

Статья. Библиографические данные:

Житников В.П., Шерыхалина Н.М., Житников А.П.
Программно-методический комплекс "Параллельные алгоритмы и программы".
/ Сб. Высокопроизводительные вычислительные ресурсы России:
состояние и перспективы развития.
– Уфа: УГАТУ, 2003. – С. 151-161.

ПРОГРАММНО-МЕТОДИЧЕСКИЙ КОМПЛЕКС "ПАРАЛЛЕЛЬНЫЕ АЛГОРИТМЫ И ПРОГРАММЫ"

Представлен электронный вариант оригинала печатной статьи.
Статья приводится с технической доработкой:
гиперссылки, оглавление, цветные элементы и т.п.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ.....	3
1 ИСХОДНЫЕ ПОЛОЖЕНИЯ.....	4
Общая теоретическая основа	4
Концепция операционных алгоритмических полюсников	4
Первая очередь программно-методического комплекса	5
2 СТРУКТУРНЫЕ ФОРМУЛЫ И СХЕМЫ АЛГОРИТМОВ. БАЗОВЫЙ СТРУКТУРНЫЙ КЛАСС.....	6
Ациклические алгоритмические двухполюсники (без контуров)	6
БИ: База индукции.....	6
ШИ: Шаг индукции.....	6
ЗИ: Заключение индукции.....	8
3 ТЕХНИКА СТРУКТУРНЫХ ПОСТРОЕНИЙ.....	10
3.1 Общие положения	10
3.2 Пример 1: Простой параллельный алгоритм. Двухполюсная структура	10
Исходные данные	10
Поток данных алгоритма	11
Поток управления алгоритма	11
Исполнение алгоритма.....	12
Примечание 4.1. Ручная доработка временной диаграммы	13
Длительность цикла исполнения алгоритма.....	13
4 ПСЕВДОКОДЫ АЛГОРИТМОВ	14
Вербальные тексты алгоритмов (структуры текстов в служебной лексике).	14
5 РАСШИРЕНИЯ БАЗОВОГО СТРУКТУРНОГО КЛАССА.....	16
Приведенные двухполюсники.....	16

Пример 2. Многополюсная структура. Дизъюнктивная сборка	16
Примечание 6.1. Связь структурных формул с логическими схемами алгоритмов (ЛСА)	16
ЗАКЛЮЧЕНИЕ	17
Вторая очередь программно-методического комплекса	17

ВВЕДЕНИЕ

На кафедре проектирования систем информатики УГАТУ выполняются поэтапные (очередями) опытно-поисковые разработки и сопутствующее опытное внедрение в учебный процесс *программно-методического комплекса (ПМК) параллельной алгоритмизации и параллельного программирования*.

Разработка ПМК ориентирована на комплексное, в целом, отражение *методов и средств реализации* параллельных алгоритмов: *аппаратная* реализация (в частности, опорная модельная реализация), *программная* реализация (включая объектно-ориентированные CASE-технологии) и *организационная* реализация алгоритмов в составе автоматических и автоматизированных систем общего и производственного назначения – техника, технология, организация и управление производством (материальной и информационной) продукции и услуг.

Принимается ориентация на следующую общую систему логико-алгоритмической подготовки:

- *сравнительный анализ структурной проблематики*, организуемый по уровням: малый, средний, большой и сверхбольшой (потенциальный) параллелизм процессов;
- *многопроходное развертывание содержания* проблематики (по спирали): предшествующие уровни анализа включаются как составные в более высокие уровни и представляют более простые, но более общие опорные средства.

1 ИСХОДНЫЕ ПОЛОЖЕНИЯ

Общая теоретическая основа

Теоретической основой выполняемых разработок ПМК являются:

а) **Структурная теория параллельных алгоритмов** – методы и средства структурного описания, анализа, синтеза, моделирования и преобразований параллельных алгоритмов (и последовательных алгоритмов, как базовый частный случай): двухполюсные и многополюсные, ациклические и циклические структуры, конвейерные режимы и структуры, компактные свертки описаний однородных структур, постоянные (статические) и переменные – переключательные (альтернативные) и порождаемые структуры.

б) Интегрированный **полиморфный язык** структурного и структурно-функционального описания параллельных алгоритмов – синтаксис и семантика (по классам алгоритмов). Параллельно излагаются согласованные взаимно-обратимые (знаковые) формы описания разного назначения – обобщенные одномерные и двумерные тексты алгоритмов:

1) **Литерные** тексты алгоритмов:

- **структурные и функциональные (структурно-функциональные) формулы** – инфиксная, префиксная, постфиксная и разные комбинированные синтаксические формы;
- **псевдокоды алгоритмов** – программно-ориентированные вербальные (словесные) тексты алгоритмов в разных синтаксических классах и лексических системах: алгол-подобные, паскаль-подобные, си-подобные, оккам-подобные тексты и т.п. (на основе синтаксических преобразований структурных формул как шаблонов вербальных текстов алгоритмов).

2) **Графические** тексты алгоритмов:

- **структурные схемы** – **блок-схемы** в горизонтальном (аппаратно-ориентированном) и вертикальном (программно-ориентированном) исполнении, компактные **штрих-схемы**, а также **граф-схемы**;
- линейные и сетевые **временные диаграммы** (графические протоколы) исполнения алгоритмов, их **сетевые расчеты** (по аналогии с методами сетевого планирования и управления на основе сетевых графиков комплексов работ).

3) **Табличные** средства описания алгоритмов (пока ограниченно).

в) **Функциональное обоснование** структурных методов описания и алгебры структурных преобразований параллельных алгоритмов:

временные булевы функции структур (поточков) управления – операторная форма временных булевых функций и ее интерпретация **в метрической и топологической логике времени (темпоральная логика)**.

Концепция операционных алгоритмических полюсников

Структурно-функциональные средства разделяются на два уровня анализа (Рис. 1.1):

- **неявные полюсники** (обобщенные операторы без наличия явных переменных потока управления) – исходная основа методов описания структур управления;
- **явные полюсники** (операторы с переменными цепей передачи управления) – отражение функциональной семантики структур управления.

Алгоритм $O' = (A' :: B')$ или команда $E' = (Z' :: U')$ – это операционное предписание операционному исполнителю O'' , E'' выполнить комплекс операций O''' , E''' (дискретный процесс) в определенном (последовательном или параллельном) порядке в соответствии с определенными функциями:

$$O = (A :: B) = ((A: Y_a = F_a(X_a)) :: (B: Y_b = F_b(X_b))) ;$$

$$E = (Z :: U) = ((Z: Y_z = F_z(X_z)) :: (U: Y_u = F_u(X_u))) ;$$

A / Z – субъект управления: управляющая структура потока управления ($У$);

B / U – объект управления: управляемая структура потока данных ($Д$);

$X = (X_1, X_2, \dots)$, $Y = (Y_1, Y_2, \dots)$ – объекты переменных (полюсы) потоков;

F – функциональные объекты: функциональные преобразователи потока управления и потока данных.

Исходные условия постановки задачи структурного описания – рабочие функции потока данных (без учета функций потока управления):

$$O = (A :: Y_b = F_b(X_b)); \quad E = (Z :: Y_u = F_u(X_u)).$$

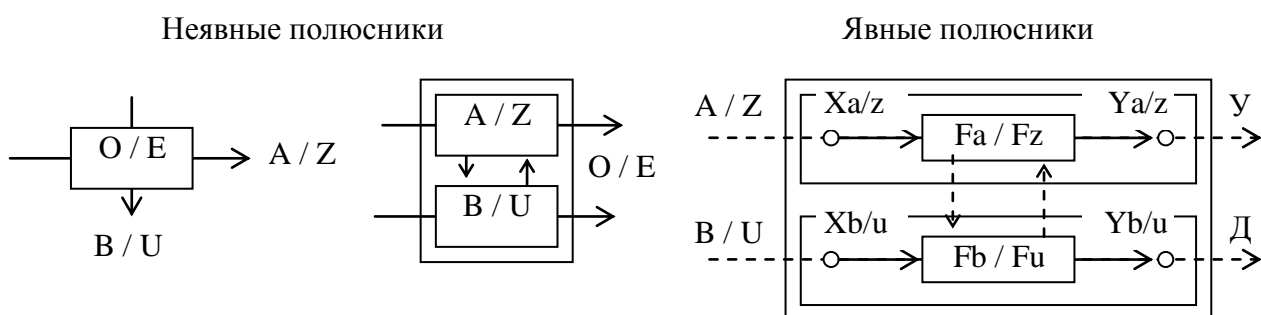


Рис. 1.1. Поток управления и поток данных алгоритма

Первая очередь программно-методического комплекса

Первая очередь ПМК включает опытно-поисковые разработки комплекса программой поддержки по следующим аспектам:

- **первичное теоретическое обучение** в области **структурной теории** параллельных алгоритмов – структурное описание и анализ ациклических двухполюсных структур алгоритмов (до изложения алгебры их структурных преобразований);

- **практика** по **технике структурных построений** на основе комплектов примеров разной сложности и ключевой концептуальной значимости:

практика проходит в опережающем порядке, достаточно результативно и успешно, что стимулирует усиление внимания к освоению теории.

Основной состав комплексов первой очереди включает:

а) Три программы (поисковые макетные разработки) – **автоматизация** трудоемких **алгоритмических построений** по заданной структурной формуле алгоритма:

- структурные схемы и вербальные тексты параллельных алгоритмов разных визуальных и синтаксических форм представления;

- временные диаграммы исполнения алгоритмов – статические и динамические диаграммы (на основе имитационного моделирования).

б) **Графический тренажер-тестер** – автоматизация тренировки и контроля построений структурных схем параллельных алгоритмов (первые опыты).

2 СТРУКТУРНЫЕ ФОРМУЛЫ И СХЕМЫ АЛГОРИТМОВ. БАЗОВЫЙ СТРУКТУРНЫЙ КЛАСС

Ациклические алгоритмические двухполюсники (без контуров)

Формируется ключевая структурная основа построения полюсников – ациклические двухполюсники. Используется упрощенное краткое описание синтаксиса методом индукции (умеренная формализация) и обеспечивается подготовка описания синтаксиса на основе формальных грамматик (под разные конкретные задачи формализации и автоматизации структурных построений):

БИ: База индукции

Определяются первичные конструктивные элементы, интерпретируемые как элементарные (вырожденные) структуры, на основе которых затем определяются составные структуры:

$A_i = ()$ – пустой алгоритм (типа: begin end). Интерпретации пустого полюсника:

$() = ()' = \rightarrow(\rightarrow)\rightarrow$ – постоянно замкнутая перемычка;

$() = ()'' = \rightarrow(\rightarrow|\rightarrow)\rightarrow$ – постоянно разомкнутая перемычка;

$A_i = (Z_k) = \rightarrow(\rightarrow Z_k \rightarrow)\rightarrow$ – единичный (однокомандный) алгоритм.

ШИ: Шаг индукции

Определяются производные структуры полюсников:

ШИ0: Вспомогательные многополюсные представления. Могут представлять самостоятельный практический интерес:

ШИ01: Пара двухполюсников – простая операция группировки (парный **комплект двухполюсников** – в более поздней терминологии, Рис. 2.1):

$A_i = (A_{i1}, A_{i2})$

Уточнение структурной семантики (двухмерная структурная формула):

$A_i = (A_{i1}, A_{i2}) = (A_{i1} \downarrow A_{i2}) = \quad // \downarrow$ - внутрискобочный перевод строки
 $= (A_{i1}) = \rightarrow(\rightarrow A_{i1} \rightarrow)\rightarrow$
 $(A_{i2}) \quad \rightarrow(\rightarrow A_{i2} \rightarrow)\rightarrow$

ШИ02: Разветвление пары двухполюсников (**дивергенция** или **разделение потоков** – в более поздней терминологии, Рис. 2.1):

$A_i = \#(A_{i1}, A_{i2}) = (A_{i1} \# A_{i2}) = (A_{i1}, A_{i2}) \#$

Уточнение структурной семантики (двухмерная структурная формула):

$A_i = (A_{i1} \# A_{i2}) = \#(A_{i1}, A_{i2}) = \#(A_{i1}) = \rightarrow\#|\rightarrow(A_{i1})\rightarrow$
 $(A_{i2}) \quad |\rightarrow(A_{i2})\rightarrow$

где $\#$ – вилка (fork – узел вилки): множественный повторитель.

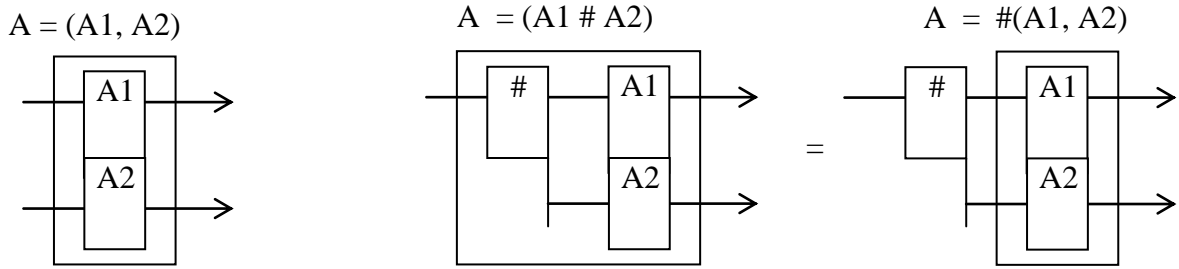


Рис. 2.1. Комплект и дивергенция операторов

ШИ03: Объединение пары двухполюсников (**конвергенция** или **соединение** потоков – в более поздней терминологии, Рис. 2.2):

$$A_i = \circ (A_{i1}, A_{i2}) = (A_{i1} \circ A_{i2}) = (A_{i1}, A_{i2}) \circ$$

Уточнение структурной семантики (двухмерная формула):

$$A_i = (A_{i1} \circ A_{i2}) = (A_{i1}, A_{i2}) \circ = \begin{matrix} (A_{i1}) \circ & = & \rightarrow (A_{i1}) \rightarrow | \circ \rightarrow \\ (A_{i2}) & & \rightarrow (A_{i2}) \rightarrow | \end{matrix}$$

где $\circ \in \{\&, \vee, \dots\}$ – сборка (join – узел сборки) по конъюнкции & (И), дизъюнкции \vee (Или) или, возможно, по другим булевым функциям.

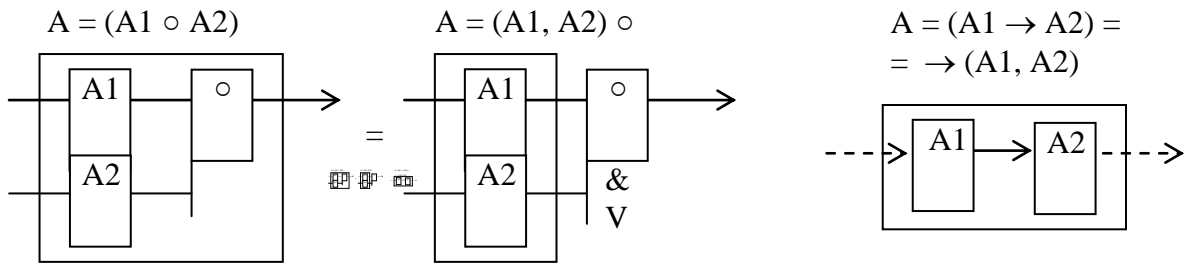


Рис. 2.2. Конвергенция (соединение) и секвенция операторов (потоков)

ШИ1: Последовательное соединение операторов (seq: sequential или **секвенция** – в более поздней терминологии, Рис. 2.2):

$$A_i = \rightarrow (A_{i1}, A_{i2}) = (A_{i1} \rightarrow A_{i2}) = (A_{i1}, A_{i2}) \rightarrow$$

Знак " \rightarrow " (\rightarrow) означает логическую связку типа "затем" (then), структурную операцию секвенции (последовательной связи):

$A_i = (A_{i1} \rightarrow A_{i2}) = (A_{i1} - A_{i2})$ – упрощение записи для правостороннего направления.

ШИ2: Параллельное соединение операторов (par: parallel или *параллель* – в более поздней терминологии, Рис. 2.3):

$$A_i = \#(A_{i1}, A_{i2}) \circ = (A_{i1} \# \circ A_{i2}) = \# \circ (A_{i1}, A_{i2}) = (A_{i1} \circ \# A_{i2}) = \dots$$

В более поздних обозначениях:

$$A_i = (A_{i1} \parallel A_{i2}) = (A_{i1} \# \circ A_{i2}) = \# \circ (A_{i1}, A_{i2}) = (A_{i1}, A_{i2}) \# \circ = \#(A_{i1}, A_{i2}) \circ = \dots ,$$

где $\parallel = \# \circ$ – где парная операция параллельного объединения.

Уточнение структурной семантики (двухмерная структурная формула):

$$A_i = (A_{i1} \# \circ A_{i2}) = \#(A_{i1}, A_{i2}) \circ = \#(A_{i1}) \circ = \rightarrow | \rightarrow (A_{i1}) \rightarrow | \circ \rightarrow$$

$$(A_{i2}) \quad | \rightarrow (A_{i2}) \rightarrow |$$

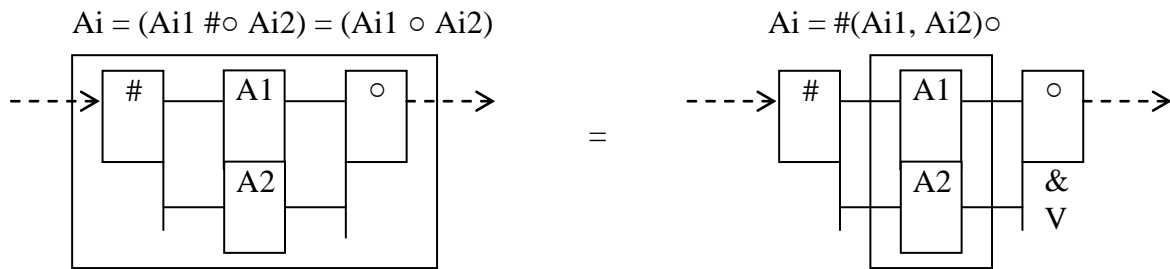


Рис. 2.3. Параллель (параллельная связь) операторов

Если не используются многополюсные схемы по ШИ0, то:

$$(A_{i1} \# \circ A_{i2}) = (A_{i1} \# \circ A_{i2}) = (A_{i1} \circ A_{i2}) \quad \text{– упрощение записи формулы.}$$

В частности, возможно упрощение обозначения парных операций $\# \&$, $\# \vee$:

$$A_i = \#(A_{i1}, A_{i2}) \& = (A_{i1} \# \& A_{i2}) = (A_{i1} \# \& A_{i2}) = (A_{i1} \& A_{i2})$$

$$A_i = \#(A_{i1}, A_{i2}) \vee = (A_{i1} \# \vee A_{i2}) = (A_{i1} \# \vee A_{i2}) = (A_{i1} \vee A_{i2})$$

ЗИ: Заключение индукции

Подводятся итоги и оговариваются расширения исходных определений:

ЗИ1: Определены правильно построенные формулы (ППФ) и схемы (ППС) объектов исходного класса – последовательное и параллельное соединение двухполюсников (без контуров и петель) с двухместными структурными операциями. Других объектов исходного класса нет.

ЗИ2: На основе исходных определений (для разных частных случаев ППФ и ППС) допускается вводить эквивалентные производные обозначения:

1) Навешивание и удаление внешних оболочек:

$$A_i = (A_i) \quad (A_i) = A_i$$

$$(Z_i) = Z_i \quad (A_i \rightarrow A_k) = A_i \rightarrow A_k \quad (A_i \circ A_k) = A_i \circ A_k$$

2) Удаление явной секвенции (неявная секвенция):

$$A_i \rightarrow A_k = A_i \text{--} A_k = A_i A_k = A_i A_k \quad \text{--};$$

3) Введение многоместных структурных операций (по левой ассоциативности):

$$(A_{i1} \rightarrow A_{i2}) \rightarrow A_{i3} = A_{i1} \rightarrow A_{i2} \rightarrow A_{i3}, \quad (A_{i1} \circ A_{i2}) \circ A_{i3} = A_{i1} \circ A_{i2} \circ A_{i3}$$

3) Продольные и поперечные свертки однородных структур по некоторому множеству индексов $S = \{ i_1, i_2, \dots, i_n \}$ (аналоги операторов типа for, for all, for any):

$$A_{i1} \rightarrow A_{i2} \rightarrow \dots \rightarrow A_{in} = L(k \in S) A_k$$

$$A_{i1} \& A_{i2} \& \dots \& A_{in} = \forall(k \in S) A_k$$

$$A_{i1} \vee A_{i2} \vee \dots \vee A_{in} = \exists(k \in S) A_k$$

3 ТЕХНИКА СТРУКТУРНЫХ ПОСТРОЕНИЙ

3.1 Общие положения

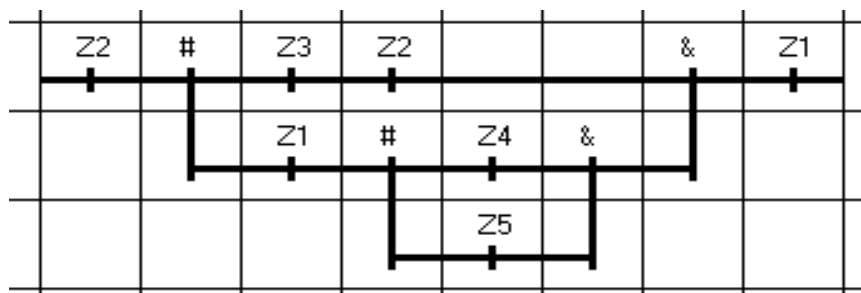
Особое внимание уделяется отработке техники сопряженных полиморфных структурных построений. Применяется тщательная проработка полиморфного синтаксиса с целью решения различных задач ручных и автоматизированных построений и их упрощения. В частности, обеспечивается однозначная взаимная обратимость записи формул, вербальных текстов (псевдокодов) и схем алгоритмов. Для двухполюсников (планарные графы) обеспечивается построение схем только горизонтальными и вертикальными линиями без пересечений и изломов связей (существенное упрощение ручных и автоматизированных построений) и т.п.

Используются два стиля построения схем – линейные и фрагментированные схемы (ячеистая структура знакового поля согласно уточнению структурной семантики определения ППФ и ППС). В частности, для графического тренажера используется способ набора схем по заданной структурной формуле из элементарных графических фрагментов (условно: набор схем из "кубиков"). При выравнивании элементов вверх и влево обеспечивается однозначное построение схем по формулам. Например:

СФА: Структурная формула алгоритма / У: Поток управления:

$$A = (Z2 - ((Z3 - Z2) \& (Z1 - (Z4 \& Z5))) - Z1) = Z_2(Z_3Z_2 \& Z_1(Z_4 \& Z_5))Z_1$$

ШСА: Штрих-схема алгоритма (упрощенная БСА: Блок-схема алгоритма):



Далее на простом примере представлены исходные аспекты техники структурных построений в заданном синтаксисе.

3.2 Пример 1: Простой параллельный алгоритм. Двухполюсная структура

Исходные данные

Д: Поток данных. Задана некоторая ОРФ: Общая (обобщенная) рабочая функция:

$$y = f(x) = f(x_1, x_2, x_3) = f_4(f_1(x_1), f_2(x_2), f_3(x_3))$$

Общая функция разлагается на частные функции f_1, f_2, f_3, f_4 .

СРФ: Система (частных) рабочих функций

$$(v_1, v_2, v_3) = f_0(x) = f_0(x_1, x_2, x_3) \quad // \text{ подготовка исходных данных}$$

$$u_1 = f_1(v_1), \quad u_2 = f_2(v_2), \quad u_3 = f_3(v_3), \quad y = f_4(u_1, u_2, u_3)$$

ООА: Общее обозначение алгоритма

$$A :: y = f(x_1, x_2, x_3)$$

СКА: Система команд алгоритма (возможно – это вызовы подпрограмм):

$$Z1 :: (v_1, v_2, v_3) = f_0(x) = f_0(x_1, x_2, x_3)$$

$$Z2 :: u1 = f1(v1) \quad Z3 :: u2 = f2(v2) \quad Z4 :: u3 = f3(v3)$$

$$Z5 :: y = f(x) = f4(u1, u2, u3)$$

Примечание. Рассматриваются прикладные интерпретации задачи:

- численная задача – наиболее доступный и общепонятный объект анализа;
- технология оперативной полиграфии – подготовка оригинал-макета:
 $f0$ – редакционно-издательская обработка авторского оригинала, $f1$ – набор текста, $f2$ – подготовка графики, $f3$ – набор сложных формул, $f4$ – общая верстка документа (кодированный оригинал-макет документа);
- управление агрегатным станком – обработка корпусной детали:
 $f0$ – загрузка детали (заготовки), $f1, f2, f3$ – трехсторонняя обработка корпуса, $f4$ – разгрузка (обработанной) детали и т.п.

Поток данных алгоритма

ГСА: Граф-схема алгоритма / Д: Поток данных (двудольный граф):

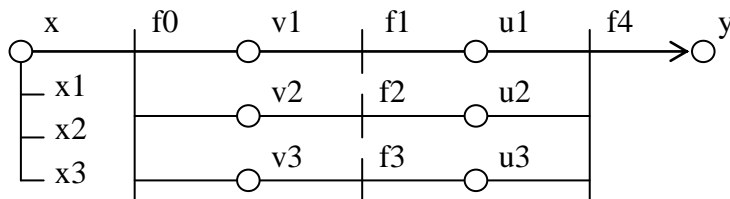


Рис. 3.1. Графическое представление потока данных алгоритма

Очевидно (Рис. 3.1) наличие параллельных потоков данных – основа потенциального (естественного) параллелизма алгоритмов. Функции $f1, f2, f3$ независимы между собой (по операндам и результатам) и соответствующие им команды $Z2, Z3, Z4$ могут выполняться в любом (последовательном или параллельном) порядке. Возможен формализованный анализ на распараллеливание задачи. На основе исходных функциональных данных определяется поток управления алгоритма:

Поток управления алгоритма

СФА: Структурная формула алгоритма / У: Поток управления

ИнФ: Инфиксная форма представления (полная и краткая). Варианты:

Последовательный алгоритм (вырожденная степень параллелизма $p = 1$)

$$A51 = (Z1 \rightarrow Z2 \rightarrow Z3 \rightarrow Z4 \rightarrow Z5) = Z1 - Z2 - Z3 - Z4 - Z5 = Z1 Z2 Z3 Z4 Z5$$

Параллельный алгоритм (максимальный параллелизм $p = p_{\max} = 3$)

$$A53 = (Z1 \rightarrow (Z2 \& Z3 \& Z4) \rightarrow Z5) = Z1 - (Z2 \& Z3 \& Z4) - Z5 = Z1 (Z2 \& Z3 \& Z4) Z5$$

На основе заданной СФА строится схема алгоритма (выводится на экран монитора, Рис. 3.2):

ССА: Структурная схема алгоритма / ГИ: Горизонтальное исполнение
Используются два вида ССА:

БСА: Блок-схема алгоритма

ШСА: Штрих-схема алгоритма

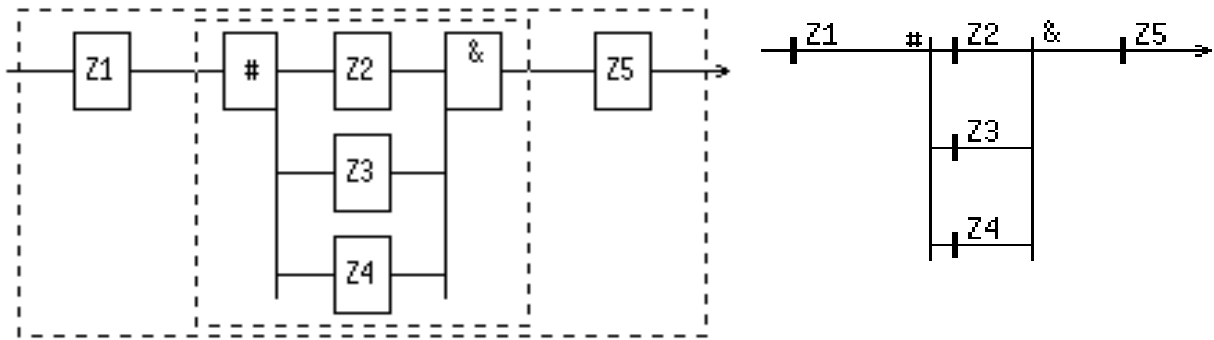


Рис. 3.2. Графическое представление потока управления алгоритма

Показаны два варианта исполнения ССА (БСА и ШСА):

- с отражением схемных оболочек ССА – штриховые блоки соответственно скобочным оболочкам СФА (что необходимо для разных целей);
- без отражения схемных оболочек – разгрузка отображения структуры.

Исполнение алгоритма

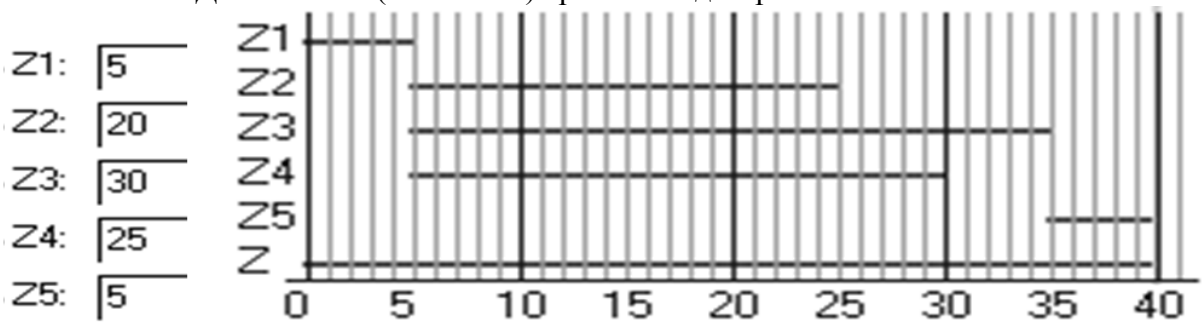
Вводятся длительности исполнения команд (условные единицы времени):

$mz1 = 5, mz2 = 20, mz3 = 30, mz4 = 25, mz5 = 5.$

Выводится временная диаграмма (Рис. 3.3) – статическая линейная диаграмма. При необходимости она достраивается до сетевой диаграммы вручную (пока).

ДИА: Диаграмма исполнения алгоритма / ГИ: Горизонтальное исполнение

ЛВД: Линейная (ленточная) временная диаграмма



СВД: Сетевая временная диаграмма

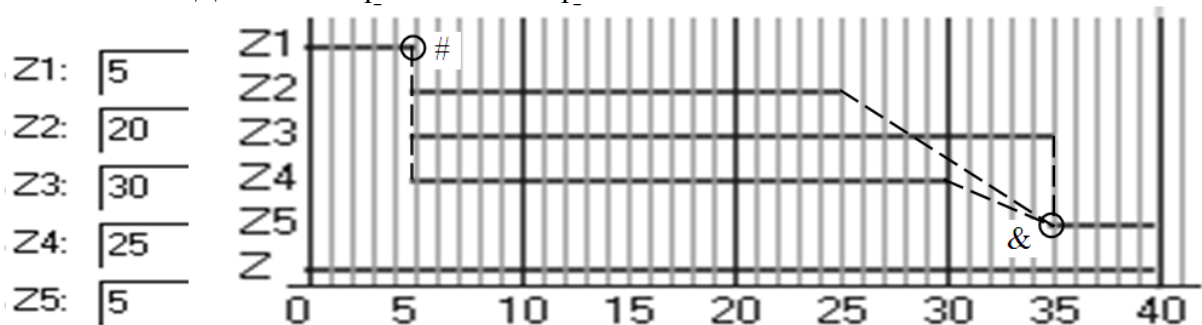


Рис. 3.3. Временная диаграмма процесса исполнения алгоритма

Примечание 4.1. Ручная доработка временной диаграммы

Необходимость ручной доработки временной диаграммы оказалось методически полезным фактором:

- обеспечивается быстрое автоматическое построение линейной временной диаграммы – это большая трудоемкая часть работы;
- ручная ее доработка до сетевой диаграммы значительно менее трудоемкая и требует понимания структурной сути дела.

Длительность цикла исполнения алгоритма

Для двухполюсных структур используется способ сетевых расчетов по длительности исполнения алгоритмов на основе СФА. Например:

РДА: Расчет длительности (исполнения) алгоритма:

ШФР: Шаблон формулы расчета (модификация СФА):

ИнФ: Инфиксная форма ИнПрФ: Инфиксно-префиксная форма

$$A53 = (Z1 \rightarrow (Z2 \ \& \ Z3 \ \& \ Z4) \rightarrow Z5) = Z1 \rightarrow \&(Z2, Z3, Z4) \rightarrow Z5$$

ФРД: Формула расчета длительности (на основе ИнПрФ):

$$ma53 = mz1 + \text{Max}(mz2, mz3, mz4) + mz5 = 5 + \text{Max}(20, 30, 25) + 5 = 40;$$

$$\text{резервы процессов (простои): } rzi = \text{Max}(i) - mzi, \quad rz2 = 10, \quad rz3 = 0, \quad rz4 = 5.$$

4 ПСЕВДОКОДЫ АЛГОРИТМОВ

Вербальные тексты алгоритмов (структуры текстов в служебной лексике)

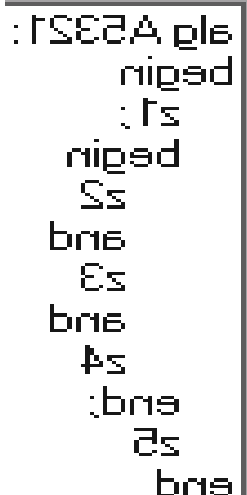
Вербальные тексты (псевдокоды) алгоритмов строятся на основе СФА (как шаблонов) в разной лексике, в горизонтальном (для теоретических задач) и вертикальном исполнении (построчная пооператорная запись с отступами строк).

ВТА: Вербальный текст алгоритма / ВИ: Вертикальное исполнение

АлгПТ: Алгол-подобный текст / ПасПТ: Паскаль-подобный текст

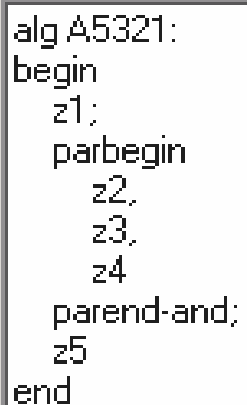
ИнФ: Инфиксная форма

$A53 = (Z1 \rightarrow (Z2 \# \& Z3 \# \& Z4) \rightarrow Z5) = (Z1 \rightarrow (Z2 \& Z3 \& Z4) \rightarrow Z5)$

Вариант 1		Вариант 2	
A53 =	alg A53:	A53 =	
(begin	(:rS&Z&A gle
Z1→	Z1;	Z1→	ri&ed
(begin	(:rS
Z2 &	Z2 and	Z2	ri&ed
Z3 &	Z3 and	&	Ss
Z4	Z4	Z3	bn&
)→	end;	&	Es
Z5	Z5	Z4	bn&
)	end)→	As
		Z5	:bn&
)	as
			bn&

ИнПрПоФ: Инфиксно-префиксно-постфиксная форма

$A53 = (Z1 \rightarrow (Z2 \# \& Z3 \# \& Z4) \rightarrow Z5) = (Z1 \rightarrow \#(Z2, Z3, Z4) \& \rightarrow Z5)$

A53 =	alg A53:		alg A53 :: y = f(x):
(begin	alg A5321:	begin
Z1→	Z1;	begin	(v1,v2,v3) := f0(x);
#(parbegin	z1;	parbegin // cobegin
Z2,	Z2,	parbegin	u1 := f1(v1),
Z3,	Z3,	z2,	u2 := f2(v2),
Z4	Z4	z3,	u3 := f3(v3)
)&→	parend;	z4	parend; // coend
Z5	Z5	parend-and;	y := f4(u1, u2, u3)
)	end	z5	end
		end	

ОкПТ: Оккам-подобный текст / ПрФ: Префиксная форма:

$$A53 = (Z1 \rightarrow (Z2 \ \#\& \ Z3 \ \#\& \ Z4) \rightarrow Z5) = \rightarrow(Z1, \ \#\&(Z2, \ Z3, \ Z4), \ Z5)$$

<pre>A53 = → (Z1, #& (Z2, Z3, Z4), Z5)</pre>	<pre>A53 = → Z1 #& Z2 Z3 Z4 Z5</pre>	<pre>alg A53: seq Z1 par Z2 Z3 Z4 Z5</pre>	<pre>alg A: seq z1 par-and z2 z3 z4 z5</pre>	<pre>alg A53 :: y = f(x): seq (v1,v2,v3) := f0(x) par // par = par-and u1 := f1(v1) u2 := f2(v2) u3 := f3(v3) y := f4(u1, u2, u3)</pre>
--	--	--	--	---

Уточнение семантики стандартной лексики для конъюнкции (&):

ИнФ:	and
ИнПрПоФ	parend = parend-and = end-and.
ПрФ:	par = par-and

5 РАСШИРЕНИЯ БАЗОВОГО СТРУКТУРНОГО КЛАССА

Приведенные двухполюсники

На основе исходных определений двухполюсников применяется приведение многополюсников к условным двухполюсникам:

- определяются основной вход и основной выход – по ним проводятся структурные операции как с обычными двухполюсниками $A_i = \rightarrow A_i \rightarrow$
- прочие выходы и входы определяются как побочные $A_i = \rightarrow A_i (\uparrow, \downarrow) \rightarrow$

На этой основе возможно описание любых классов структур алгоритмов.

Сохраняется общий синтаксис двухполюсников, появляются дополнения и особенности, усложняются сетевые расчеты длительности алгоритмов и т.п.

Пример 2. Многополюсная структура. Дизъюнктивная сборка

Частичный параллелизм ($p=2$) для исходной задачи Примера 1 ($p_{\max}=3$).

СФА: Структурная формула алгоритма / У: Поток управления

ИнФ: Инфиксная форма (полная и краткая):

$$A_{52} = (Z_1 - ((Z_2 \uparrow 1) \vee (Z_3 \uparrow 2)) - Z_4 \downarrow \& 1,2 - Z_5) = Z_1 (Z_2 \uparrow^1 \vee Z_3 \uparrow^2) Z_4 \downarrow \&^{1,2} Z_5$$

ССА: Структурная схема алгоритма (литерный набор схемы):

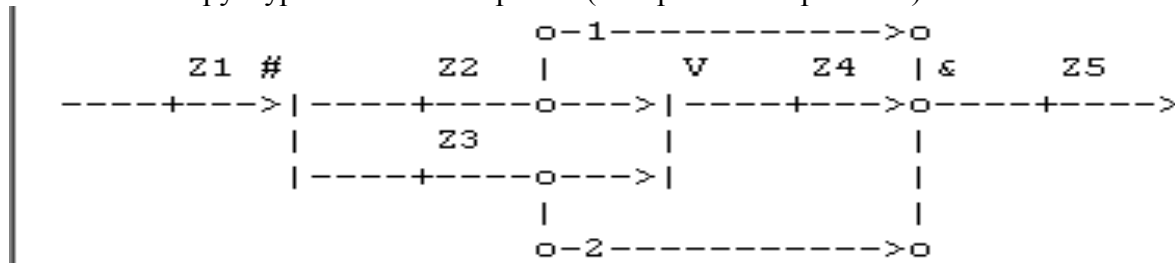


Рис. 5.1. Штрих-схема алгоритма: клавиатурная псевдографика

ДИА: Диаграмма исполнения алгоритма (динамическая реализация)

ЛВД: Линейная временная диаграмма

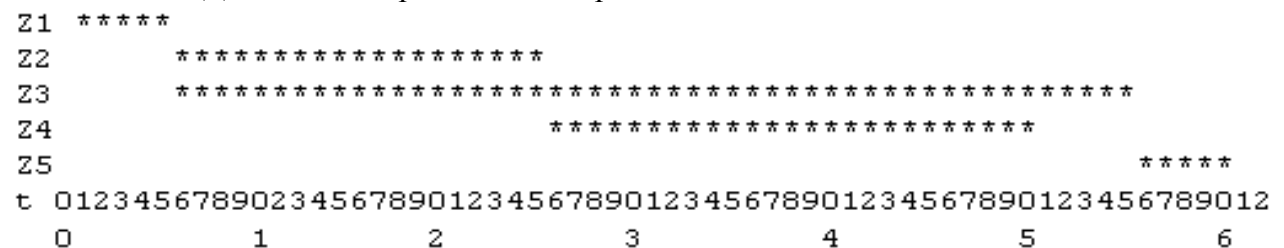


Рис. 5.2. Временная диаграмма – клавиатурная псевдографика

ВТА: Вербальный текст алгоритма / ГИ: Горизонтальное исполнение

АлгПТ: Алгол-подобный текст / ИнФ: Инфиксная форма

alg A20: begin Z1; begin begin Z2; fork(L1) end or begin Z3; fork(L2) end end; Z4; join(L1, L2); Z5 end

Используется статическая разновидность операторов fork: \uparrow^i и join: $\downarrow \&^i$ – параллельные переходы по меткам L_i (статические параллельные структуры).

По диаграмме процесс Z4 использует резерв процесса Z2.

Примечание 6.1. Связь структурных формул с логическими схемами алгоритмов (ЛСА)

1) Структурные формулы приведенных двухполюсников – это особые стрелочные формулы, которые содержат стрелочные связи двух типов:

- горизонтальные (условно продольные) стрелочные связи – линейная передача управления между операторами:
эти стрелки в записи структурных формул могут опускаться – неявная секвенция;
- вертикальные (условно поперечные) стрелочные связи – нелинейная передача управления между операторами.

2) Такие стрелочные структурные формулы являются обобщением языка так называемых *логических схем алгоритмов* (ЛСА) и его расширения – языка *параллельных ЛСА* (ПЛСА):

на самом деле это особые стрелочные структурные формулы алгоритмов – с поперечными стрелками (и без продольных стрелок – неявная секвенция);

ЛСА и ПЛСА обеспечивают отображение нелинейных условных переходов, безусловных переходов (типа *goto*), организацию циклов разных типов и т.п.

ЗАКЛЮЧЕНИЕ

Вторая очередь программно-методического комплекса

Выполняются разработки по унификации, интеграции в единый комплекс и развитию программной поддержки, алгоритмическому анализу учебных параллельных программ – параллельный счет в локальной сети в среде MPI (истинный параллелизм), многопоточные программы на основе тредов Java (псевдо-параллелизм), обмен данными, взаимные исключения процессов и т.п.