

ПРОЕКТИРОВАНИЕ СПЕЦПРОЦЕССОРОВ

2.1. ТАЙМИРУЮЩАЯ ФУНКЦИЯ

Пусть алгоритм задан в виде системы рекуррентных уравнений вида (1.1). Будем считать, что граф $G = (V, E)$ строго направленный, т. е. конус допустимых направлений графа G не пуст: $K(G) = \{\, \tau \in \in \mathbf{Z}^d \mid \tau \cdot \varphi^{(k)} > 0, \quad \forall \varphi^{(k)} \in \Phi \,\} \neq \emptyset$. Любой вектор $\tau \in K(G)$ называется *направляющим вектором* графа G. Введем в рассмотрение функцию $t\colon\ \mathbf{Z}^d \to \mathbf{Z}$ вида

$$t(v) = \tau \cdot v + t_0, \quad v \in V,\ \tau \in K(G),\ t_0 \in \mathbf{Z}, \tag{2.1}$$

где «$\cdot$» означает скалярное произведение в арифметическом пространстве $\mathbf{Z}^d$.

Предположим, что любая макрооперация алгоритма выполняется за единицу времени. Значение $t(v),\ v \in V_\lambda$, будем интерпретировать как временной момент выполнения макрооперации $F_\lambda^{(k)},\ 1 \le k \le K$, приписанной точке $v \in V_\lambda$. Функцию t далее будем называть *таймирующей функцией*.

Далее будем считать, что выбранный нами вектор $\tau \in K(G)$ имеет взаимно простые координаты. В этом случае множество значений, принимаемых таймирующей функцией, заполняет целочисленный промежуток $[m, M]$ без пропусков, где $m = \min\limits_{v \in V} t(v),\ M = \max\limits_{v \in V} t(v)$. При таких условиях высота параллельной формы определяется формулой

$$|T(\tau)| = \max_{v \in V} t(v) - \min_{v \in V} t(v) + 1 = \max_{v \in V \ominus V} \tau \cdot v + 1, \tag{2.2}$$

где $V \ominus V = \{s \in \mathbf{Z}^d \mid s = v' - v'',\ v', v'' \in V\}$.

Так как любая макрооперация по предположению выполняется за единицу времени, то таймирующая функция вида (2.1) должна удовлетворять условию $t(v) - t(v - \varphi^{(k)}) \ge 1, \quad v, v - \varphi^{(k)} \in V,\ (v - \varphi^{(k)}, v) \in E,\ 1 \le k \le K$, что равносильно условию $\tau \in K(G)$.

Таймирующая функция (2.1) распределяет все вершины графа алгоритма по ярусам параллельной формы, значение таймирующей функции $t(v)$ можно трактовать как номер яруса ПФ, которому принадлежит вершина $v \in V$.

Пример перемножения двух квадратных матриц

Рассмотрим алгоритм (1.9) перемножения двух квадратных матриц. Найдем для этого алгоритма таймирующую функцию, соответствующую параллельной форме минимальной высоты (определяющую максимальную параллельную форму). Высота ПФ зависит от выбора направляющего вектора графа алгоритма G и ее наименьшее значение можно выразить формулой

$$|T_{\min}| = \min_{\tau \in K(G)} |T(\tau)| = \min_{\tau \in K(G)} (\max_{v \in V \ominus V} \tau \cdot v + 1).$$

Для алгоритма (1.9) $V \ominus V = \{(i_1, i_2, i_3) \in \mathbf{Z}^3 \mid 1 - N \leq i_1, i_2, i_3 \leq \leq N - 1\}$, следовательно, $|T_{\min}| = \min\limits_{\tau \in K(G)} ((\tau_1 + \tau_2 + \tau_3)(N-1) + 1)$. Поскольку конус допустимых направлений алгоритма (1.9) $K(G) = \{ \tau \in \mathbf{Z}^3 \mid \tau_1, \tau_2, \tau_3 > 0 \}$, то минимум высоты ПФ достигается при $\tau_1 = \tau_2 = \tau_3 = 1$ и равен $|T_{\min}| = 3N - 2$. Минимальная высота достигается при распараллеливании таймирующей функцией $t(v) = i_1 + i_2 + i_3 + t_0,\ v \in V,\ t_0 \in \mathbf{Z}$. Константу t_0 определим следующим образом: значение таймирующей функции в точке $v_{\min}$, определяемой равенством $\min\limits_{v \in V} \tau \cdot v = \tau \cdot v_{\min}$, положим равным единице. Для алгоритма (1.9) такая точка $(1, 1, 1)$. Следовательно, получим $t(1, 1, 1) = 3 + t_0 = 1$, откуда $t_0 = -2$.

Таким образом, для алгоритма (1.9) получили таймирующую функцию $t(v) = i_1 + i_2 + i_3 - 2$, которая определяет параллельную форму минимальной высоты.

2.2. ЛОКАЛЬНАЯ ПАМЯТЬ. БАЗОВЫЙ ВЫЧИСЛИТЕЛЬНЫЙ ГРАФ

Каждая вершина $v \in V$ графа зависимостей $G = (V, E)$ алгоритма (1.1), размещенного в пространстве $\mathbf{Z}^d$, помечена макрооперацией вычисления значения переменных $x^{(k)},\ 1 \leq k \leq K$, в точке $v \in V$ и

значением таймирующей функции $t(v)$, которое можно трактовать как номер такта, на котором по входящим в вершину дугам, соответствующим направлениям $\varphi^{(k)} \in \Phi$, поступают значения переменных алгоритма (1.1) $x^{(k)}(v - \varphi^{(k)}),\ 1 \le k \le K$, и осуществляется вычисление значений переменных $x^{(k)}(v),\ 1 \le k \le K$. Такая трактовка приводит к необходимости введения понятия задержки вычисленного значения $x^{(k)}(v)$ на определенное число тактов в вершине v по дуге, соответствующей направлению $\varphi^{(k)}$.

Задержкой в вершине $v \in V$ по дуге $(v, v + \varphi^{(k)}) \in E$, определяемой вектором $\varphi^{(k)} \in \Phi,\ 1 \le k \le K$, будем называть целое положительное число

$$h_{\varphi^{(k)}}(v) = t(v + \varphi^{(k)}) - t(v) = \tau \cdot \varphi^{(k)}, \quad v \in V,\ \varphi^{(k)} \in \Phi. \qquad (2.3)$$

Заметим, что задержка $h_{\varphi^{(k)}}(v),\ v \in V$, не зависит от вершины v, т. е. для каждого вектора $\varphi^{(k)}$ задержки по дугам, определяемым этим вектором, одинаковы для всех $v \in V$. Поэтому задержку по дуге $(v, v + \varphi^{(k)}) \in E$ для каждой вершины $v \in V$ будем обозначать $h_{\varphi^{(k)}}$.

Пометим каждую вершину $v \in V$ помимо приписанной ей макрооперации и значения таймирующей функции еще одной меткой – *множеством задержек* $H = \{h_{\varphi^{(k)}},\ 1 \le k \le K\}$. Множество значений таймирующей функции $t(v)$, соответствующих всем вершинам $v \in V$, обозначим $T = \{t(v),\ v \in V\}$.

Помеченный таким образом граф алгоритма (1.1), размещенный в пространстве $\mathbf{Z}^d$, будем называть *базовым вычислительным графом* (БВГ) и обозначать $G = (V, E, T, H)$, где V, E – множество вершин и множество дуг графа алгоритма (1.1), вершины графа помечены выполняемыми макрооперациями алгоритма, значениями таймирующих функций из множества T и задержками по дугам из множества H.

Базовый вычислительный граф можно рассматривать как граф семейства специализированных виртуальных вычислительных систем, реализующих алгоритм (1.1). Вершины $v \in V_\lambda$ соответствуют процессорным элементам $\text{ПЭ}_\lambda(v)$, размещенным в точках пространства $\mathbf{Z}^d$, выполняющим в момент времени $t(v)$ приписанную вершине макрооперацию и осуществляющим задержку в своей локальной памяти результатов вычислений на требуемое число тактов $h_{\varphi^{(k)}},\ 1 \le k \le K$, а дуги характеризуют соединения между процессорными элементами.

В такой интерпретации БВГ содержит Λ типов процессорных элементов: $\text{ПЭ}_\lambda(v),\ v \in V_\lambda,\ 1 \le \lambda \le \Lambda$.

Каждый $\text{ПЭ}_\lambda(v),\ v \in V_\lambda,\ 1 \le \lambda \le \Lambda$, имеет набор входов $I^{(k)}_{\varphi^{(k)}}$ и набор выходов $O^{(k)}_{\varphi^{(k)}}$. Число входов равно числу операндов макрооперации $F^{(k)}_\lambda,\ 1 \le k \le K$, по вычислению $x^{(1)}(v), \ldots, x^{(K)}(v)$. Число выходов равно числу макроопераций, использующих результат макрооперации $F^{(k)}_\lambda,\ 1 \le k \le K$, в качестве операнда. Каждый $\text{ПЭ}_\lambda(v)$ БВГ выполняет только одно вычисление (одну макроопреацию) на такте $t(v)$. Для этого в $\text{ПЭ}_\lambda(v)$ имеется соответствующее устройство, позволяющее выполнить эту операцию над входными данными, поступившими в $\text{ПЭ}_\lambda(v)$ БВГ. Каждый $\text{ПЭ}_\lambda(v),\ v \in V_\lambda$, БВГ имеет K блоков линейно связанных ячеек локальной памяти для хранения значений переменных $x^{(k)},\ 1 \le k \le K$, длина каждого такого блока $h_{\varphi^{(k)}}$: $\{R^{(k)}_\alpha(t),\ 1 \le \alpha \le h_{\varphi^{(k)}}\},\ 1 \le k \le K$. Все ячейки памяти меняют свое содержимое в зависимости от номера такта t. В первую ячейку памяти в момент времени $t = t(v)$ записывается результат выполнения операции $F^{(k)}_\lambda$: $R^{(k)}_1(t) = F^{(k)}_\lambda(I^{(1)}_{\varphi^{(1)}}, \ldots, I^{(K)}_{\varphi^{(K)}})$ (функции $F^{(k)}_\lambda,\ 1 \le k \le K$, не обязательно строго зависят от всех K переменных). На каждом такте происходит перезапись содержимого каждой ячейки памяти с номером $\alpha - 1$ в ячейку памяти с номером α, $2 \le \alpha \le h_{\varphi^{(k)}}$: $R^{(k)}_\alpha(t + \alpha - 1) = R^{(k)}_{\alpha-1}(t + \alpha - 2),\ 1 \le k \le K$. На каждый выход $O^{(k)}_{\varphi^{(k)}},\ 1 \le k \le K$, в момент времени $t(v) + h_{\varphi^{(k)}}$ подается содержимое ячейки памяти с номером $h_{\varphi^{(k)}}$:

$$O^{(k)}_{\varphi^{(k)}} = R^{(k)}_{h_{\varphi^{(k)}}}(t(v) + h_{\varphi^{(k)}} - 1).$$

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

Процессы приема, вычисления и передачи информации, происходящие в процессорных элементах $\text{ПЭ}(v),\ v \in V$, в момент времени $t = t(v)$, удобно кратко представлять в виде *описания функционирования локальной памяти* $\text{ПЭ}_\lambda(v)$:

$$
\begin{aligned}
P_\lambda(v):\{&R_1^{(k)}(t) = F_\lambda^{(k)}(I^{(1)}_{\varphi^{(1)}}, \ldots, I^{(K)}_{\varphi^{(K)}}),\\
&R_\alpha^{(k)}(t+\alpha-1) = R_{\alpha-1}^{(k)}(t+\alpha-2),\ 2 \le \alpha \le h_{\varphi^{(k)}},\\
&O^{(k)}_{\varphi^{(k)}} = R^{(k)}_{h_{\varphi^{(k)}}}(t + h_{\varphi^{(k)}} - 1),\ 1 \le k \le K\},\\
v \in V_\lambda,\ &1 \le \lambda \le \Lambda,\ t = t(v).
\end{aligned}
\tag{2.4}
$$

Заметим, что время реализации алгоритма виртуальной вычислительной системой может отличаться от высоты параллельной формы алгоритма, поскольку вывод результата из процессорных элементов, соответствующих вершинам каждого яруса ПФ, производится с задержками, не обязательно равными единице. *Время реализации алгоритма* вычисляется по формуле

$$
T(\tau) = \max_{v \in V \ominus V} \tau \cdot v + \max_r\ \tau \cdot \varphi^{(k_r)}, \tag{2.5}
$$

где $\varphi^{(k_r)}$ – вектор зависимости, соответствующий вычисляемой переменной $x^{(k_r)}$.

Пример перемножения двух квадратных матриц (продолжение)

Построим БВГ алгоритма (1.9) и опишем функционирование локальной памяти процессорных элементов виртуальной вычислительной системы, реализующей этот алгоритм.

Вычислим задержки в вершинах графа алгоритма:

$$
\begin{aligned}
h_{\varphi^{(1)}} &= h_{e_1} = (1,1,1)\cdot(1,0,0) = 1,\\
h_{\varphi^{(2)}} &= h_{e_2} = (1,1,1)\cdot(0,1,0) = 1,\\
h_{\varphi^{(3)}} &= h_{e_3} = (1,1,1)\cdot(0,0,1) = 1.
\end{aligned}
$$

Множество задержек $H = \{1,1,1\}$.

Множество значений таймирующей функции $T = \{i_1+i_2+i_3-2,\ 1 \le \le i_1, i_2, i_3 \le N\}$ заполняет отрезок $[1, 3N-2]$ без пропусков.

Теперь каждая вершина графа алгоритма (1.9) помечена значением таймирующей функции в этой вершине и множеством задержек, т. е. мы получили БВГ алгоритма (1.9).

Опишем структуру процессорных элементов соответствующей этому БВГ виртуальной вычислительной системы.

Каждый $\text{ПЭ}_1(v),\ v \in V_1$, имеет входы $I_{e_1}^{(1)}$, $I_{e_2}^{(2)}$, выходы $O_{e_1}^{(1)}$, $O_{e_2}^{(2)}$, $O_{e_3}^{(3)}$ и три ячейки локальной памяти для хранения значений переменных $x^{(1)}$, $x^{(2)}$, $x^{(3)}$: $R_1^{(1)}(t)$, $_1^{(2)}(t)$, $R_1^{(3)}(t)$ соответственно, а также содержит устройство, позволяющее выполнять операцию «умножение».

Каждый $\text{ПЭ}_2(v),\ v \in V_2$, имеет входы $I_{e_1}^{(1)}$, $I_{e_2}^{(2)}$, $I_{e_3}^{(3)}$, выходы $O_{e_1}^{(1)}$, $O_{e_2}^{(2)}$, $O_{e_3}^{(3)}$ и такие же ячейки локальной памяти, как $\text{ПЭ}_1(v),\ v \in \in V_1$. $\text{ПЭ}_2(v),\ v \in V_2$, содержит устройство, позволяющее выполнять операцию «умножение со сложением».

Описание функционирования локальной памяти $\text{ПЭ}_\lambda(v),\ v \in V_\lambda$, $\lambda = 1, 2$, имеет вид:

$$
\begin{aligned}
P_1(v):\{&R_1^{(1)}(t) = I_{e_1}^{(1)},\ O_{e_1}^{(1)} = R_1^{(1)}(t),\\
&R_1^{(2)}(t) = I_{e_2}^{(2)},\ O_{e_2}^{(2)} = R_1^{(2)}(t),\\
&R_1^{(3)}(t) = I_{e_1}^{(1)} \cdot I_{e_2}^{(2)},\ O_{e_3}^{(3)} = R_1^{(3)}(t)\ \},\ v \in V_1;\\
P_2(v):\{&R_1^{(1)}(t) = I_{e_1}^{(1)},\ O_{e_1}^{(1)} = R_1^{(1)}(t),\\
&R_1^{(2)}(t) = I_{e_2}^{(2)},\ O_{e_2}^{(2)} = R_1^{(2)}(t),\\
&R_1^{(3)}(t) = I_{e_1}^{(1)} \cdot I_{e_2}^{(2)} + I_{e_3}^{(3)},\ O_{e_3}^{(3)} = R_1^{(3)}(t)\ \},\ v \in V_2.
\end{aligned}
\tag{2.6}
$$

Поскольку задержка вычисляемой алгоритмом переменной $x^{(3)}$ равна 1, то время реализации алгоритма совпадает с высотой параллельной формы $|T_{\min}| = 3N - 2$.

Покажем, как можно получить виртуальную вычислительную систему, реализующую алгоритм перемножения двух матриц, однородной структуры, т. е. состоящую из одинаковых процессорных элементов. Для этого изменим алгоритм (1.9) следующим образом. В качестве начальных данных переменной $x^{(3)}$ в вершинах $(i_1, i_2, 0) \in \mathbf{Z}^3$ будем использовать нули и всем вершинам графа алгоритма $V = V_1 \cup V_2$ поставим в соответствие макрооперацию «умножение со сложением и переопределения». Таким образом получим алгоритм

$$
\begin{aligned}
&x^{(3)}(v) = x^{(2)}(v - e_2)x^{(1)}(v - e_1) + x^{(3)}(v - e_3), && v \in V,\\
&x^{(1)}(v) = x^{(1)}(v - e_1), && v \in V,\\
&x^{(2)}(v) = x^{(2)}(v - e_2), && v \in V.
\end{aligned}
\tag{2.7}
$$

Граф зависимостей алгоритма (2.7) при $N = 2$ изображен на рис. 1.1.

Множество векторов зависимостей и конус допустимых направлений алгоритма (2.7) не изменились. Однако структура процессорных элементов виртуальной вычислительной системы, соответствующей БВГ алгоритма (2.7), и описание функционирования локальной памяти этих ПЭ отличаются от приведенных ранее для алгоритма (1.9).

Вершины ввода начальных данных приведены в табл. 2.1.

Таблица 2.1

Вершины ввода начальных данных для алгоритма (2.7)

Данные	Вершины ввода v_{in}
$x^1_{i_3 i_2}$	$(1, i_2, i_3),\ 1 \le i_2, i_3 \le N$
$x^2_{i_1 i_3}$	$(i_1, 1, i_3),\ 1 \le i_1, i_3 \le N$
0	$(i_1, i_2, 1),\ 1 \le i_1, i_2 \le N$

Каждый ПЭ(v), $v \in V$, имеет входы $I^{(1)}_{e_1}$, $I^{(2)}_{e_2}$, $I^{(3)}_{e_3}$, выходы $O^{(1)}_{e_1}$, $O^{(2)}_{e_2}$, $O^{(3)}_{e_3}$ и три ячейки локальной памяти для хранения значений переменных $x^{(1)}$, $x^{(2)}$, $x^{(3)}$: $R^{(1)}_1(t)$, $R^{(2)}_1(t)$, $R^{(3)}_1(t)$ соответственно. Каждый ПЭ(v), $v \in V$, содержит устройство, позволяющее выполнять операцию «умножение со сложением и переопределения».

Описание функционирования локальной памяти ПЭ(v), $v \in V$, имеет вид

$$
\begin{aligned}
P(v):\{&R^{(1)}_1(t) = I^{(1)}_{e_1},\ O^{(1)}_{e_1} = R^{(1)}_1(t),\\
&R^{(2)}_1(t) = I^{(2)}_{e_2},\ O^{(2)}_{e_2} = R^{(2)}_1(t),\\
&R^{(3)}_1(t) = I^{(1)}_{e_1} \cdot I^{(2)}_{e_2} + I^{(3)}_{e_3},\ O^{(3)}_{e_3} = R^{(3)}_1(t)\ \},\ v \in V.
\end{aligned}
$$

2.3. ОТОБРАЖЕНИЕ БАЗОВЫХ ВЫЧИСЛИТЕЛЬНЫХ ГРАФОВ

Пусть алгоритм задан в виде системы рекуррентных уравнений вида (1.1) и описывается d-мерным БВГ $G = (V, E, T, H)$, $V \subset \mathbf{Z}^d$. Для реализации такого алгоритма на матричном процессоре возникает необходимость отображения БВГ в пространства меньших размерностей (в физически реализуемые пространства $\mathbf{Z}^1, \mathbf{Z}^2, \mathbf{Z}^3$). При

этом необходимо указать, в каком ПЭ матричного процессора и в какой момент времени будут выполнены операции, соответствующие вершинам БВГ. Полученную после отображения структуру будем называть *вычислительным графом* (ВГ). Вычислительный граф является графом вычислительной системы, реализующей алгоритм (1.1). В каждом $\text{ПЭ}_\lambda(v),\ v \in V_\lambda,\ 1 \le \lambda \le \Lambda$, виртуальной вычислительной системы, соответствующей БВГ, вычисление значений переменных $x^{(k)}(v),\ v \in V_\lambda$, производится только один раз в соответствии с макрооперацией $F_\lambda^{(k)},\ 1 \le k \le K$, приписанной вершине $v \in V_\lambda$, в момент времени, равный значению таймирующей функции $t(v)$. В каждом процессорном элементе вычислительной системы, соответствующей ВГ, вычисление значений переменных может производиться несколько раз, в зависимости от числа прообразов вершины ВГ.

Функцию, осуществляющую отображение d-мерных алгоритмов в r-мерную архитектуру, будем называть *функцией размещения*. Рассмотрим функцию размещения, задаваемую линейным оператором π: $\mathbf{Z}^d \to \mathbf{Z}^r$. Оператор π не должен порождать связей, зависящих от размера задачи, т. е. $||\pi(\varphi^{(k)})||_\infty \le \text{const}$ (векторы $\pi(\varphi^{(k)})$ должны иметь координаты, являющиеся постоянными величинами, не зависящими от размера задачи). Если функция размещения π отображает точки $v' \in V_{\lambda_1}$ и $v'' \in V_{\lambda_2}$ БВГ в одну и ту же вершину ВГ, то приписанные этим точкам макрооперации $F_{\lambda_1}^{(k)},\ 1 \le k \le K$, и $F_{\lambda_2}^{(k)},\ 1 \le k \le K$, должны выполняться в разное время, т. е.

$$t(v') \ne t(v''), \quad \text{если } \pi(v') = \pi(v''),\ v' \in V_{\lambda_1},\ v'' \in V_{\lambda_2},\ v' \ne v''. \qquad (2.8)$$

Если $\lambda_1 \ne \lambda_2$, то ограничение (2.8) необходимо для того, чтобы в каждый момент времени работы $\text{ПЭ}(\pi(v'))$ вычислительной системы, соответствующей ВГ, выполнялось вычисление переменных $x^{(k)},\ 1 \le \le k \le K$, в вершинах одного типа. Если $\lambda_1 = \lambda_2$, то ограничение (2.8) необходимо для экономии оборудования в $\text{ПЭ}(\pi(v'))$ так, чтобы макрооперации, приписанные вершинам-прообразам одного типа, выполнялись в $\text{ПЭ}(\pi(v'))$ на одном и том же оборудовании в разное время. Ограничение (2.8) называют *условием совместимости таймирующей функции t и функции размещения π*.

Рассмотрим простейший случай, когда размерности БВГ и получаемого ВГ отличаются на единицу ($d - r = 1$). В силу линейности опе-

ратора π из условия $\pi(v') = \pi(v'')$ следует $\pi(v' - v'') = 0$. Поскольку $t(v) = \tau \cdot v + t_0$, то условие $t(v') \neq t(v'')$ равносильно условию $\tau \cdot (v' - v'') \neq 0$. Следовательно, условие (2.8) можно записать в следующем виде:

$$\tau \cdot s \neq 0, \text{ если } \pi(s) = 0, \ s \neq 0. \tag{2.9}$$

Если $d - r \geq 2$, то условие (2.9) допускает следующее обобщение.

Теорема 2.1. *Таймирующая функция (2.1) совместима с функцией размещения π тогда и только тогда, когда*

$$\tau \cdot s \neq 0, \quad s \in S, \ S = \{s \in V \ominus V \ s \neq 0 \mid \pi(s) = 0\}, \tag{2.10}$$

где $V \ominus V = \{s \in \mathbf{Z}^d \mid s = v' - v'', \ v', \ v'' \in V\}$.

Доказательство. Докажем необходимость. Выполняется условие (2.8), требуется получить условие (2.10). Пусть $v', v'' \in V, \ v' \neq v''$ и $\pi(v') = \pi(v'')$. Тогда в силу линейности отображения π имеем $\pi(v' - v'') = 0$. Обозначим $s = v' - v''$. Так как $s = v' - v''$ и $v', v'' \in V$, то $s \in V \ominus V$; так как $\pi(s) = \pi(v' - v'') = 0$ и $s \neq 0$ в силу того, что $v' \neq v''$, то $s \in S$.

Вычислим разность $t(v') - t(v'')$, где t – функция вида (2.1): $t(v') - t(v'') = \tau \cdot v' + t_0 - \tau \cdot v'' - t_0 = \tau \cdot (v' - v'') = \tau \cdot s$. Так как $t(v') \neq t(v'')$, то $\tau \cdot s \neq 0, \ s \in S$. Необходимость доказана.

Докажем достаточность. Выполняется условие (2.10), требуется получить условие (2.8). Доказательство проведем от противного. Пусть $v', v'' \in V, \ v' \neq v'', \ \pi(v') = \pi(v'')$, такие, что $t(v') = t(v'')$. Поскольку функция t имеет вид (2.1), то получим $\tau \cdot v' + t_0 = \tau \cdot v'' + t_0$, т. е. $\tau \cdot (v' - v'') = 0$.

Поскольку $v', v'' \in V$, то во множестве $V \ominus V$ найдется такой элемент s_0, что $s_0 = v' - v''$. Так как $v' \neq v''$ и $\pi(v') = \pi(v'')$, то $s_0 = v' - v'' \neq 0$, и $\pi(s_0) = \pi(v' - v'') = \pi(v') - \pi(v'') = 0$, т. е. $s_0 \in S$. Тогда получим противоречие $\tau \cdot s_0 = 0, \ s_0 \in S$. Достаточность доказана.

Теорему 2.1 можно переформулировать иначе, если условие (2.10) записать в виде

$$\{s \in V \ominus V \mid \tau \cdot s \neq 0, \ \pi(s) = 0, \ s \neq 0\} = \emptyset. \tag{2.11}$$

Обозначим $\Pi \in \mathbf{Z}^{r \times d}$ – матрицу преобразования π. Тогда условие (2.11) эквивалентно требованию, чтобы однородная система линейных алгебраических уравнений с матрицей $\begin{pmatrix} \Pi \\ \tau \end{pmatrix} \in \mathbf{Z}^{(r+1) \times d}$ либо

имела только тривиальное решение, либо решения, не принадлежащие множеству $V \ominus V$.

Условие (2.10) мало конструктивно. С его помощью можно лишь проверить, удовлетворяют ли условию (2.8) конкретные оператор π и таймирующая функция t. Приведем основанное на теореме 2.1 правило, позволяющее во многих случаях для заданного отображения π найти множество всех таймирующих функций вида (2.1), удовлетворяющих вместе с отображением π условию (2.8).

1. Выбрать в ядре $\mathrm{Ker}\pi$ оператора π базис $g_1, \ldots, g_{d-r}$, обладающий следующими свойствами: $g_i \in \mathbf{Z}^d,\ 1 \le i \le d-r$, и для всякого $v \in \mathrm{Ker}\pi \cap \mathbf{Z}^d$ существуют $c_1, ..., c_{d-r} \in \mathbf{Z}$, такие, что $v = \sum\limits_{i=1}^{d-r} c_i g_i$ (т. е. выбранный базис позволяет представить любой элемент с целочисленными координатами из ядра оператора π в виде линейной комбинации с целочисленными коэффициентами).

2. Поставить в соответствие каждому вектору $s \in S$ вектор $c = (c_1, \ldots, c_{d-r}) \in \mathbf{Z}^{d-r}$, составленный из коэффициентов разложения вектора s по базису $\{g_i,\ 1 \le i \le d-r\},\ s = \sum\limits_{i=1}^{d-r} c_i g_i$. Образ множества S при таком отображении обозначим C. Так как

$$\tau \cdot s = \sum_{i=1}^{d-r} c_i (\tau \cdot g_i) = \sum_{i=1}^{d-r} c_i \eta_i = \eta \cdot c,$$

где $\eta = (\eta_1, \ldots, \eta_{d-r}),\ \eta_i = \tau \cdot g_i,\ 1 \le i \le d-r$, то условие (2.10) преобразуется к виду

$$\eta \cdot c \neq 0 \text{ для } c \in C. \tag{2.12}$$

Условие (2.12) означает, что векторы $(c_1, ..., c_{d-r})$ не принадлежат проходящей через начало координат гиперплоскости с нормальным вектором $\eta = (\eta_1, \ldots, \eta_{d-r})$.

3. Определить в $\mathbf{Z}^{d-r}$ все гиперплоскости с нормальными векторами $\eta \in \mathbf{Z}^{d-r}$, каждая из которых проходит через начало координат и не содержит векторов (точек) множества C.

4. Все векторы $\tau \in K(G)$, при которых выполняется условие (2.12), а следовательно, и условие (2.8), найти из систем уравнений

$$\tau \cdot g_i = \eta_i,\ 1 \le i \le d-r. \tag{2.13}$$

В случае $d - r = 2$ полезным оказывается следующее утверждение, позволяющее определять все векторы η, удовлетворяющие условию (2.12).

Теорема 2.2. *Пусть* $(\eta_1, \eta_2) \in \mathbf{Z}^2$, C *– симметричное относительно точки* $(0,0)$ *множество такое, что вместе с каждой точкой* (c_1, c_2) *оно содержит все точки с целочисленными координатами промежутка* $[(0,0),(c_1,c_2)]$. *Тогда для фиксированных* (η_1, η_2) *условие*

$$\eta_1 c_1 + \eta_2 c_2 \neq 0 \text{ для всех } (c_1, c_2) \in C \tag{2.14}$$

выполнено тогда и только тогда, когда $(\eta_2/q, -\eta_1/q) \notin C$, *где* q *– наибольший общий делитель чисел* η_1 *и* η_2.

Доказательство. Пусть выполнено условие (2.14). Тогда $(\eta_2/q, -\eta_1/q) \notin C$, так как в противном случае должно иметь место $\eta_1(\eta_2/q) + \eta_2(-\eta_1/q) \neq 0$, что неверно.

Обратно, пусть $(\eta_2/q, -\eta_1/q) \notin C$. Так как q – наибольший общий делитель чисел η_1 и η_2, то точка $p = (p_1, p_2) = (\eta_2/q - \eta_1/q)$ имеет взаимно простые координаты. Рассмотрим прямую с нормальным вектором $\eta = (\eta_1, \eta_2)$, проходящую через начало координат. Точка p принадлежит этой прямой, но не принадлежит области C. Поэтому не существует точки $(c_1, c_2) \in C$ такой, что $\eta_1 c_1 + \eta_2 c_2 = 0$: все точки с целочисленными координатами, лежащие на прямой, кратны точке с целочисленными взаимно простыми координатами, и в силу условий теоремы было бы $(p_1, p_2) \in C$, что неверно.

Следствие 2.1. *Всякий вектор* η, *удовлетворяющий условию (2.14), имеет вид* $\eta = (-qp_2, qp_1)$, *где* $q \in \mathbf{Z}\backslash\{0\}$, $(p_1, p_2) \in \mathbf{Z}^2\backslash C$, p_1 *и* p_2 *взаимно просты.*

Замечание 2.1. *Если выбрать вектор* η *указанного вида, то высота параллельной формы, задаваемая вычислительным графом (время выполнения алгоритма) определяется функцией, линейно зависящей от координат точки* (p_1, p_2). *Решая задачу минимизации этой функции на множестве точек* $(p_1, p_2) \in \mathbf{Z}^2\backslash C$ *со взаимно простыми координатами, найдем точку, в которой этот минимум достигается, и соответствующее ей значение вектора* τ. *Как правило, минимум функции достигается в одной из точек, расположенных рядом с областью* C.

В случае $d - r > 2$ для нахождения всех векторов η, удовлетворяющих условию (2.12), часто бывает удобным пользоваться следующим утверждением.

Теорема 2.3. *Пусть $d - r = m > 2$ и область $C \subset \mathbf{Z}^m$ является прямоугольным параллелепипедом вида $C = \{c \in \mathbf{Z}^m \mid |c_1| \leq \leq J_1 - 1, \ldots, |c_m| \leq J_m - 1\}$. Возьмем любое $\alpha_1 \in \{1, 2, \ldots, m\}$. Положим $|\eta_{\alpha_1}| = 1$. Затем возьмем любое $\alpha_2 \in \{1, 2, \ldots, m\} \backslash \{\alpha_1\}$ и положим $|\eta_{\alpha_2}| = J_{\alpha_1}$. Затем выберем произвольно $\alpha_3 \in \in \{1, 2, \ldots, m\} \backslash \{\alpha_1, \alpha_2\}$ и положим $|\eta_{\alpha_3}| = J_{\alpha_1} J_{\alpha_2}$, и т. д. Для последнего $\alpha_m \in \{1, 2, \ldots, m\} \backslash \{\alpha_1, \alpha_2, \ldots, \alpha_{m-1}\}$ положим $|\eta_{\alpha_m}| = = J_{\alpha_1} J_{\alpha_2} \ldots J_{\alpha_{m-1}}$. Полученный таким образом вектор η удовлетворяет условию (2.12).*

Доказательство. Так как $s = \sum\limits_{i=1}^{m} c_i g_i \in S$, то не все c_i равны нулю. Пусть κ – наибольшее число из множества $\{1, 2, \ldots, m\}$, для которого $c_{\alpha_\kappa} \neq 0$. При $\kappa = 1$ справедливость теоремы очевидна. Пусть $\kappa > 1$. Имеем: $\eta \cdot c = \sum\limits_{i=1}^{m} c_i \eta_i = \sum\limits_{i=1}^{m} c_{\alpha_i} \eta_{\alpha_i} = \sum\limits_{i=1}^{\kappa} c_{\alpha_i} \cdot \eta_{\alpha_i} = \sum\limits_{i=1}^{\kappa-1} c_{\alpha_i} \eta_{\alpha_i} + c_{\alpha_\kappa} \eta_{\alpha_k}$, где $J_{\alpha_0} = 1$ по определению, $|c_{\alpha_\kappa}| \geq 1$ в силу целочисленности, $|c_{\alpha_i}| \leq \leq J_{\alpha_i} - 1$ по условию. Из оценок

$$|\sum_{i=1}^{\kappa-1} c_{\alpha_i} \eta_{\alpha_i}| \leq \sum_{i=1}^{\kappa-1} |c_{\alpha_i}| \prod_{j=0}^{i-1} J_{\alpha_j} \leq \sum_{i=1}^{\kappa-1} (J_{\alpha_i} - 1) \prod_{j=0}^{i-1} J_{\alpha_j} =$$
$$= \sum_{i=1}^{\kappa-1} (\prod_{j=0}^{i} J_{\alpha_j} - \prod_{j=0}^{i-1} J_{\alpha_j}) = \prod_{j=1}^{\kappa-1} J_{\alpha_j} - 1 < \prod_{j=1}^{\kappa-1} J_{\alpha_j},$$
$$|c_{\alpha_\kappa} \eta_{\alpha_k}| = |c_{\alpha_\kappa}| \prod_{j=1}^{\kappa-1} J_{\alpha_j} \geq \prod_{j=1}^{\kappa-1} J_{\alpha_j},$$

следует отличие от нуля $\eta \cdot c$ для любого $c = (c_1, \ldots, c_{d-r}) \in C$.

Пусть выбрана функция размещения π и найдена совместимая с ней таймирующая функция t. Описание функционирования локальной памяти процессорных элементов ВГ можно получить из описания функционирования локальной памяти ПЭ исходного БВГ.

Определим компоненты ВГ $G^\pi = (V^\pi, E^\pi, T^\pi, H^\pi)$:

$V^\pi = \{\widetilde{v} \in \mathbf{Z}^r \mid \widetilde{v} = \pi(v),\ v \in V\}$ – множество вершин ВГ, размещенных в пространстве $\mathbf{Z}^r$, в частности $V_\lambda^\pi = \{\widetilde{v} \in \mathbf{Z}^r \mid \widetilde{v} = = \pi(v),\ v \in V_\lambda\}$, $\bigcup\limits_{\lambda=1}^{\Lambda} V_\lambda^\pi = V^\pi$;

$E^{\pi} = \{(\widetilde{v}_1, \widetilde{v}_2) \in \mathbf{Z}^r \times \mathbf{Z}^r \mid \widetilde{v}_1 = \pi(v_1),\ \widetilde{v}_2 = \pi(v_2),\ (v_1, v_2) \in E\}$ – множество дуг ВГ;

$t^{\pi}(\widetilde{v}) = \{t(v) \mid \widetilde{v} = \pi(v),\ v \in V\},\ \widetilde{v} \in V^{\pi}$, — многозначная таймирующая функция, для которой выполняется условие совместимости таймирующей функции и функции размещения (2.8). Значения $T^{\pi} = \{\, t^{\pi}(\widetilde{v}) \mid \widetilde{v} \in V^{\pi} \,\}$ для каждого $\widetilde{v} \in V^{\pi}$ определяют номера тактов, на которых процессорный элемент $\text{ПЭ}(\widetilde{v})$ вычислительной системы, соответствующий вершине $\widetilde{v} \in V^{\pi}$ вычислительного графа, выполняет операции, приписанные прообразам этой вершины в БВГ;

$H^{\pi} = H$ — множество задержек, которое определяет длины блоков ячеек локальной памяти процессорных элементов. Множество H не меняется при отображении π.

Для хранения результатов процессорный элемент типа λ $\text{ПЭ}_{\lambda}(\widetilde{v}),\ \widetilde{v} = \pi(v),\ v \in V_{\lambda}$, имеет K блоков локальной памяти, таких же, как у всех прообразов вершины $\widetilde{v} \in V^{\pi}$.

Описание функционирования локальной памяти определяется принадлежностью векторов $\varphi^{(k)} \in \Phi$ ядру оператора π. При выполнении условия $\bigcap\limits_{\lambda=1}^{\Lambda} V_{\lambda}^{\pi} = \emptyset$ формально описание функционирования локальной памяти процессорного элемента типа λ $\text{ПЭ}_{\lambda}(\widetilde{v}),\ \widetilde{v} = \pi(v),\ v \in V_{\lambda}$, вычислительной системы можно получить из описания (2.4) функционирования локальной памяти процессорного элемента $\text{ПЭ}_{\lambda}(v),\ v \in V_{\lambda}$, виртуальной вычислительной системы следующим образом. Каждый вектор $\varphi^{(k)}$, не принадлежащий ядру оператора π, т. е. $\pi(\varphi^{(k)}) \neq 0$, следует заменить на вектор $\pi(\varphi^{(k)})$. Если для некоторого $1 \leq k_0 \leq K$ вектор $\varphi^{(k_0)}$ принадлежит ядру оператора π, т. е. $\pi(\varphi^{(k_0)}) = 0$, то следует заменить $I^{(k_0)}_{\varphi^{(k_0)}}$ на $R^{(k_0)}_{h_{\varphi^{(k_0)}}}(t-1)$ и опустить равенство, соответствующее подаче результата вычислений на выход $O^{(k_0)}_{\varphi^{(k_0)}}$.

Осуществление ввода и вывода данных в граничных процессорных элементах вычислительной системы

Одно из принципиальных ограничений, накладываемых СБИС технологией при проектировании спецпроцессоров, — ввод (вывод) данных только через граничные ПЭ спецпроцессора. Рассматриваемое в этом разделе линейное отображение БВГ в пространства физически

реализуемых размерностей $\mathbf{Z}^1,\ \mathbf{Z}^2,\ \mathbf{Z}^3$ приводит, как правило, к отображению вершин ввода (вывода) данных во внутренние процессорные элементы проектируемого спецпроцессора. Для устранения этого недостатка введем дополнительные магистрали, связывающие соответствующие вершины ввода (вывода) с вершинами БВГ, которые выбранным линейным оператором отображаются в граничные вершины ВГ. Введение новых магистралей для транспортировки начальных данных, как правило, приводит к образованию новых типов вершин и как следствие к усложнению проектируемого спецпроцессора. Иногда корректный ввод (вывод) данных можно осуществить, продлив уже имеющиеся в БВГ магистрали до вершин, отображаемых линейным оператором в граничные ПЭ спецпроцессора. Рассмотрим этот подход на примере проектирования плоского спецпроцессора для перемножения двух квадратных матриц.

Пример проектирования плоского спецпроцессора для перемножения двух квадратных матриц

В качестве реализуемого алгоритма возьмем алгоритм (1.9), в качестве таймирующей функции — функцию, определяющую максимальную параллельную форму этого алгоритма: $t(i_1, i_2, i_3) = i_1 + i_2 + i_3 + N - 3,\ 1 \le i_1, i_2, i_3 \le N$, а в качестве линейного оператора отображения — оператор $\pi\colon \mathbf{Z}^3 \to \mathbf{Z}^2$, задаваемый матрицей $\Pi = \begin{pmatrix} 1 & -1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$.

При таком выборе оператора отображения почти все вершины ввода элементов перемножаемых матриц отобразятся во внутренние вершины вычислительного графа. Действительно, точки $\pi(v^1_{in}) = \pi(1, i_2, i_3) = (1 - i_2, i_3)$ для ввода $x^1_{i_3 i_2}$ при $1 \le i_2 \le N-1,\ 1 \le i_3 \le N$, и точки $\pi(v^2_{in}) = \pi(i_1, 1, i_3) = (i_1 - 1, i_3)$ для ввода $x^2_{i_1 i_3}$ при $1 \le i_1 \le N-1,\ 1 \le i_3 \le N$, являются внутренними точками области $V^\pi = \{\, (x, y) \in \mathbf{Z}^2 \mid |x| \le N-1,\ 1 \le y \le N \,\}$. Продлевая магистрали для транспортировки элементов перемножаемых матриц путем добавления вершин (того же типа, что и вершины ввода) до пересечения с плоскостями $i_1 - i_2 = \pm(N-1)$, мы расширим индексные пространства:

$$\widehat{V}_1 = \{\, (i_1, i_2, i_3) \in \mathbf{Z}^3 \mid -N+2 \le i_1 \le 0,\ 1 \le i_2 \le i_1 + N - 1,\ i_3 = 1 \,\} \cup$$
$$\cup \{\, (i_1, i_2, i_3) \in \mathbf{Z}^3 \mid 1 \le i_1 \le N,\ i_1 - N + 1 \le i_2 \le N,\ i_3 = 1 \,\};$$

$$\widehat{V}_2 = \{\,(i_1,i_2,i_3) \in \mathbf{Z}^3 \mid -N+2 \le i_1 \le 0,\ 1 \le i_2 \le i_1+N-1,\ 1 < i_3 \le N\,\} \cup \{\,(i_1,i_2,i_3) \in \mathbf{Z}^3 \mid 1 \le i_1 \le N,\ i_1-N+1 \le i_2 \le N,\ 1 < i_3 \le N\,\};$$

$$\widehat{V} = \widehat{V}_1 \cup \widehat{V}_2 = \{\,(i_1,i_2,i_3) \in \mathbf{Z}^3 \mid -N+2 \le i_1 \le 0,\ 1 \le i_2 \le i_1+N-1,\ 1 \le i_3 \le N\,\} \cup \{\,(i_1,i_2,i_3) \in \mathbf{Z}^3 \mid 1 \le i_1 \le N,\ i_1-N+1 \le i_2 \le N,\ 1 \le i_3 \le N\,\}.$$

В расширенном индексном пространстве $\widehat{V}$ новые вершины ввода $\widehat{v}^1_{in} = (1-N+i_2, i_2, i_3)$, $\widehat{v}^2_{in} = (i_1, i_1-N+1, i_3) \in \widehat{V}$ оператором π теперь отобразятся в граничные процессорные элементы проектируемого матричного спецпроцессора: $\pi(\widehat{v}^1_{in}) = (1-N, i_3)$ для $x^1_{i_3 i_2}$ и $\pi(\widehat{v}^2_{in}) = (N-1, i_3)$ для $x^2_{i_1 i_3}$, $1 \le i_1, i_2, i_3 \le N$. Расширенный БВГ при $N = 2$ изображен на рис. 2.1.

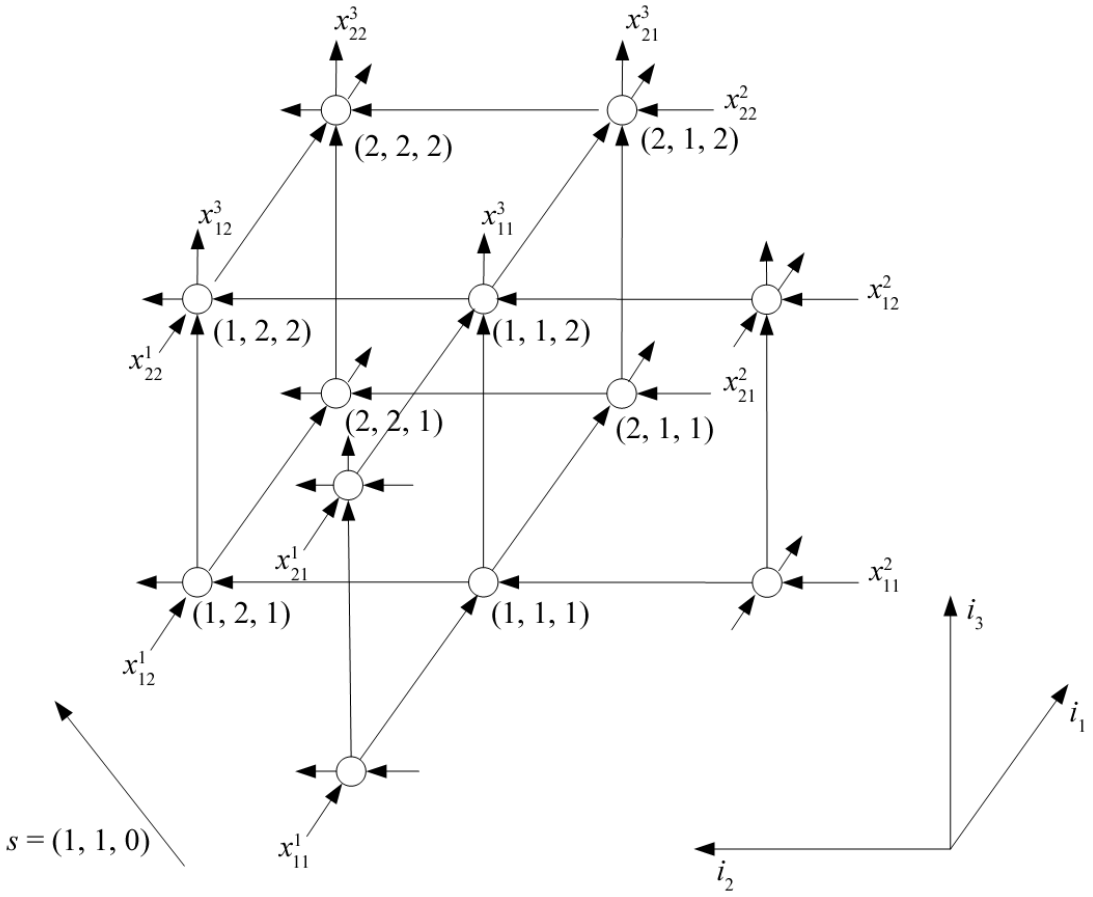


Рис. 2.1. Расширенный БВГ при $N = 2$; s – направляющий вектор проецирования (отображения); $\pi(s) = 0$

Проверим выполнение условия (2.9) совместимости таймирующей функции и функции размещения. Для таких s, что $\pi(s) = 0,\ s \neq \neq 0$, т. е. для $s = (i_1, i_2, i_3),\ i_1 = i_2 \neq 0,\ i_3 = 0$, условие $\tau \cdot s \neq 0$, т. е. $i_1 + i_2 + i_3 \neq 0$, выполняется. Следовательно, выбранные оператор отображения и таймирующая функция совместимы друг с другом.

Определим области размещения процессорных элементов:
$\widehat{V}_1^{\pi} = \{\, (x, y) \in \mathbf{Z}^2 \mid |x| \leq N - 1,\ y = 1 \,\}$;
$\widehat{V}_2^{\pi} = \{\, (x, y) \in \mathbf{Z}^2 \mid |x| \leq N - 1,\ 1 < y \leq N \,\}$.

Так как $\widehat{V}_1^{\pi} \cap \widehat{V}_2^{\pi} = \emptyset$, то проектируемый матричный спецпроцессор имеет два типа процессорных элементов, расположенных соответственно в области $\widehat{V}_1^{\pi}$ и $\widehat{V}_2^{\pi}$.

Образы векторов зависимостей алгоритма (1.9) при отображении π:

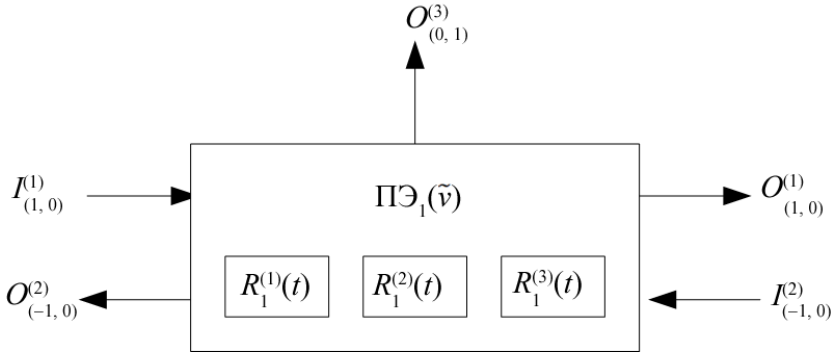
$$\pi(e_1) = (1, 0),\ \pi(e_2) = (-1, 0),\ \pi(e_3) = (0, 1).$$

Видно, что оператор π не порождает связей, зависящих от размера задачи.

Многозначная таймирующая функция $t^{\pi}(\widetilde{v}) = t^{\pi}(x, y) = \{\, i_1 + i_2 + + i_3 + N - 3 \mid i_1 - i_2 = x,\ i_3 = y,\ (i_1, i_2, i_3) \in \widehat{V} \,\},\ \widetilde{v} \in \widehat{V}^{\pi}$.

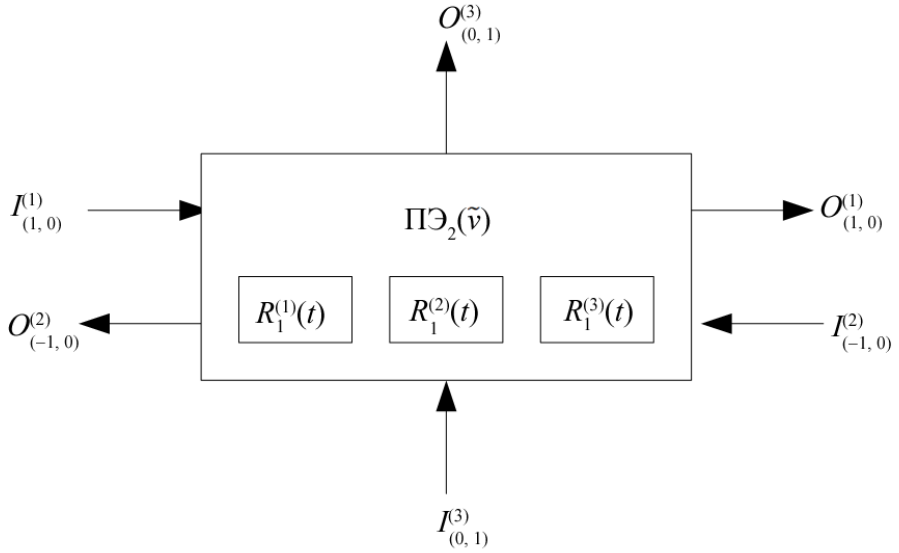
Учитывая равенства (2.6) опишем функционирование локальной памяти процессорных элементов,

Для процессорных элементов первого типа $\text{ПЭ}_1(\widetilde{v}),\ \widetilde{v} \in \widehat{V}_1^{\pi}$, имеем



$$P_1^{\pi}(\widetilde{v}) \colon \{\, R_1^{(1)}(t) = I^{(1)}_{(1,0)},\ O^{(1)}_{(1,0)} = R_1^{(1)}(t),$$
$$R_1^{(2)}(t) = I^{(2)}_{(-1,0)},\ O^{(2)}_{(-1,0)} = R_1^{(2)}(t),$$
$$R_1^{(3)}(t) = I^{(1)}_{(1,0)} \cdot I^{(2)}_{(-1,0)},\ O^{(3)}_{(0,1)} = R_1^{(3)}(t) \,\}.$$

Для процессорных элементов второго типа $\text{ПЭ}_2(\widetilde{v}),\ \widetilde{v} \in \widehat{V}_2^{\pi}$, имеем



$$P_2^{\pi}(\widetilde{v})\colon \{\, R_1^{(1)}(t) = I^{(1)}_{(1,0)},\ O^{(1)}_{(1,0)} = R_1^{(1)}(t),$$
$$R_1^{(2)}(t) = I^{(2)}_{(-1,0)},\ O^{(2)}_{(-1,0)} = R_1^{(2)}(t),$$
$$R_1^{(3)}(t) = I^{(1)}_{(1,0)} \cdot I^{(2)}_{(-1,0)} + I^{(3)}_{(0,1)},\ O^{(3)}_{(0,1)} = R_1^{(3)}(t)\,\}.$$

Полное время решения задачи спроектированным матричным спецпроцессором $4N-3$ такта.

Режим ввода начальных данных в граничные процессорные элементы $\text{ПЭ}(-(N-1), i_3),\ \text{ПЭ}(N-1, i_3)$ и вывода результатов приведем в табл. 2.2.

Таблица 2.2

Режим ввода начальных данных и вывода результатов плоского спецпроцессора для перемножения двух квадратных матриц

Данные	$\widehat{v}_{in}$	$\text{ПЭ}(\pi(\widehat{v}_{in}))$	Порт	$t(\widehat{v}_{in})$	Изменение параметров
$x^1_{i_3 i_2}$	$(i_2 - N + 1, i_2, i_3)$	$\text{ПЭ}(-(N-1), i_3)$	$I^{(1)}_{(1,0)}$	$2i_2 + i_3 - 2$	$1 \le i_2, i_3 \le N$
$x^2_{i_1 i_3}$	$(i_1, i_1 - N + 1, i_3)$	$\text{ПЭ}(N-1, i_3)$	$I^{(2)}_{(-1,0)}$	$2i_1 + i_3 - 2$	$1 \le i_1, i_3 \le N$
Результат	v_{out}	$\text{ПЭ}(\pi(v_{out}))$	Порт	$t(v_{out})$	Изменение параметров
$x^3_{i_1 i_2}$	(i_1, i_2, N)	$\text{ПЭ}(i_1 - i_2, N)$	$O^{(3)}_{(0,1)}$	$i_1 + i_2 + 2N - 3$	$1 \le i_1, i_2 \le N$

Структурная схема (при $N = 2$) спроектированного матричного спецпроцессора изображена на рис. 2.2.

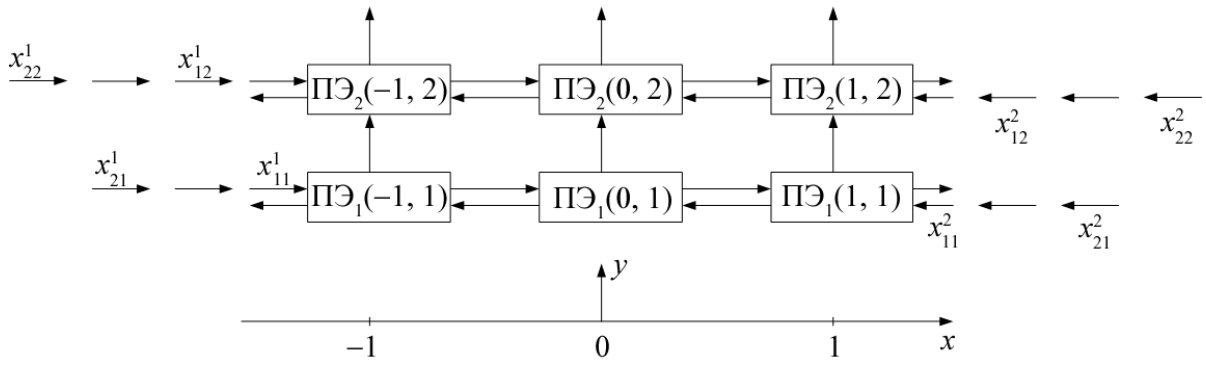


Рис. 2.2. Структурная схема матричного спецпроцессора при $N = 2$

2.4. УПРАВЛЕНИЕ

Пусть алгоритм задан в виде системы рекуррентных уравнений вида (1.1). При отображении БВГ алгоритма из пространства $\mathbf{Z}^d$ в пространство $\mathbf{Z}^r$ может возникнуть ситуация, когда в одну и ту же вершину ВГ отобразятся вершины БВГ разных типов (в такой вершине ВГ вычисление переменных $x^{(k)}$ должно производиться по разным правилам). В этом случае для того, чтобы было возможно управлять работой ПЭ($\widetilde{v}$), осуществим разметку вершин БВГ разных типов соответствующими управляющими метками.

Для введения управляющих меток необходимым, как уже было сказано, является выполнение следующего условия:

$$V^{\pi}_{\lambda_1} \cap V^{\pi}_{\lambda_2} \neq \emptyset,\ \lambda_1 \neq \lambda_2,\ 1 \leq \lambda_1, \lambda_2 \leq \Lambda. \tag{2.15}$$

Пометим каждую вершину $v \in V_\lambda,\ 1 \leq \lambda \leq \Lambda$, управляющей меткой m_λ. Областью влияния управляющей метки m_λ является множество V_λ: $\Omega(m_\lambda) = V_\lambda$. Пусть множество V_λ имеет общую границу с границей области V: $\partial V_\lambda \subseteq \partial V$. Тогда во многих случаях разметку вершин V_λ можно осуществить, используя принцип магистральной рассылки. Для этого алгоритм (1.1) дополняется управляющей переменной m, которая осуществляет транспортировку метки m_λ от границы ∂V_λ ко

всем внутренним точкам области V_λ по магистралям, определяемым вектором зависимости $\varphi^{(m)}$:

$$\begin{gathered} m(v) = m(v - \varphi^{(m)}), \quad v \in V; \\ m(v - \varphi^{(m)}) = m_\lambda, \quad v \in \partial V_\lambda,\ v - \varphi^{(m)} \notin V_\lambda. \end{gathered} \tag{2.16}$$

При необходимости управляющих переменных может быть несколько.

Для транспортировки управляющих меток следует брать вектор $\varphi^{(m)}$, не попадающий в ядро оператора отображения $\pi\colon\ \pi(\varphi^{(m)}) \neq 0$, а для ввода управляющих меток – те участки границы ∂V_λ, которые оператором π отображаются на границу области $\pi(V)$.

Рассмотрим влияние управления на временные характеристики вычислительной системы. Вектор $\varphi^{(m)}$ определяет дуги между вершинами $v \in V$ и $v + \varphi^{(m)} \in V$. Задержка в вершинах $v \in V$, согласно формуле (2.3), вычисляется следующим образом: $h_{\varphi^{(m)}} = \tau \cdot \varphi^{(m)}$. Если $\varphi^{(m)} \in \Phi$, то введенное управление не увеличивает высоту ПФ алгоритма, определяемую формулой (2.2), поскольку не изменяет конус допустимых направлений $K(G)$ и область вычисления алгоритма V. Если же взять $\varphi^{(m)} \notin \Phi$, то такое управление может увеличить высоту ПФ алгоритма.

При описании функционирования памяти ПЭ($\widetilde{v}$), $\widetilde{v} \in V^\pi$, после введения управляющих меток следует учесть, что каждый ПЭ($\widetilde{v}$) дополняется входом $I^{(m)}_{\pi(\varphi^{(m)})}$, выходом $O^{(m)}_{\pi(\varphi^{(m)})}$ и блоком ячеек локальной памяти $R^{(m)}_\alpha(t),\ 1 \leq \alpha \leq h_{\varphi^{(m)}}$, которые соответствуют управляющей переменной m. Описание функционирования памяти ПЭ($\widetilde{v}$) дополняется следующими равенствами:

$$\begin{aligned} &R^{(m)}_1(t) = I^{(m)}_{\pi(\varphi^{(m)})}; \\ &R^{(m)}_\alpha(t + \alpha - 1) = R^{(m)}_{\alpha-1}(t + \alpha - 2),\ 2 \leq \alpha \leq h_{\varphi^{(m)}}; \\ &O^{(m)}_{\pi(\varphi^{(m)})} = R^{(m)}_{h_{\varphi^{(m)}}}(t + h_{\varphi^{(m)}} - 1), t = t(\widetilde{v}). \end{aligned}$$

При поступлении в ПЭ($\widetilde{v}$) управляющей метки $m_\lambda,\ 1 \leq \lambda \leq \Lambda$, ПЭ($\widetilde{v}$) должен выполнить макрооперацию $F^{(k)}_\lambda,\ 1 \leq k \leq K$. Таким образом, правило вычисления переменных $x^{(k)},\ 1 \leq k \leq K$, определяется значением управляющей метки, поступившей на вход $I^{(m)}_{\pi(\varphi^{(m)})}$. Поэтому ПЭ($\widetilde{v}$) дополняется устройством управления, которое анализирует значения управляющих меток, поступивших на вход $I^{(m)}_{\pi(\varphi^{(m)})}$, и определяет

операции, которые необходимо выполнить в ПЭ($\widetilde{v}$) в данный момент времени в зависимости от этих значений.

Введение управления увеличивает число входов-выходов процессорных элементов, порождает необходимость введения устройства управления, анализирующего поступающие управляющие метки, и тем самым усложняет процессорные элементы вычислительной системы.

Пример проектирования плоского спецпроцессора для перемножения двух квадратных матриц

Спроектируем другой плоский спецпроцессор для алгоритма (1.9) перемножения двух матриц. Выберем линейный оператор отображения $\pi\colon \mathbf{Z}^3 \to \mathbf{Z}^2$, задаваемый матрицей $\Pi = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix}$.

При таком отображении вывод результата (элементов матрицы X^3) будет производиться во внутренних точках полученного спецпроцессора: точки $\pi(v_{out}) = \pi(i_1, i_2, N) = (i_1, i_2)$ при $2 \le i_1, i_2 \le N-1$ являются внутренними точками области $V^\pi = \{\, (x, y) \in \mathbf{Z}^2 \mid 1 \le x, y \le N \,\}$.

Изменим алгоритм (1.9) таким образом, чтобы вывод результата производился в вершинах БВГ, которые отображаются оператором π в граничные ПЭ получаемой структуры. Для этого введем дополнительную переменную $\widehat{x}^{(3)}$, с помощью которой будем осуществлять транспортировку элементов матрицы $X^{(3)}$ от вершин, принадлежащих плоскости $i_3 = N$, к вершинам плоскости $i_1 = N$. Транспортировку будем осуществлять по магистралям, определяемым вектором $\widehat{\varphi}^{(3)} = e_1 - e_3$.

Получим следующий алгоритм перемножения матриц:

$$
\begin{aligned}
\widehat{x}^{(3)}(v) &= \begin{cases} x^{(2)}(v - e_2)x^{(1)}(v - e_1) + x^{(3)}(v - e_3), & v \in V_3, \\ \widehat{x}^{(3)}(v - e_1 + e_3), & v \in V_1 \cup V_2, \end{cases} \\
x^{(3)}(v) &= \begin{cases} x^{(2)}(v - e_2)x^{(1)}(v - e_1), & v \in V_1, \\ x^{(2)}(v - e_2)x^{(1)}(v - e_1) + x^{(3)}(v - e_3), & v \in V_2, \end{cases} \\
x^{(1)}(v) &= x^{(1)}(v - e_1), \ v \in V_1 \cup V_2 \cup V_3, \\
x^{(2)}(v) &= x^{(2)}(v - e_2), \ v \in V_1 \cup V_2 \cup V_3,
\end{aligned}
\tag{2.17}
$$

где $V_1 = \{\, (i_1, i_2, i_3) \in \mathbf{Z}^3 \mid 1 \le i_1, i_2 \le N, \ i_3 = 1 \,\}$;

$V_2 = \{\,(i_1, i_2, i_3) \in \mathbf{Z}^3 \mid 1 \le i_1, i_2 \le N,\ 2 \le i_3 \le N-1\,\}$;
$V_3 = \{\,(i_1, i_2, i_3) \in \mathbf{Z}^3 \mid 1 \le i_1, i_2 \le N,\ i_3 = N\,\}$.

Граф зависимостей алгоритма (2.17) при $N = 2$ изображен на рис. 2.3.

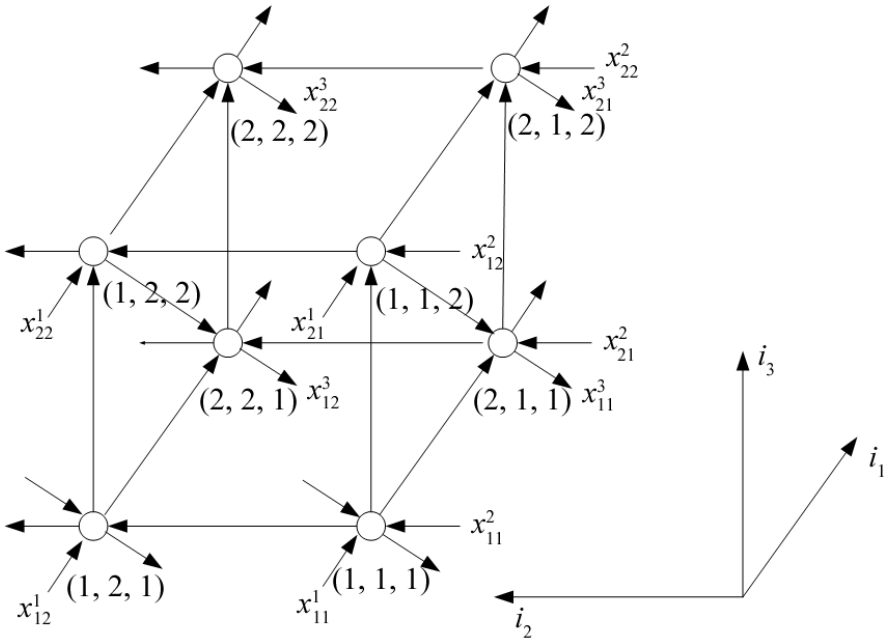


Рис. 2.3. Граф зависимостей алгоритма (2.17) при $N = 2$

Область вычислений алгоритма (2.17) совпадает с областью вычислений алгоритма (1.9) $V = V_1 \cup V_2 \cup V_3 = \{\,(i_1, i_2, i_3) \in \mathbf{Z}^3 \mid 1 \le \le i_1, i_2, i_3 \le N\,\}$. Множество векторов зависимостей алгоритма (2.17) $\Phi = \{\,e_1, e_2, e_3, e_1 - e_3\,\}$.

Определим вершины вывода результата $x^3_{i_1 i_2}$ для алгоритма (2.17). Согласно алгоритму (2.17), от вершин $v_{out} = (i_1, i_2, N)$ данные транспортируются по направлению вектора $e_1 - e_3$ к точкам плоскости $i_1 = = N$. Следовательно, $v_{out} + \lambda(e_1 - e_3) = (i_1, i_2, N) + \lambda(1, 0, -1) = (i_1 + + \lambda, i_2, N - \lambda) = \widehat{v}_{out}$, где $i_1 + \lambda = N$, т. е. $\lambda = N - i_1$. Таким образом, получаем, что вершины вывода результата $\widehat{v}_{out} = (N, i_2, i_1)$.

Конус допустимых направлений алгоритма (2.17) отличен от конуса допустимых направлений алгоритма (1.9): $K(G) = \{\,\tau \in \mathbf{Z}^3 \mid \tau_2 > > 0,\ \tau_3 > 0,\ \tau_1 - \tau_3 > 0\,\}$. В качестве таймирующей функции, соответствующей максимальной параллельной форме алгоритма (2.17), возьмем функцию $t(v) = 2i_1 + i_2 + i_3 - 3,\ v \in V$. Высота параллель-

ной формы в этом случае равна $4N-3$. Задержки в вершинах графа алгоритма:

$$h_{\varphi^{(1)}} = h_{e_1} = 2,\ h_{\varphi^{(2)}} = h_{e_2} = 1,\ h_{\varphi^{(3)}} = h_{e_3} = 1,\ h_{\widehat{\varphi}^{(3)}} = h_{e_1-e_3} = 1.$$

Множество задержек $H = \{\, 2, 1, 1, 1 \,\}$.

Множество значений таймирующей функции $T = \{\, 2i_1 + i_2 + i_3 - 3,\ 1 \le i_1, i_2, i_3 \le N \,\}$ заполняет отрезок $[1, 4N-3]$ без пропусков.

Таким образом, мы получили БВГ алгоритма (2.17). Опишем структуру ПЭ виртуальной вычислительной системы, соответствующей этому БВГ.

Каждый $\text{ПЭ}_1(v),\ v \in V_1$, имеет входы $I^{(1)}_{e_1}$, $I^{(2)}_{e_2}$, $\widehat{I}^{(3)}_{e_1-e_3}$, выходы $O^{(1)}_{e_1}$, $O^{(2)}_{e_2}$, $O^{(3)}_{e_3}$, $\widehat{O}^{(3)}_{e_1-e_3}$ и устройство, позволяющее выполнять операцию «умножение».

Каждый $\text{ПЭ}_2(v),\ v \in V_2$, имеет входы $I^{(1)}_{e_1}$, $I^{(2)}_{e_2}$, $I^{(3)}_{e_3}$, $\widehat{I}^{(3)}_{e_1-e_3}$, выходы $O^{(1)}_{e_1}$, $O^{(2)}_{e_2}$, $O^{(3)}_{e_3}$, $\widehat{O}^{(3)}_{e_1-e_3}$ и устройство, позволяющее выполнять операцию «умножение со сложением».

Каждый $\text{ПЭ}_3(v),\ v \in V_3$, имеет входы $I^{(1)}_{e_1}$, $I^{(2)}_{e_2}$, $I^{(3)}_{e_3}$, выходы $O^{(1)}_{e_1}$, $O^{(2)}_{e_2}$, $\widehat{O}^{(3)}_{e_1-e_3}$ и устройство, позволяющее выполнять операцию «умножение со сложением».

Каждый $\text{ПЭ}_i(v),\ v \in V_i,\ i = 1, 2$, имеет соответственно три ячейки локальной памяти для хранения значений переменных $x^{(2)}$, $x^{(3)}$, $\widehat{x}^{(3)}$: $R^{(2)}_1(t)$, $R^{(3)}_1(t)$, $\widehat{R}^{(3)}_1(t)$ и один блок ячеек локальной памяти длиной 2 для хранения значений переменной $x^{(1)}$: $\{\, R^{(1)}_{\alpha}(t),\ 1 \le \alpha \le 2 \,\}$.

Каждый $\text{ПЭ}_3(v),\ v \in V_3$, имеет две ячейки локальной памяти для хранения значений переменных $x^{(2)}$, $\widehat{x}^{(3)}$: $R^{(2)}_1(t)$, $\widehat{R}^{(3)}_1(t)$ соответственно и один блок ячеек локальной памяти длиной 2 для хранения значений переменной $x^{(1)}$: $\{\, R^{(1)}_{\alpha}(t),\ 1 \le \alpha \le 2 \,\}$.

Описание функционирования локальной памяти $\text{ПЭ}(v),\ v \in V$, имеет вид:

$$\begin{aligned} P_1(v)\colon \{\, & \widehat{R}^{(3)}_1(t) = \widehat{I}^{(3)}_{e_1-e_3},\ \widehat{O}^{(3)}_{e_1-e_3} = \widehat{R}^{(3)}_1(t), \\ & R^{(3)}_1(t) = I^{(1)}_{e_1} \cdot I^{(2)}_{e_2},\ O^{(3)}_{e_3} = R^{(3)}_1(t), \\ & R^{(1)}_1(t) = I^{(1)}_{e_1},\ R^{(1)}_2(t+1) = R^{(1)}_1(t),\ O^{(1)}_{e_1} = R^{(1)}_2(t+1), \\ & R^{(2)}_1(t) = I^{(2)}_{e_2},\ O^{(2)}_{e_2} = R^{(2)}_1(t) \,\},\ v \in V_1; \end{aligned}$$

$$
\begin{aligned}
P_2(v)\colon \{\, & \widehat{R}_1^{(3)}(t) = \widehat{I}_{e_1-e_3}^{(3)},\ \widehat{O}_{e_1-e_3}^{(3)} = \widehat{R}_1^{(3)}(t), \\
& R_1^{(3)}(t) = I_{e_1}^{(1)} \cdot I_{e_2}^{(2)} + I_{e_3}^{(3)},\ O_{e_3}^{(3)} = R_1^{(3)}(t), \\
& R_1^{(1)}(t) = I_{e_1}^{(1)},\ R_2^{(1)}(t+1) = R_1^{(1)}(t),\ O_{e_1}^{(1)} = R_2^{(1)}(t+1), \\
& R_1^{(2)}(t) = I_{e_2}^{(2)},\ O_{e_2}^{(2)} = R_1^{(2)}(t)\,\},\ v \in V_2;
\end{aligned}
$$

$$
\begin{aligned}
P_3(v)\colon \{\, & \widehat{R}_1^{(3)}(t) = I_{e_1}^{(1)} \cdot I_{e_2}^{(2)} + I_{e_3}^{(3)},\ \widehat{O}_{e_1-e_3}^{(3)} = \widehat{R}_1^{(3)}(t), \\
& R_1^{(1)}(t) = I_{e_1}^{(1)},\ R_2^{(1)}(t+1) = R_1^{(1)}(t),\ O_{e_1}^{(1)} = R_2^{(1)}(t+1), \\
& R_1^{(2)}(t) = I_{e_2}^{(2)},\ O_{e_2}^{(2)} = R_1^{(2)}(t)\,\},\ v \in V_3.
\end{aligned}
$$

Проверим выполнение условия совместимости таймирующей функции и функции размещения (2.9). В нашем случае для таких s, что $\pi(s) = 0,\ s \neq 0$, т. е. для $s = (s_1, s_2, s_3),\ s_1, s_2 = 0,\ s_3 \neq 0$, условие $\tau \cdot s \neq 0$, т. е. $2s_1 + s_2 + s_3 \neq 0$, выполняется.

Образы областей вычислений при отображении π: $V^\pi = V_1^\pi = V_2^\pi = V_3^\pi = \{\, \widetilde{v} = (x, y) \in \mathbf{Z}^2 \mid 1 \leq x, y \leq N \,\}$. Поскольку $V_1^\pi \cap V_2^\pi = V_1^\pi \cap V_3^\pi = V_2^\pi \cap V_3^\pi = \{\, \widetilde{v} = (x, y) \in \mathbf{Z}^2 \mid 1 \leq x, y \leq N \,\} \neq \emptyset$, т. е. выполняется условие (2.14) (прообразами каждой вершины ВГ $\widetilde{v} \in V^\pi$ являются вершины БВГ $v \in V$ трех различных типов), то необходимо осуществить разметку вершин БВГ разных типов управляющими метками $m_1,\ m_2$ и m_3 соответственно. Для их транспортировки введем управляющую переменную $m(v)$ и запишем алгоритм транспортировки вида (2.15):

$$
\begin{aligned}
& m(v) = m(v - e_2),\ v \in V; \\
& m(v - e_2) = m_1,\ \ v \in \{\, (i_1, i_2, i_3) \in \mathbf{Z}^3 \mid 1 \leq i_1 \leq N,\ i_2 = i_3 = 1,\, \}; \\
& m(v - e_2) = m_2,\ \ v \in \{\, (i_1, i_2, i_3) \in \mathbf{Z}^3 \mid 1 \leq i_1 \leq N,\ i_2 = 1, \\
& \qquad\qquad\qquad\qquad\qquad\qquad 2 \leq i_3 \leq N - 1\,\}; \\
& m(v - e_2) = m_3,\ \ v \in \{\, (i_1, i_2, i_3) \in \mathbf{Z}^3 \mid 1 \leq i_1 \leq N,\ i_2 = 1,\ i_3 = N\,\}.
\end{aligned}
$$

Вектор $\varphi^{(m)} = e_2$ не попадает в ядро оператора отображения: $\pi(\varphi^{(m)}) = (0, 1) \neq (0, 0)$. Схема подачи управляющих меток в БВГ алгоритма (2.17) при $N = 3$ изображена на рис. 2.4.

Поскольку $\varphi^{(m)} \in \Phi$, то введенное управление не увеличивает высоту ПФ алгоритма. Задержка по дугам, определяемым вектором $\varphi^{(m)}$: $h_{\varphi^{(m)}} = h_{e_2} = 1$.

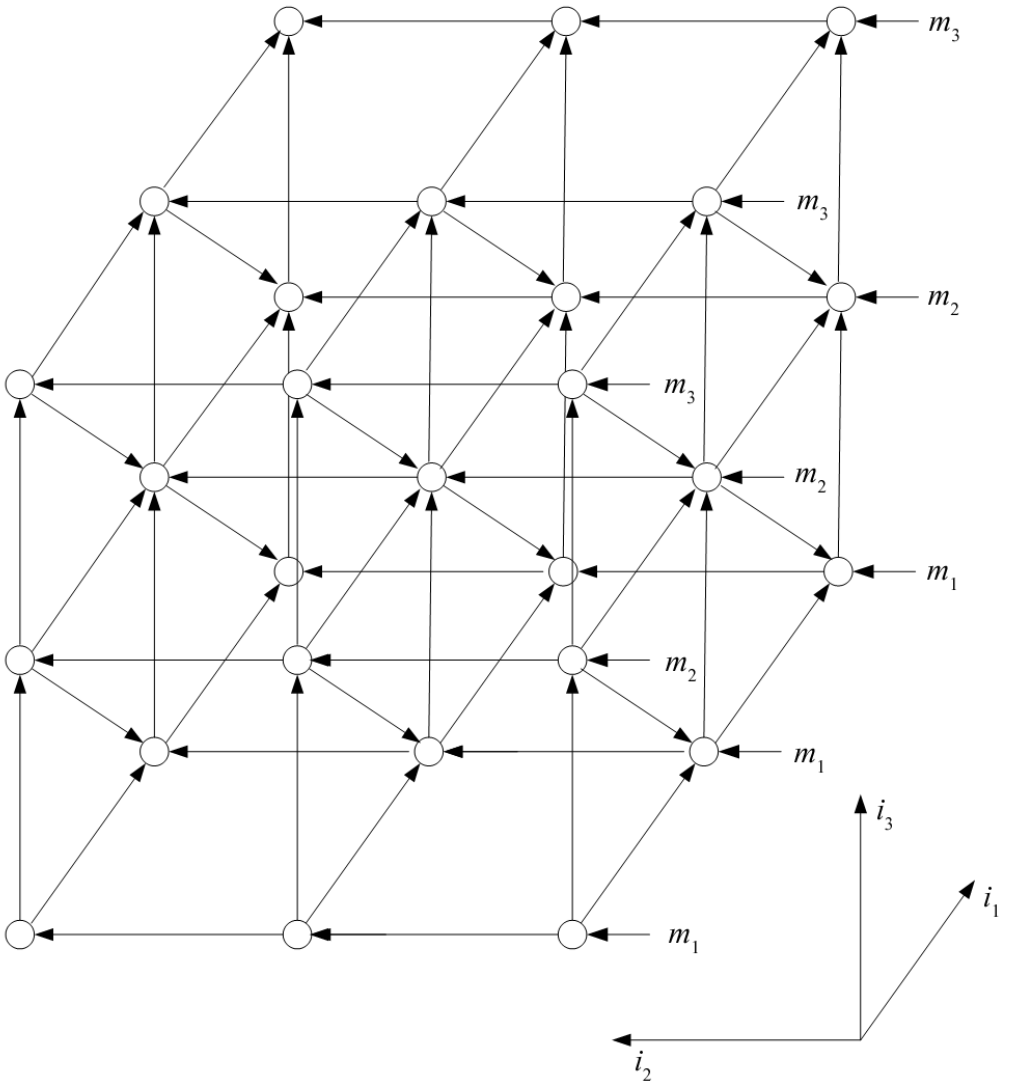


Рис. 2.4. Схема подачи управляющих меток в БВГ алгоритма (2.17) при $N = 3$

Образы векторов зависимостей алгоритма (2.17) при отображении π: $\pi(\varphi^{(1)}) = (1, 0)$, $\pi(\varphi^{(2)}) = \pi(\varphi^{(m)}) = (0, 1)$, $\pi(\varphi^{(3)}) = (0, 0)$, $\pi(\widehat{\varphi}^{(3)}) = = (1, 0)$. Отсюда видно, что оператор π не порождает связей, зависящих от размера задачи.

Многозначная таймирующая функция

$$t^{\pi}(\widetilde{v}) = t^{\pi}(x, y) = \{\, 2i_1 + i_2 + i_3 - 3, \mid i_1 = x,$$
$$i_2 = y,\ (i_1, i_2, i_3) \in V \,\},\ \widetilde{v} \in V^{\pi}.$$

Опишем функционирование локальной памяти процессорного элемента $\text{ПЭ}(\widetilde{v})$ спроектированного спецпроцессора.

Локальная память состоит из пяти блоков: три блока по одной ячейке памяти и один блок содержит две ячейки памяти.

Каждый процессорный элемент имеет четыре входа и четыре выхода.

$$
\begin{aligned}
P(\widetilde{v})\colon \{\, & R_1^{(m)}(t) = I_{(0,1)}^{(m)},\ O_{(0,1)}^{(m)} = R_1^{(m)}(t),\\
& \widehat{R}_1^{(3)}(t) = \begin{cases} \widehat{I}_{(1,0)}^{(3)},\ \text{если } I_{(0,1)}^{(m)} = m_1 \vee m_2,\\ I_{(1,0)}^{(1)} \cdot I_{(0,1)}^{(2)} + R_1^{(3)}(t-1), \text{если } I_{(0,1)}^{(m)} = m_3, \end{cases}\\
& \widehat{O}_{(1,0)}^{(3)} = \widehat{R}_1^{(3)}(t),\\
& R_1^{(3)}(t) = \begin{cases} I_{(1,0)}^{(1)} \cdot I_{(0,1)}^{(2)},\ \text{если } I_{(0,1)}^{(m)} = m_1,\\ I_{(1,0)}^{(1)} \cdot I_{(0,1)}^{(2)} + R_1^{(3)}(t-1), \text{если } I_{(0,1)}^{(m)} = m_2, \end{cases}\\
& R_1^{(1)}(t) = I_{(1,0)}^{(1)},\ R_2^{(1)}(t+1) = R_1^{(1)}(t),\ O_{(1,0)}^{(1)} = R_2^{(1)}(t+1),\\
& R_1^{(2)}(t) = I_{(0,1)}^{(2)},\ O_{(0,1)}^{(2)} = R_1^{(2)}(t)\,\},
\end{aligned}
$$

$\widetilde{v} \in V^{\pi}$.

Режим ввода начальных данных в спецпроцессор и вывода результатов приведен в табл. 2.3.

Таблица 2.3

Режим ввода начальных данных и вывода результатов плоского спецпроцессора для перемножения двух квадратных матриц

Данные	v_{in}	ПЭ($\pi(v_{in})$)	Порт	$t(v_{in})$	Изменение параметров
m_1	$(i_1, 1, 1)$	ПЭ$(i_1, 1)$	$I_{(0,1)}^{(m)}$	$2i_1 - 1$	$1 \le i_1 \le N$
m_2	$(i_1, 1, i_3)$	ПЭ$(i_1, 1)$	$I_{(0,1)}^{(m)}$	$2i_1 + i_3 - 2$	$1 \le i_1 \le N$ $2 \le i_3 \le N-1$
m_3	$(i_1, 1, N)$	ПЭ$(i_1, 1)$	$I_{(0,1)}^{(m)}$	$2i_1 + N - 2$	$1 \le i_1 \le N$
$x_{i_3 i_2}^1$	$(1, i_2, i_3)$	ПЭ$(1, i_2)$	$I_{(1,0)}^{(1)}$	$i_2 + i_3 - 1$	$1 \le i_2, i_3 \le N$
$x_{i_1 i_3}^2$	$(i_1, 1, i_3)$	ПЭ$(i_1, 1)$	$I_{(0,1)}^{(2)}$	$2i_1 + i_3 - 2$	$1 \le i_1, i_3 \le N$
Результат	v_{out}	ПЭ($\pi(v_{out})$)	Порт	$t(v_{out})$	Изменение параметров
$x_{i_1 i_2}^3$	(N, i_2, i_1)	ПЭ(N, i_2)	$\widehat{O}_{(1,0)}^{(3)}$	$i_1 + i_2 + 2N - 3$	$1 \le i_1, i_2 \le N$

Структурная схема спроектированного матричного спецпроцессора при $N = 3$ изображена на рис. 2.5.

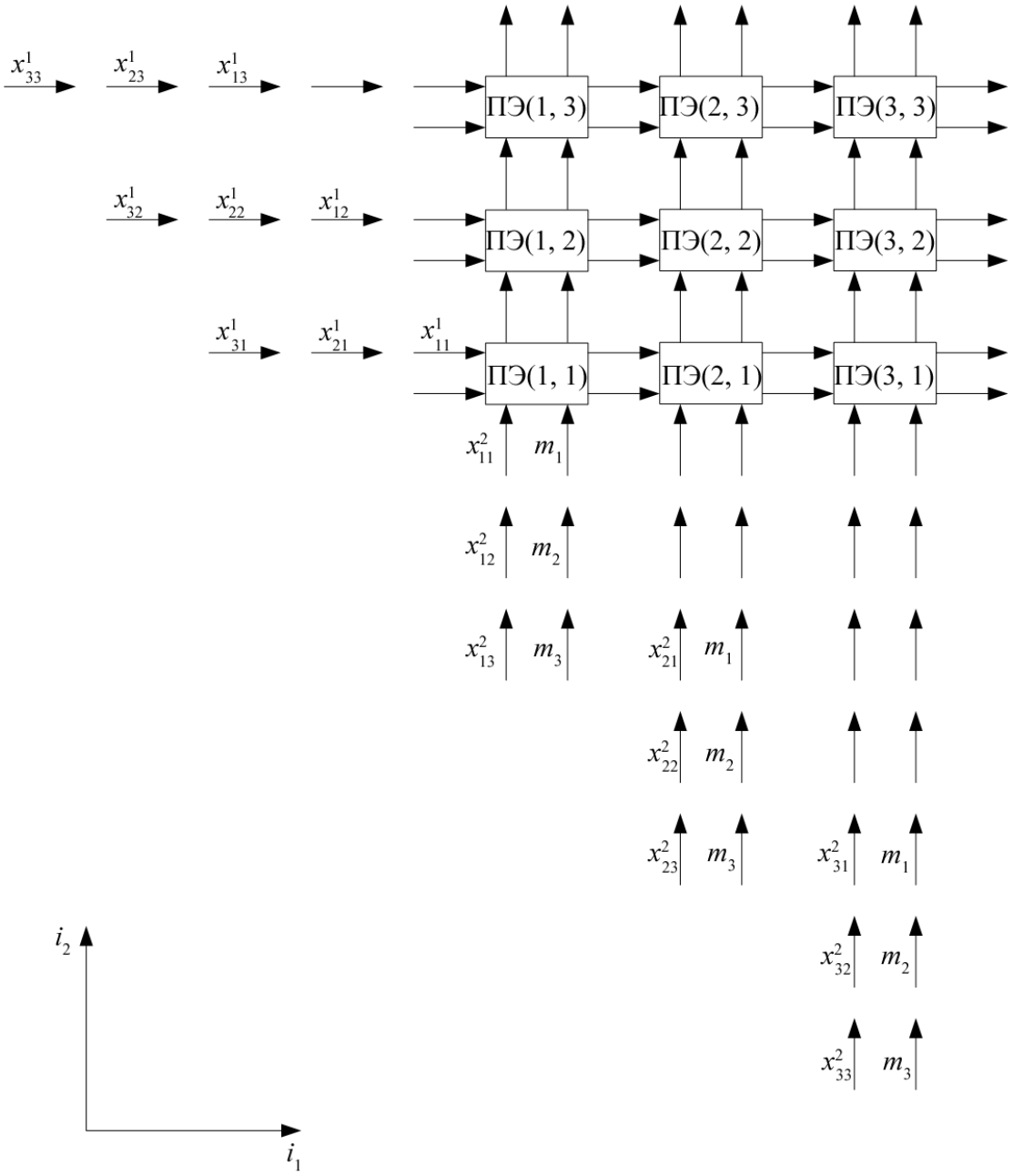


Рис. 2.5. Структурная схема матричного спецпроцессора при $N = 3$