10. Анализ потока данных на основе областей

9.1 Области графа потока управления

- Итеративный алгоритм анализа потока данных является только одним из подходов к решению задач потоков данных, существуют и другие подходы,
  в частности — анализ на основе областей (region-based analysis)
- В случае итеративных алгоритмов использовалась передаточная функция для каждого базового блока, анализ на основе областей *находит передаточную функцию*, которая подытоживает выполнение целой области программы
- Структура потока данных, использующая итеративный алгоритм определяется полурешеткой значений потока данных и семейством передаточных функций, замкнутых относительно композиции
- Структура потока данных, использующая анализ на основе областей включает полурешетку значений потока данных, ***полурешетку передаточных функций***, которая должна располагать оператором сбора, оператором композиции и *оператором замыкания*

9.1 Области графа потока управления

9.1.1. Определение области

◊ **Определение.** *Областью* графа потока называется его подграф $R = \langle N_R, E_R \rangle$ такой, что

1. существует узел $h \in N_R$, доминирующий над всеми узлами в R; этот узел называется *заголовком области* R;
2. если из некоторого узла r_2 графа потока управления можно достичь узла $r_1 \in R$, минуя заголовок h, то и $r_2 \in R$;
3. множество E_R включает все ребра графа потока управления между любыми узлами $r_1, r_2 \in R$, за исключением некоторых ребер, входящих в заголовок h;

Единственным входом в область является ее заголовок.

9.1 Области графа потока управления

9.1.1. Определение области

◊ **Определение (менее формальное).** Область – некоторое подмножество узлов ГПУ, один из которых является заголовком, и доминирует все остальные, а вход в область возможен только через заголовок.

◊ В отличие от итеративного алгоритма анализ на основе областей использует передаточную функций не для одного базового блока, а для более крупной единицы – области.

◊ Если удастся создать область для всей процедуры целиком, то применяя передаточную функцию такой области можно получить значение потока данных на выходе из процедуры.

6.1. Повторение: естественные циклы

6.1.1 Определение естественного цикла

- **Определение**. *Естественным циклом* называется цикл со следующими свойствами:
  - Цикл имеет единственный входной узел, называемый его *заголовком*,
  - Существует обратное ребро, ведущее в заголовок цикла

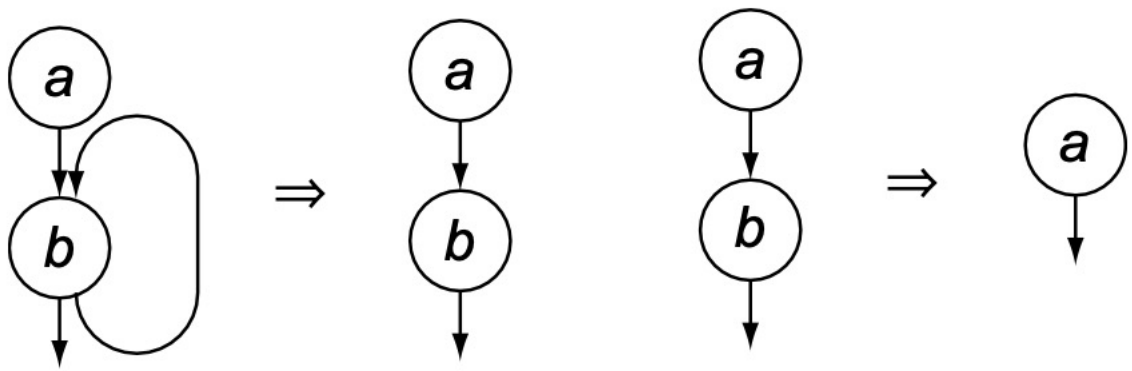
- **Определение**. *Естественный цикл обратного ребра* $\langle B_i, B_k \rangle$ составляют узел B_k (*заголовок цикла*) и все узлы ГПУ, из которых можно достичь узла B_i, не проходя через узел B_k. (эти узлы составляют *тело цикла*).

6.1. Повторение: естественные циклы

6.1.1.1 Определение приводимости цикла

- **Определение 1 (Cooper et al.)**

  Граф потока называется *приводимым (reducible)*, если применение преобразований T_1 и T_2 сводит его к одному узлу (преобразования могут быть применены многократно в любом порядке).



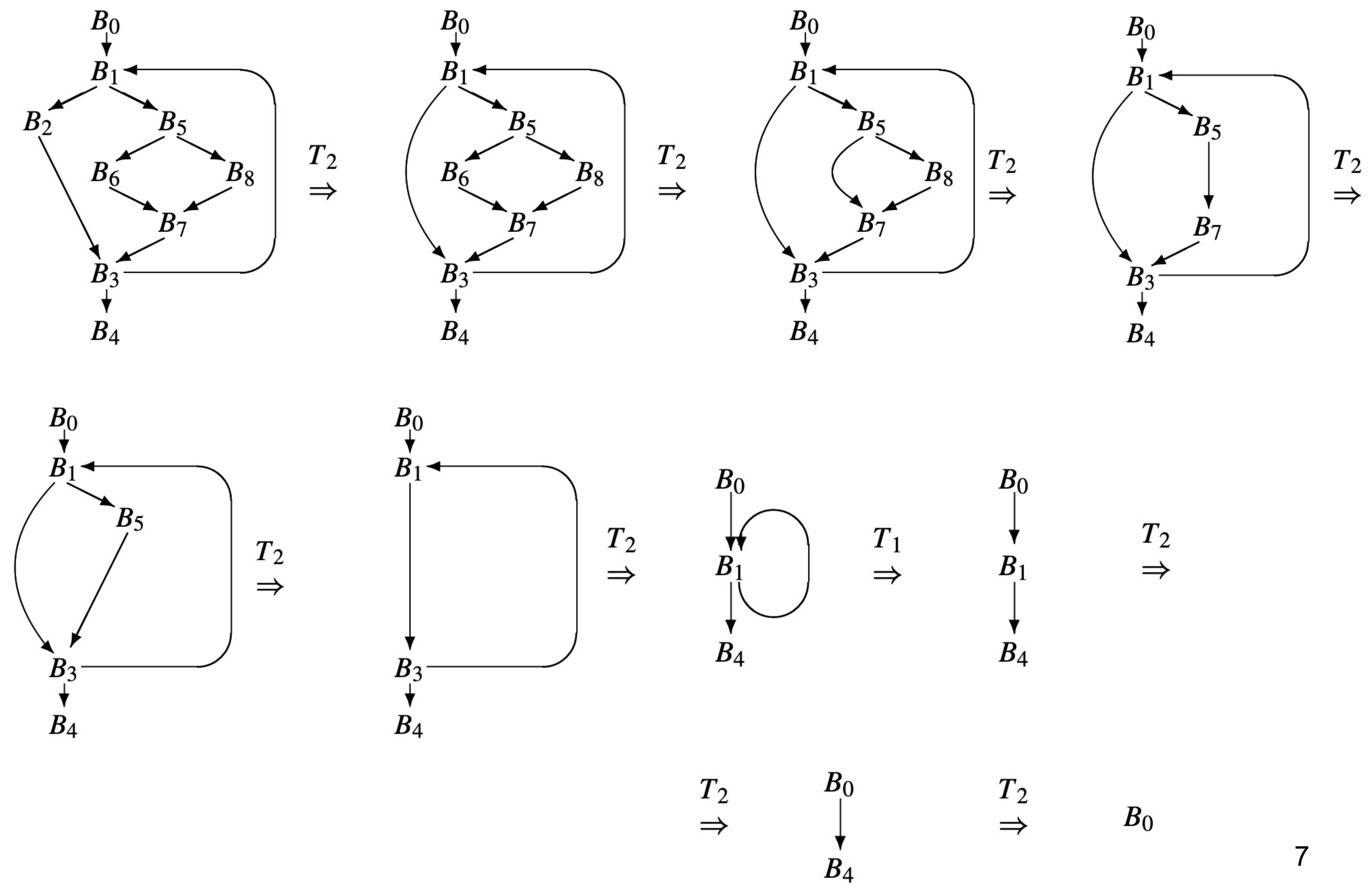
$T_1(b)$ $T_2(a,b)$

- ***T_1*: удаляет обратную дугу в цикле, состоящем из одной вершины**

- ***T_2*: сворачивает узел *b* с единственным предком, присоединяя его к *a*. При этом дуга *(a, b)* удаляется, а также а становится источником дуг, исходивших из *b*. Если при этом возникает несколько дуг из *a* в некоторый узел *n*, они объединяются.

6.1. Выделение естественных циклов

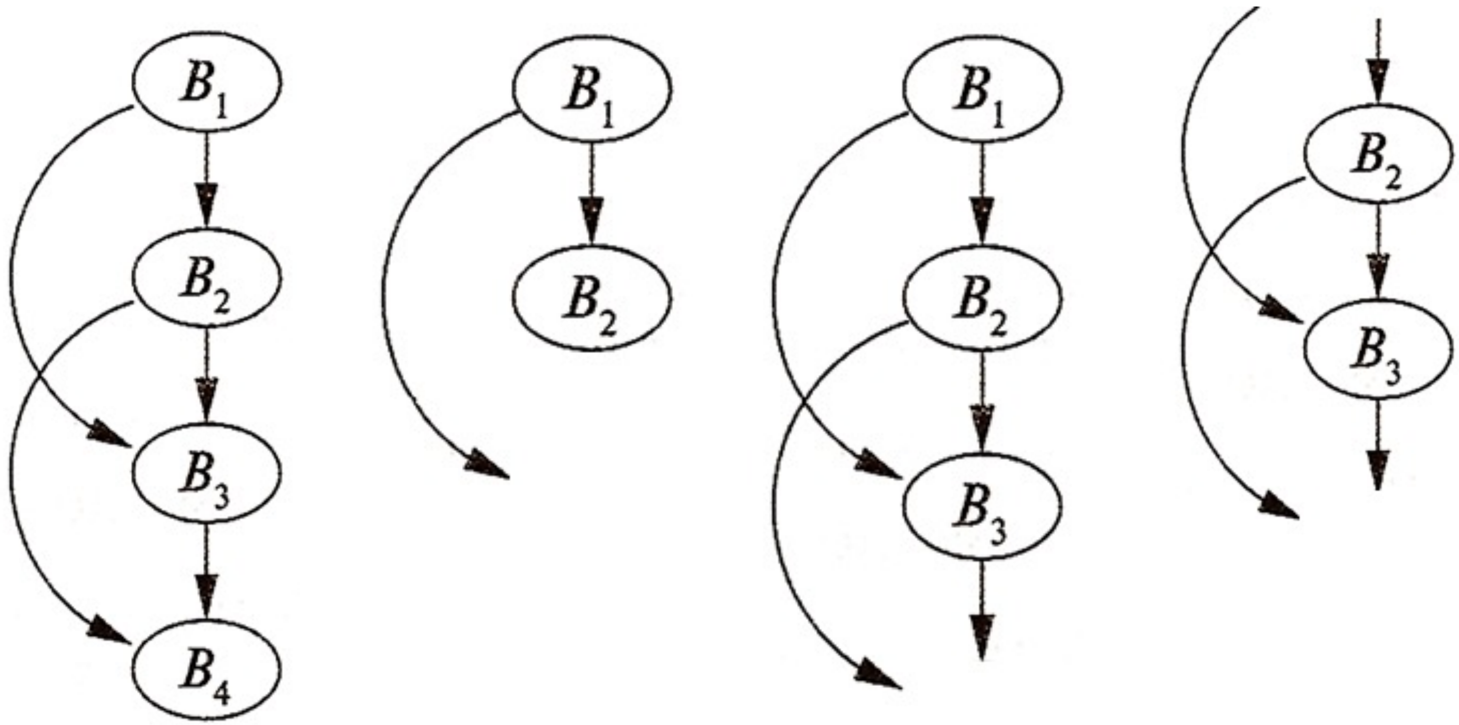
6.1.1.1 Определение приводимости цикла



9.1 Области графа потока управления

9.1.1. Определение области

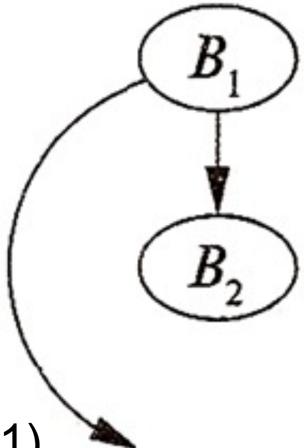
◊ **Пример 1.**



9.1 Области графа потока управления

9.1.1. Определение области

◊ **Примеры.**

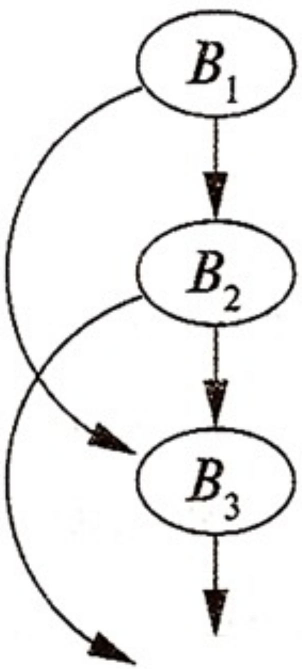


(1)

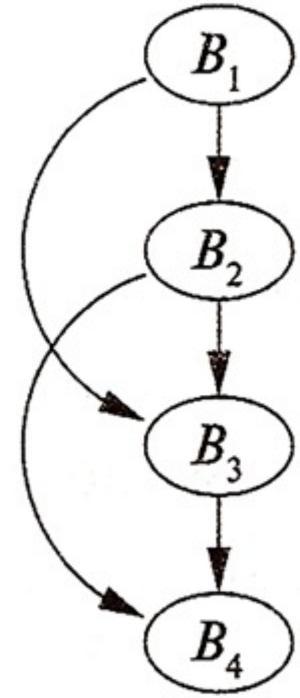
7. Рассмотрим граф на рисунке справа

1) узлы B_1 и B_2 вместе с ребром $B_1 \to B_2$ образуют область с заголовком B_1

2) узлы B_1, B_2 и B_3 и ребра $B_1 \to B_2$, $B_2 \to B_3$, $B_1 \to B_3$ образуют область с заголовком B_1



(2)

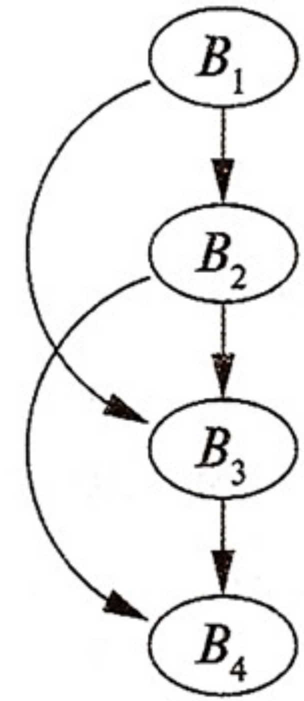


(3)

3) узлы B_1, B_2, B_3 и B_4 со всеми ребрами образуют область с заголовком $B_{1.}$ Во-первых, область можно свернуть, применяя шаблон T_2, а во-вторых, если добавить обратную дугу $B_1 \to B_4$, то сразу понятно, что область представляет собой тело естественного цикла (B_4 достижим из B_2 и B_3, не проходя через B_1)

9.1 Области графа потока управления

9.1.1. Определение области



◊ **Примеры.**

1. Рассмотрим граф на верхнем рисунке

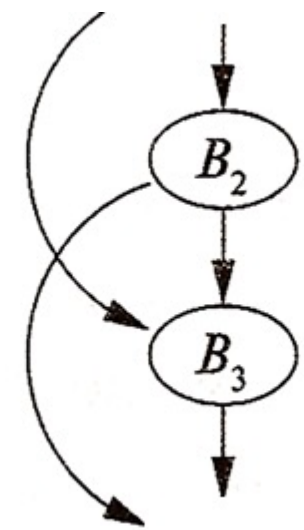
3) подграф $R = \langle \{B_2, B_3\}, \{B_2 \to B_3\} \rangle$ **область не образует**, так как управление может попасть в него и через B_2, и через B_3:

B_2 не является доминатором B_3,

B_3 не является доминатором B_2

и, следовательно, условие 1 определения 9.1.1 (существует узел $h \in N_R$, доминирующий над всеми узлами в R;)

не выполняется (нижний рисунок)



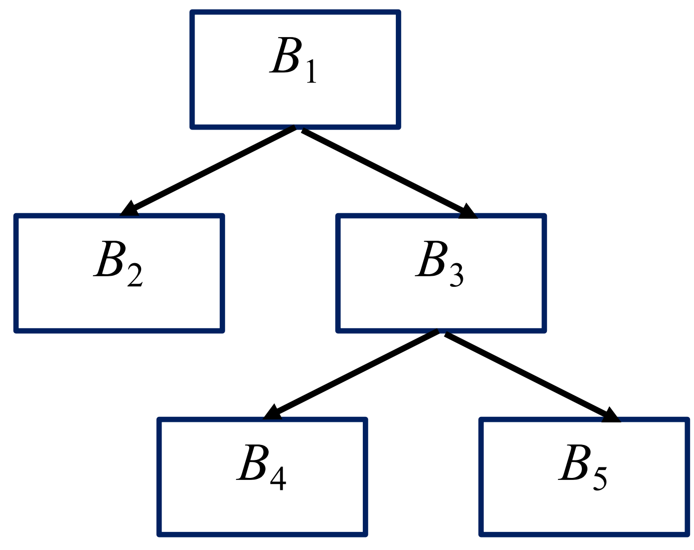
9.1 Области графа потока управления

9.1.1. Определение области

◊ **Пример 2.**

Суперблок (рассматривался, когда изучался метод глобальной нумерации значений) – пример области. В частности, граф на рисунке – область с заголовком B_1.

Узел B_1 доминирует над остальными узлами:

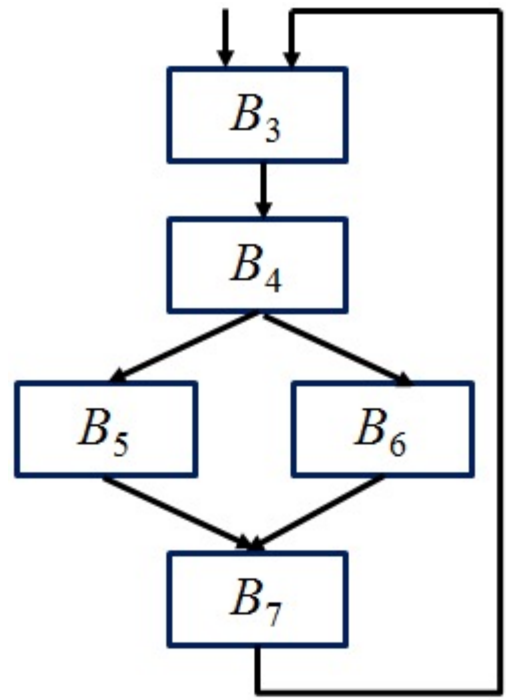


9.1 Области графа потока управления

9.1.1. Определение области

◊ **Пример 3.**

Граф на рисунке – область с заголовком B_3 (*естественный цикл*).



9.1 Структурный анализ графа потока управления

9.1.2. Классификация областей

◊ **Определения**:

(1) Каждый базовый блок B может рассматриваться как область $R = \langle \{B\}, \varnothing \rangle$. Такая область называется ***область-лист***.

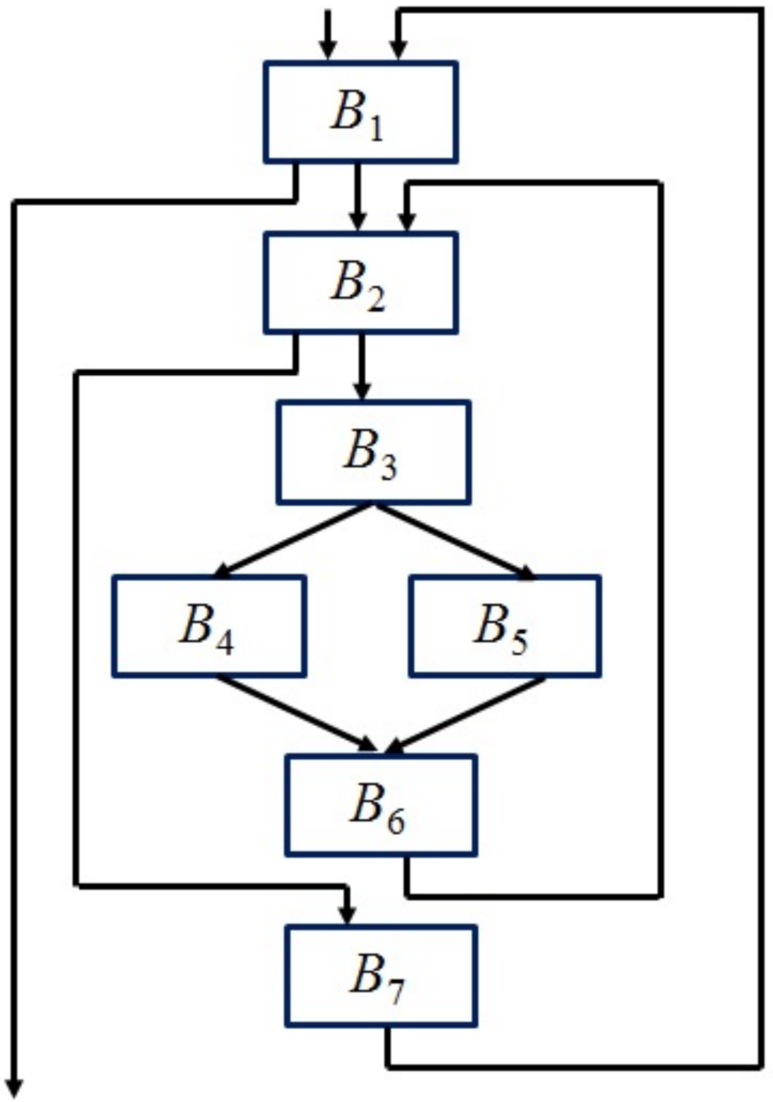
(2) Пусть L – самый внутренний цикл гнезда циклов.

Тело цикла L (все узлы и ребра, за исключением обратных ребер к заголовку цикла) можно заменить узлом, представляющим область R. Такая область называется ***область-тело***.

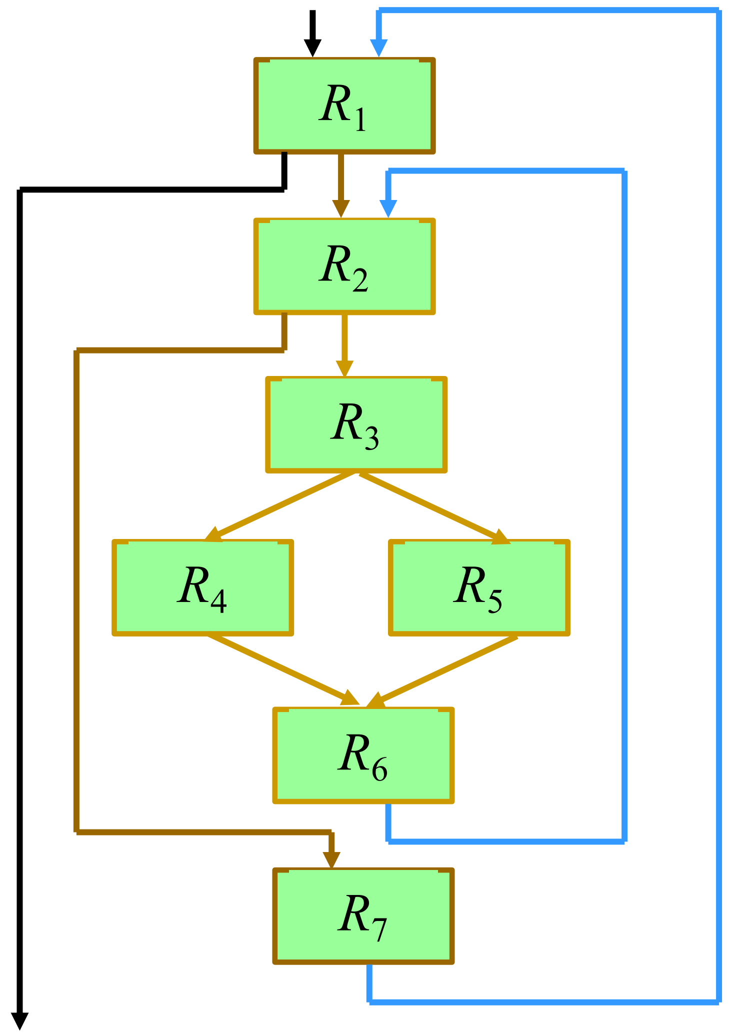
(3) Если к области-телу R, соответствующей телу цикла L присоединить обратное ребро к заголовку цикла L, получится новая область Q. Такая область называется ***область-цикл***.

9.1 Структурный анализ графа потока управления

9.1.2. Виды областей. Пример



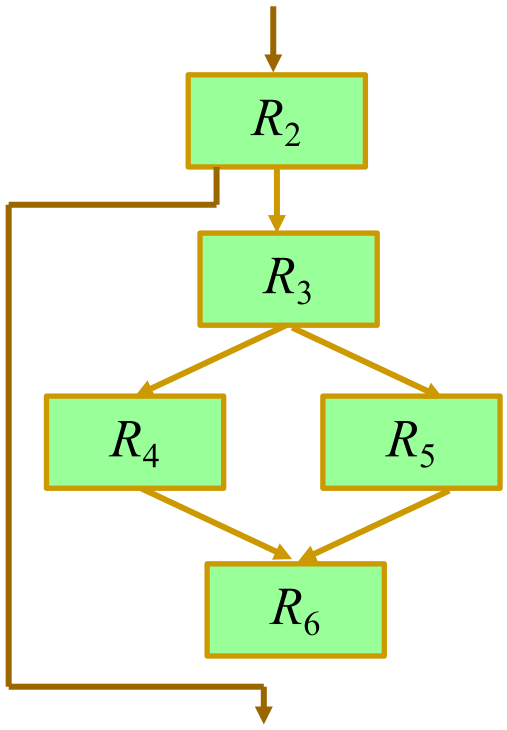
Фрагмент процедуры



Базовые блоки – области-листья

9.1 Структурный анализ графа потока управления

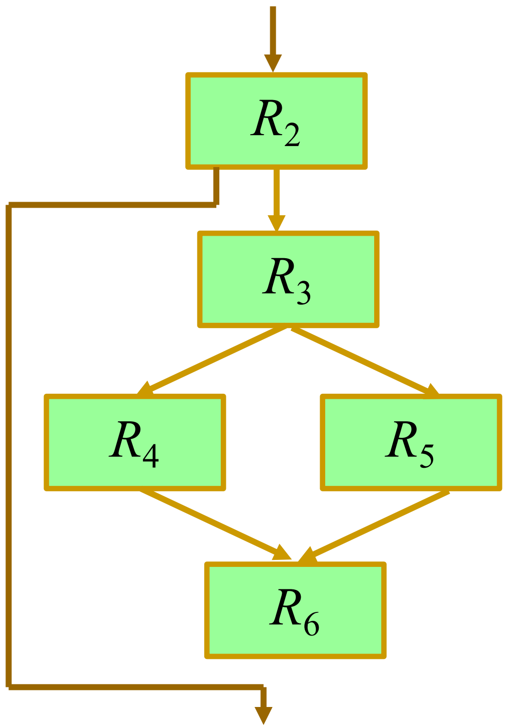
9.1.2. Виды областей. Пример



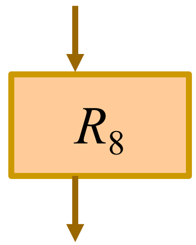
Тело внутреннего цикла

9.1 Структурный анализ графа потока управления

9.1.2. Виды областей. Пример



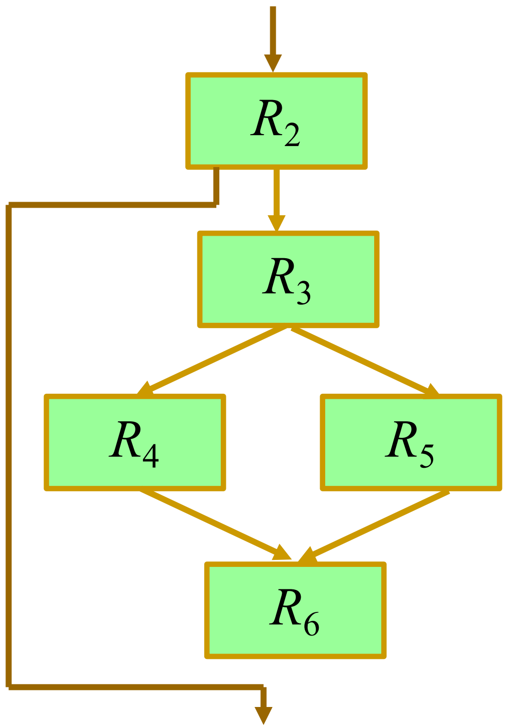
Тело внутреннего цикла



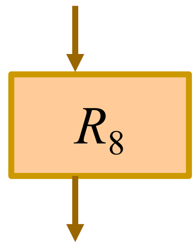
можно заменить на область-тело R_8

9.1 Структурный анализ графа потока управления

9.1.2. Виды областей. Пример

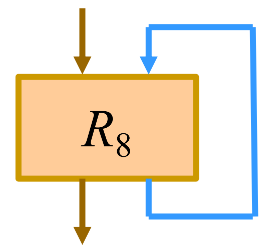


Тело внутреннего цикла

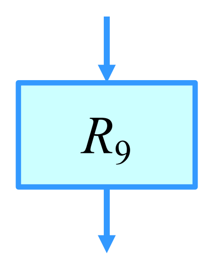


можно заменить на область-тело R_8

добавив обратную дугу,



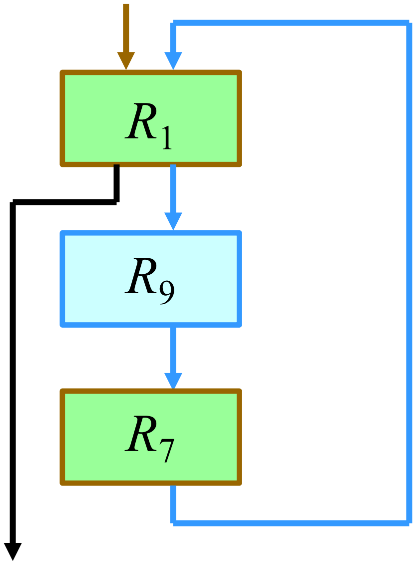
получим область-цикл R_9



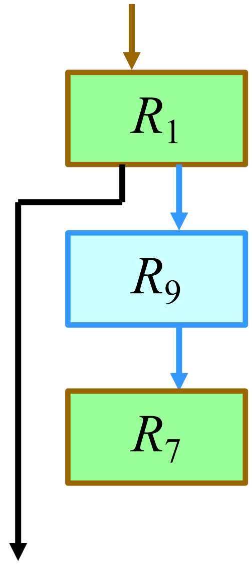
9.1 Структурный анализ графа потока управления

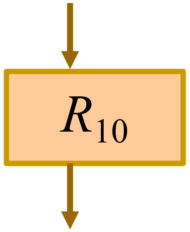
9.1.2. Виды областей. Пример

Граф потока управления примет вид

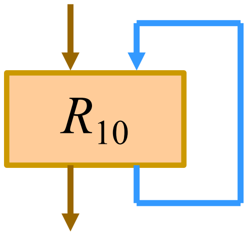


Тело внешнего цикла

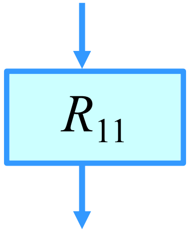




Заменяется областью-телом R_{10}



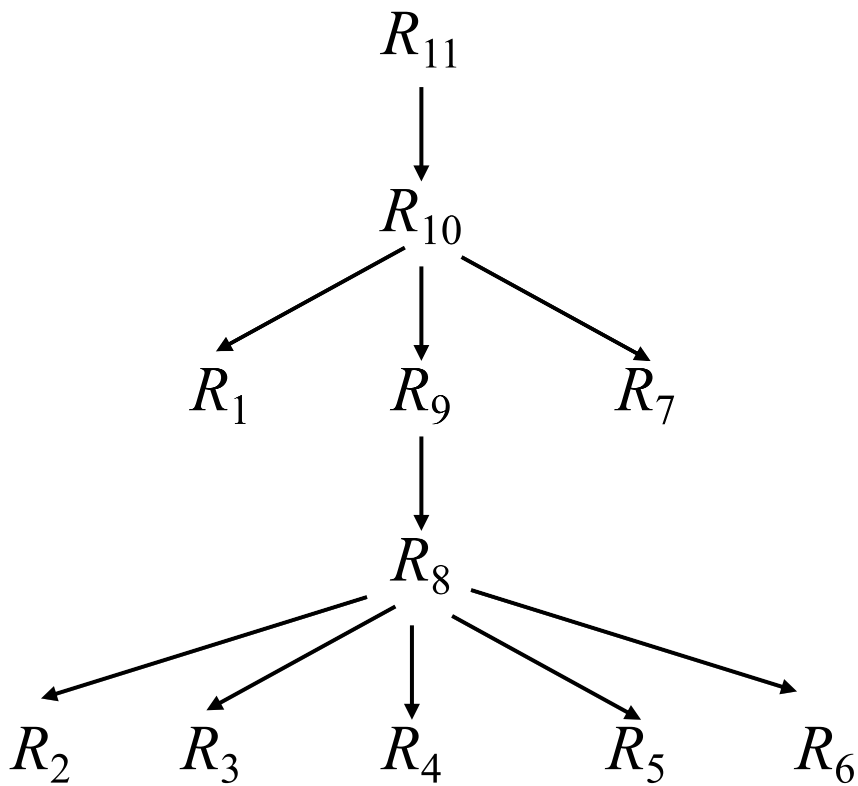
Добавив обратную дугу, получим область-цикл R_{11}



9.1 Структурный анализ графа потока управления

9.1.2. Виды областей. Пример

Дерево управления

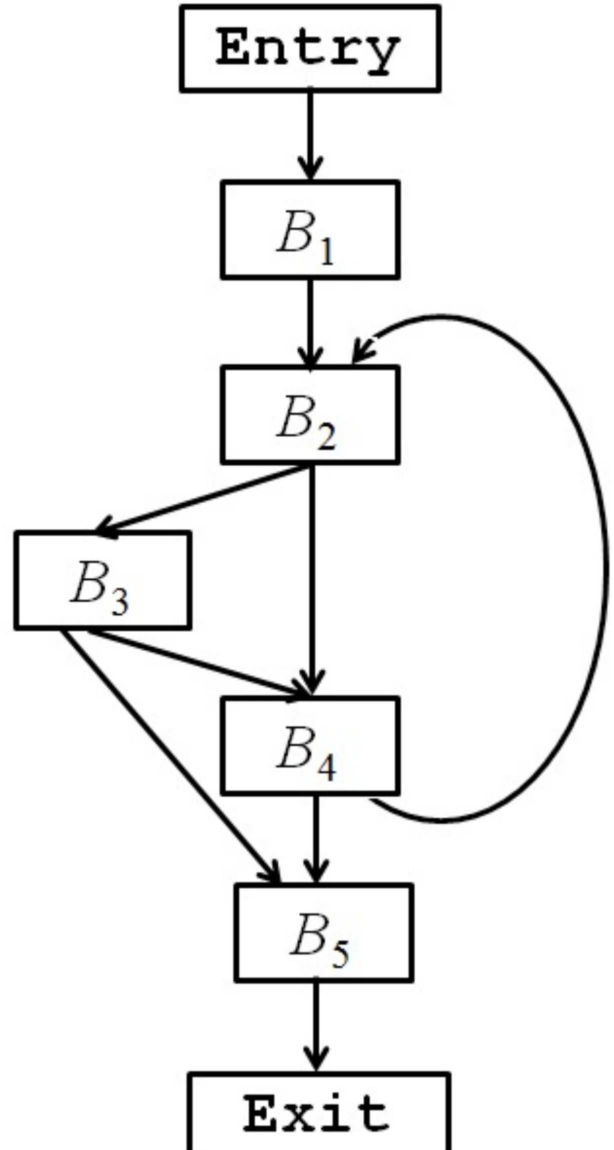


9.1 Структурный анализ графа потока управления

9.1.3. Выделение областей

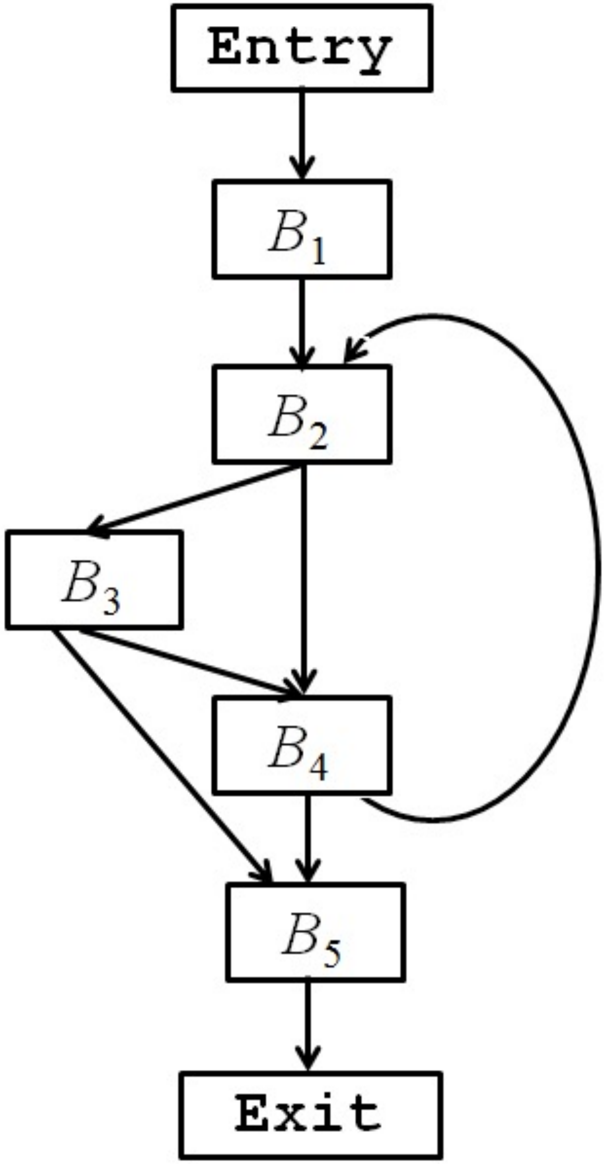
◊ Рассмотрим ГПУ, показанный на рисунке.

B_1	**i ← - m, 1**	d_1
	j ← n	d_2
	a ← u1	d_3
B_2	**i ← + i, 1**	d_4
B_3	**a ← u2**	d_5
B_4	**j ← u3**	d_6
B_5	**. . . .**	

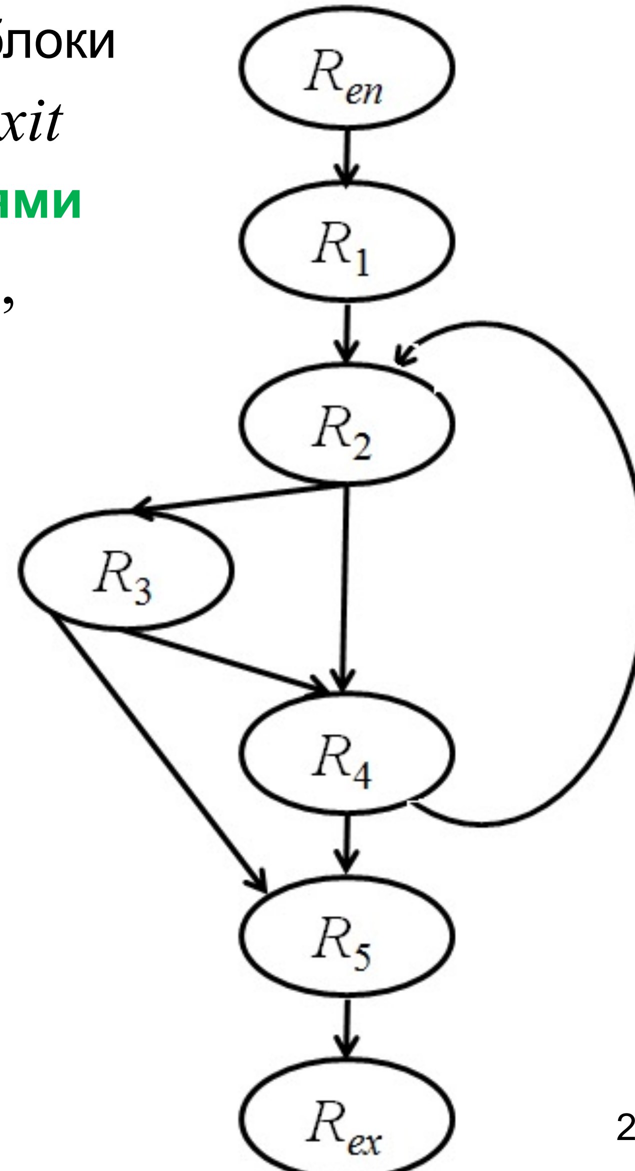


9.1 Структурный анализ графа потока управления

9.1.3. Выделение областей

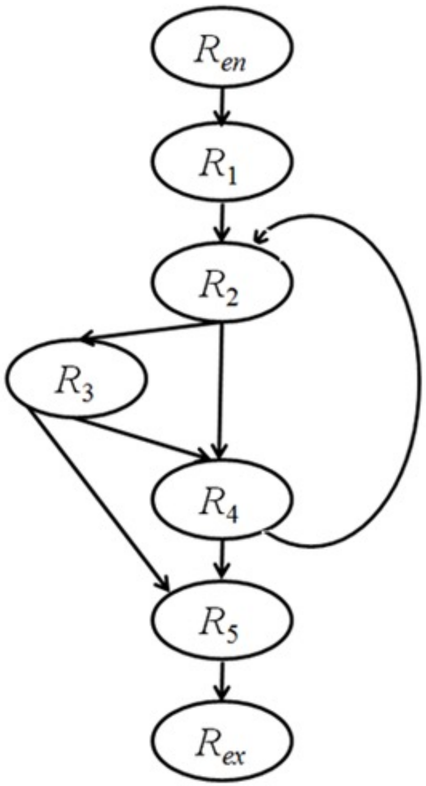


1) Заменив базовые блоки $Entry, B_1, \ldots, B_5, Exit$ **областями-листьями** $R_{en}, R_1, \ldots, R_5, R_{ex}$, получим граф



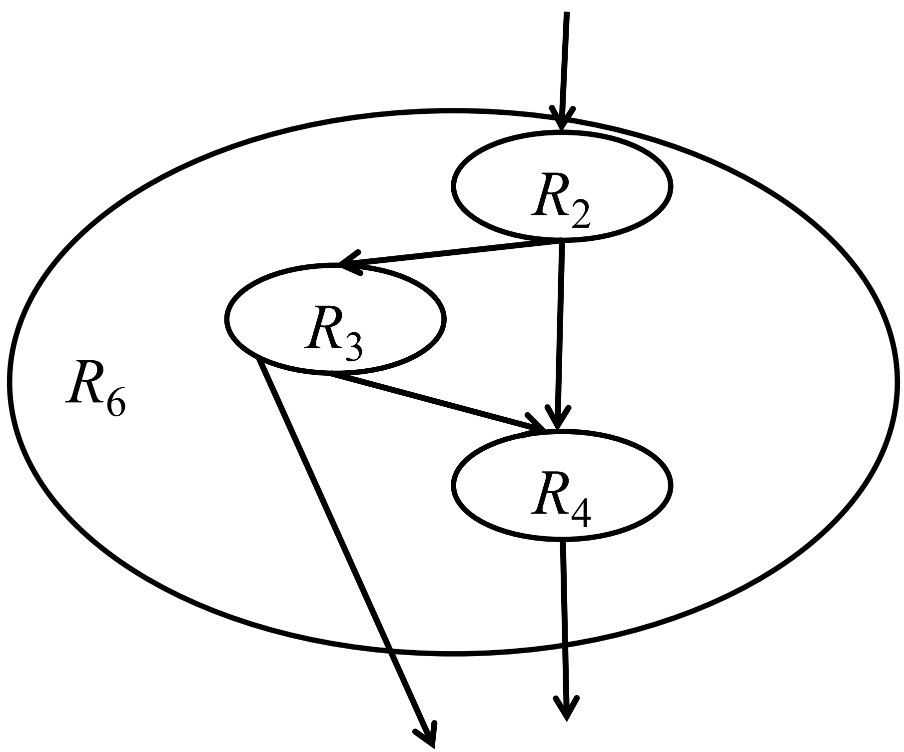
9.1 Структурный анализ графа потока управления

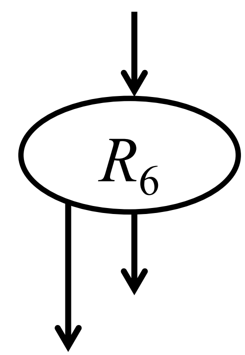
9.1.3. Выделение областей



2) Области-листья R_2, R_3 и R_4 и ребра $R_1 \to R_2$, $R_2 \to R_3$, $R_2 \to R_4$, $R_3 \to R_4$, $R_3 \to R_5$, $R_4 \to R_5$ составляют **область-тело** R_6 .

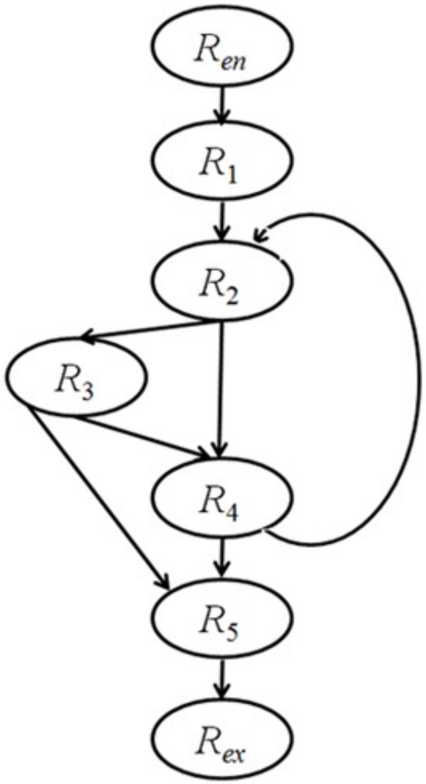
$R_6 = \langle \{R_2, R_3, R_4\}, \{R_1 \to R_2, R_2 \to R_3, R_2 \to R_4, R_3 \to R_4, R_3 \to R_5, R_4 \to R_5\} \rangle$



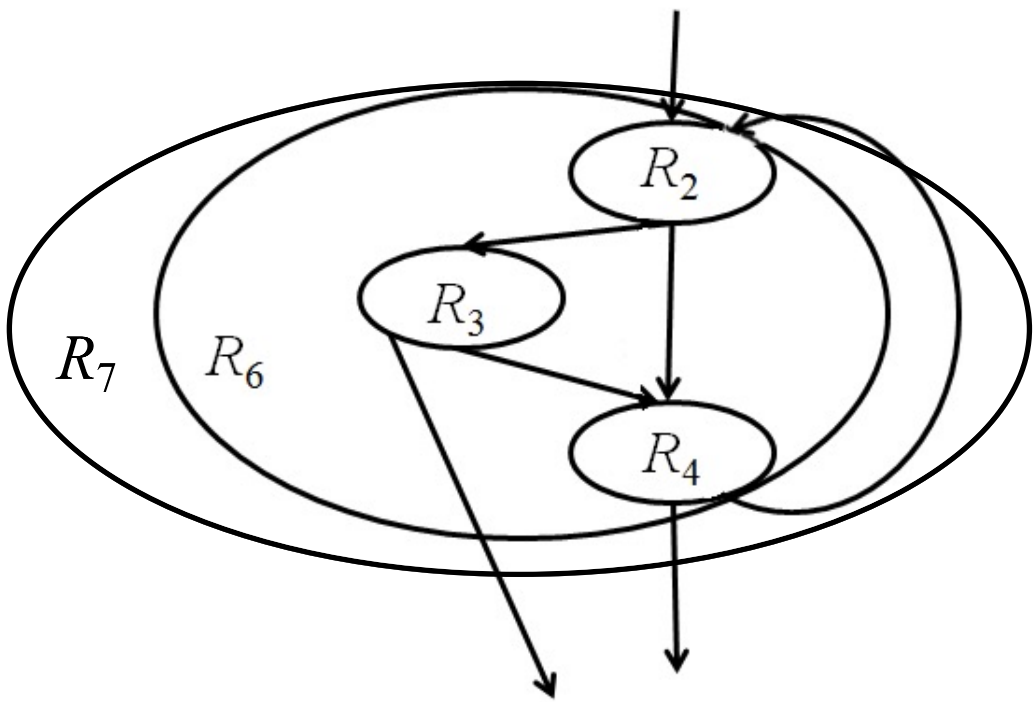


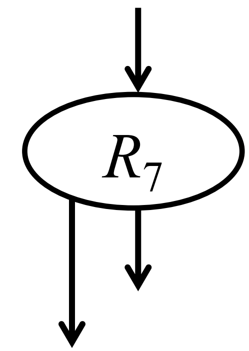
9.1 Структурный анализ графа потока управления

9.1.3. Выделение областей



3) Область-тело R_6, и обратное ребро $R_4 \to R_2$ составляют **область-цикл** $R_7 = \langle \{R_6\},\ \{R_4 \to R_2\} \rangle$.





9.1 Структурный анализ графа потока управления

9.1.4. Алгоритм построения иерархии областей

◊ **Алгоритм**.

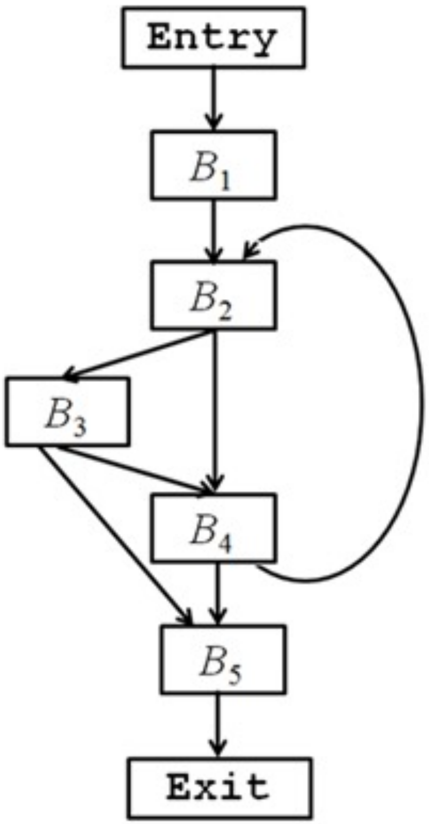
1. Найти все естественные циклы.
2. Найденные естественные циклы упорядочить изнутри гнезд наружу, т.е. начиная с наиболее внутренних циклов.
3. Выбрать очередной естественный цикл (сначала – самый первый, потом – следующий по порядку).
   Если циклов больше нет, **алгоритм заканчивается**.
4. Тело выбранного цикла L (все узлы и ребра, за исключением обратных ребер к заголовку) заместить узлом R, представляющим область-тело.

   После замещения обратное ребро в заголовок L становится петлей.
5. Построить область-цикл Q, представляющую цикл L (в отличие от области R область Q не содержит петли).
   Перейти к шагу 3.

9.1 Структурный анализ графа потока управления

9.1.5. Построение иерархии областей

◊ **Пример**. Применим алгоритм к следующему ГПУ

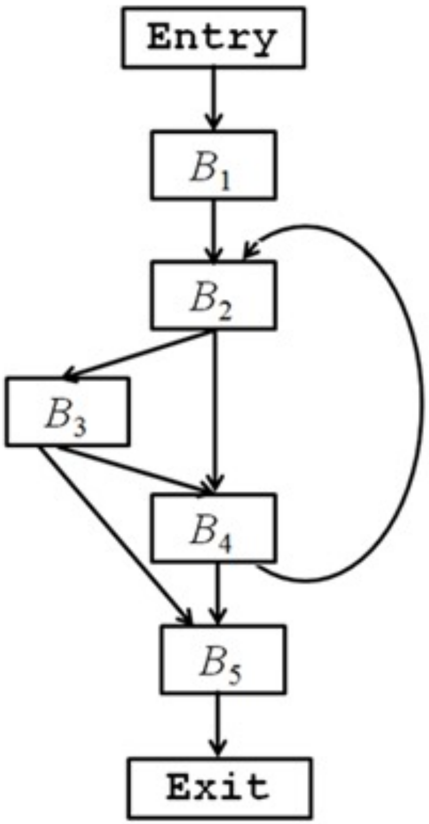


Исходный ГПУ

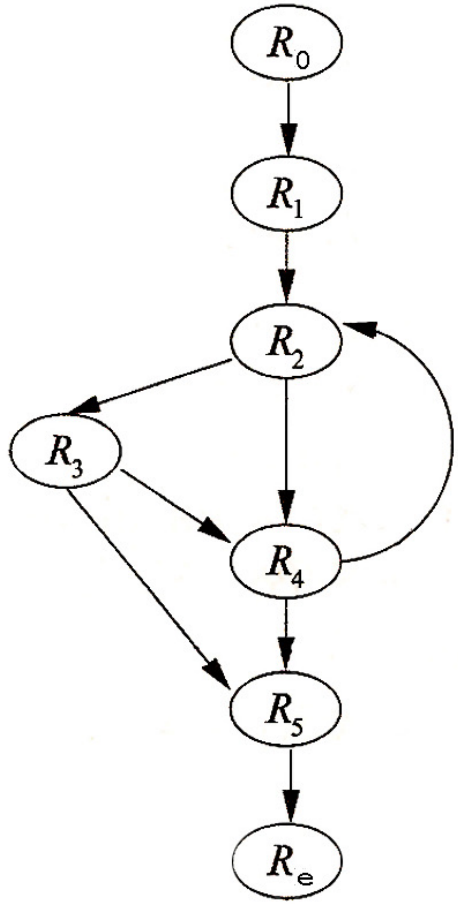
9.1 Структурный анализ графа потока управления

9.1.5. Построение иерархии областей

◊ **Пример**. Применим алгоритм к следующему ГПУ



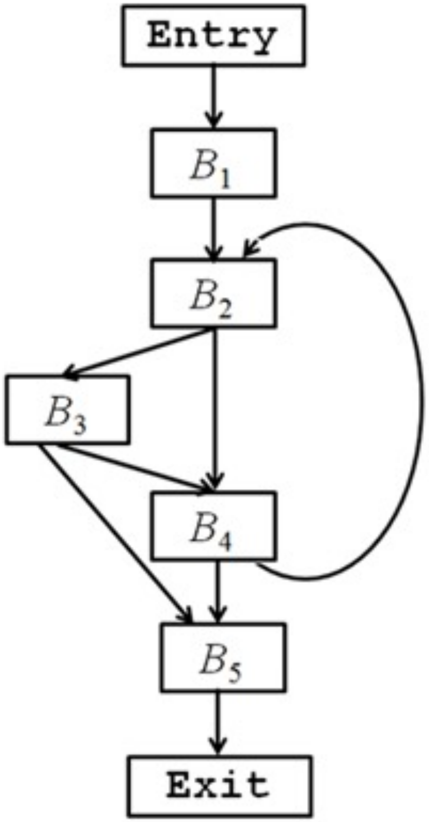
Исходный ГПУ



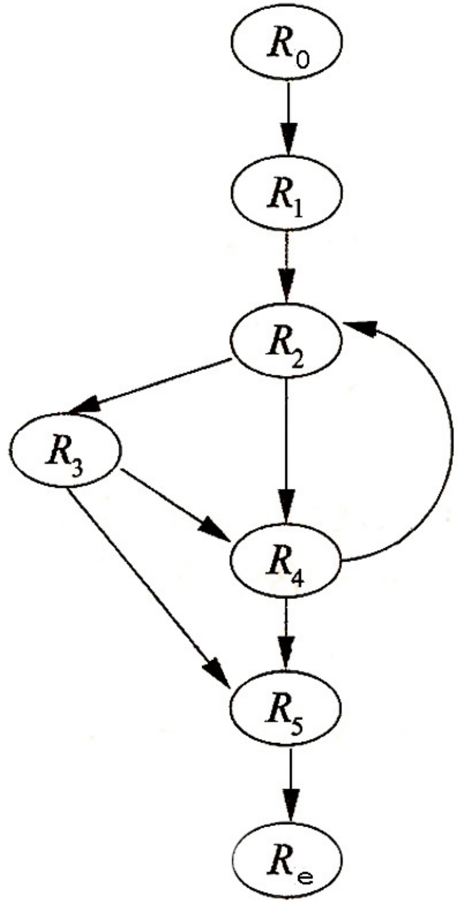
Базовые блоки заменены областями-листьями

9.1.5. Построение иерархии областей

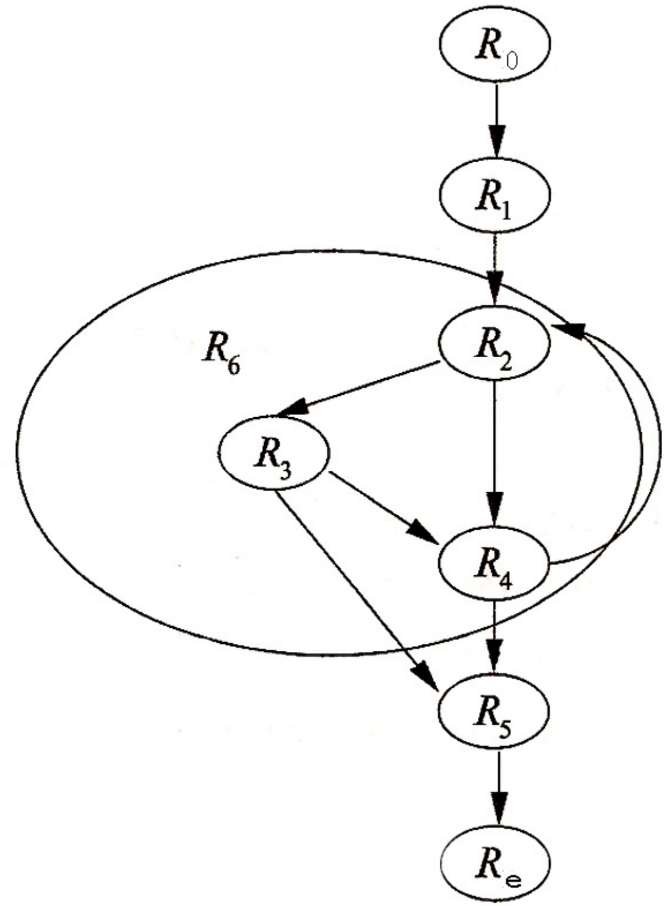
◊ **Пример**. Применим алгоритм к следующему ГПУ



Исходный ГПУ



Базовые блоки заменены областями-листьями

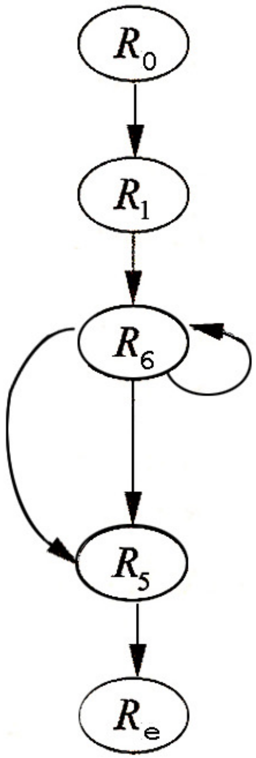


После шага 2

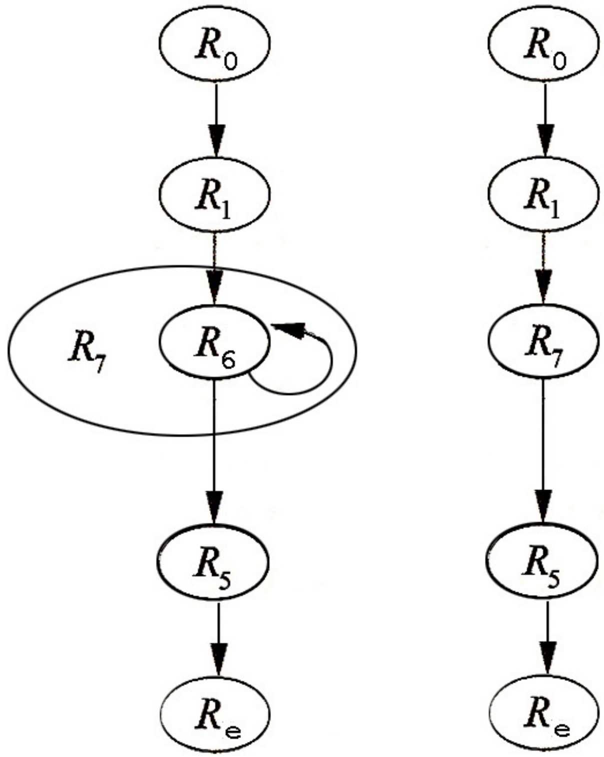
9.1 Структурный анализ графа потока управления

9.1.5. Построение иерархии областей

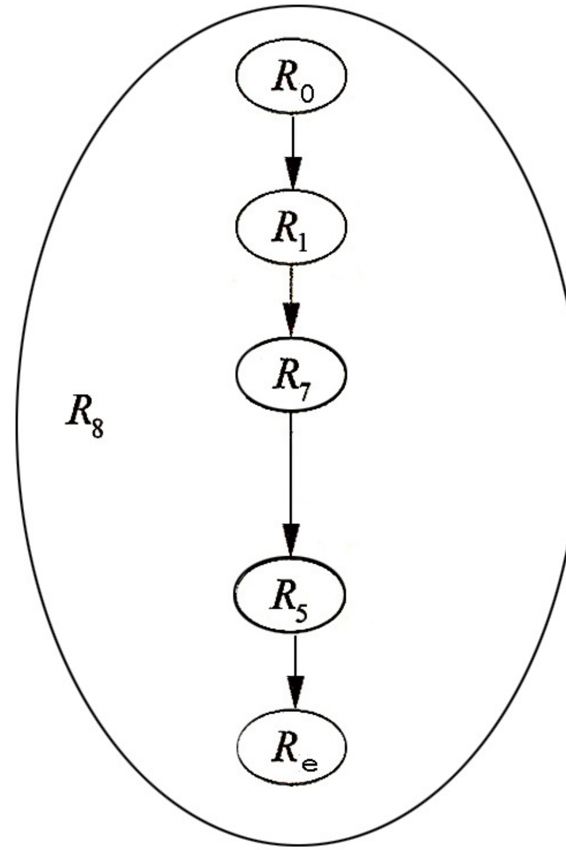
◊ **Пример**.



После шага 4



После шага 5

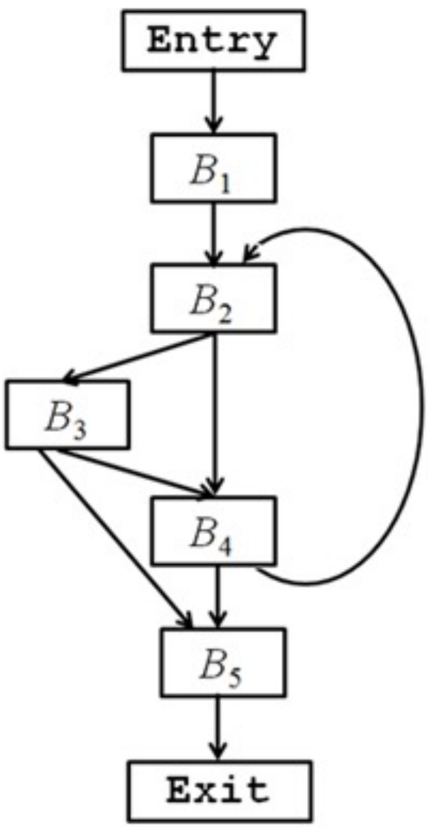


Область R_8 – весь ГПУ

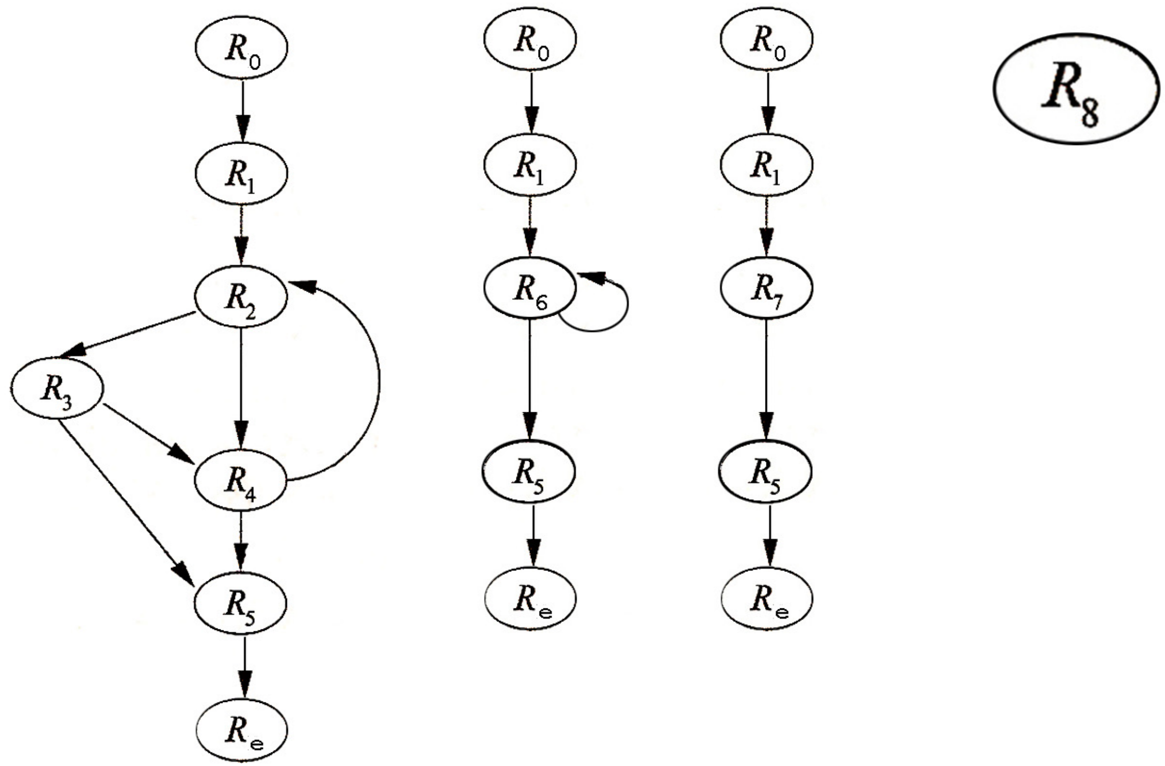
9.1 Структурный анализ графа потока управления

9.1.5. Построение иерархии областей

◊ **Пример**.



Исходный ГПУ

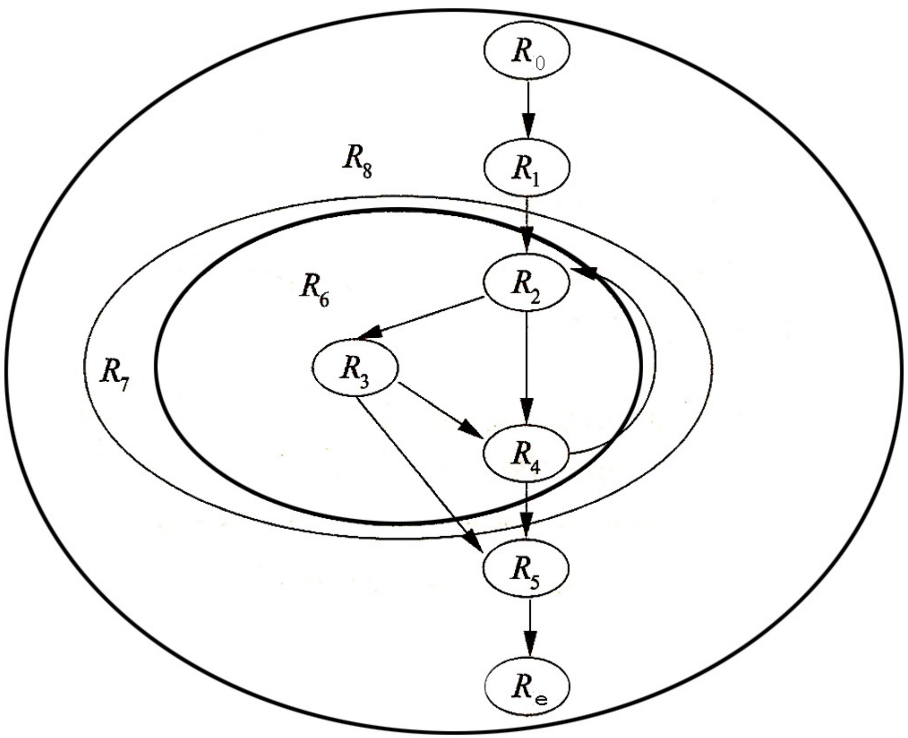


Последовательность преобразований

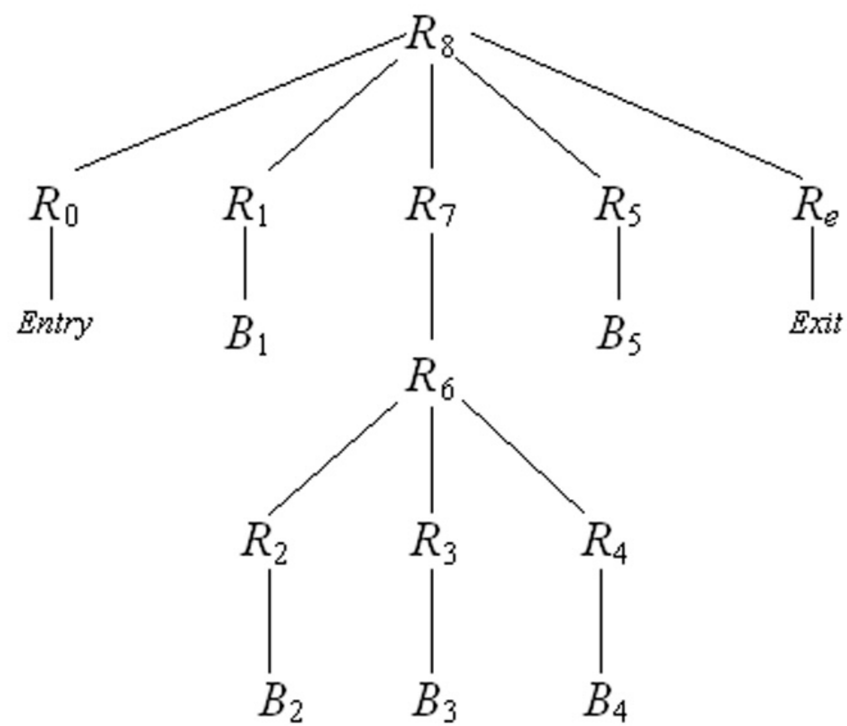
9.1 Структурный анализ графа потока управления

9.1.5. Построение иерархии областей

◊ **Пример**.



Вся иерархия

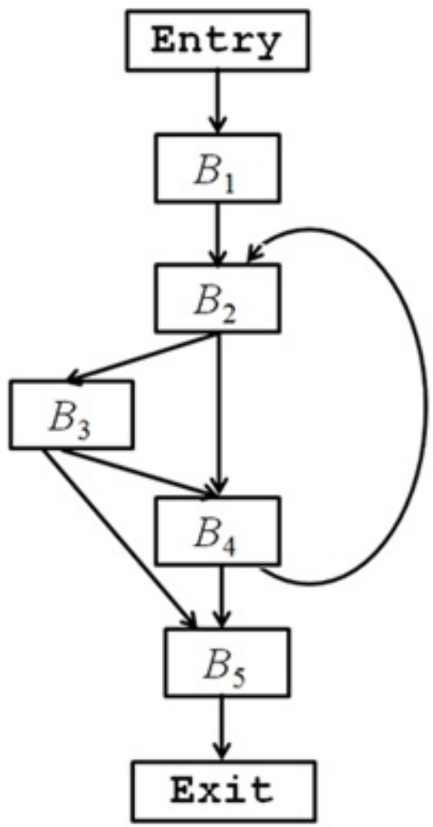


Дерево управления

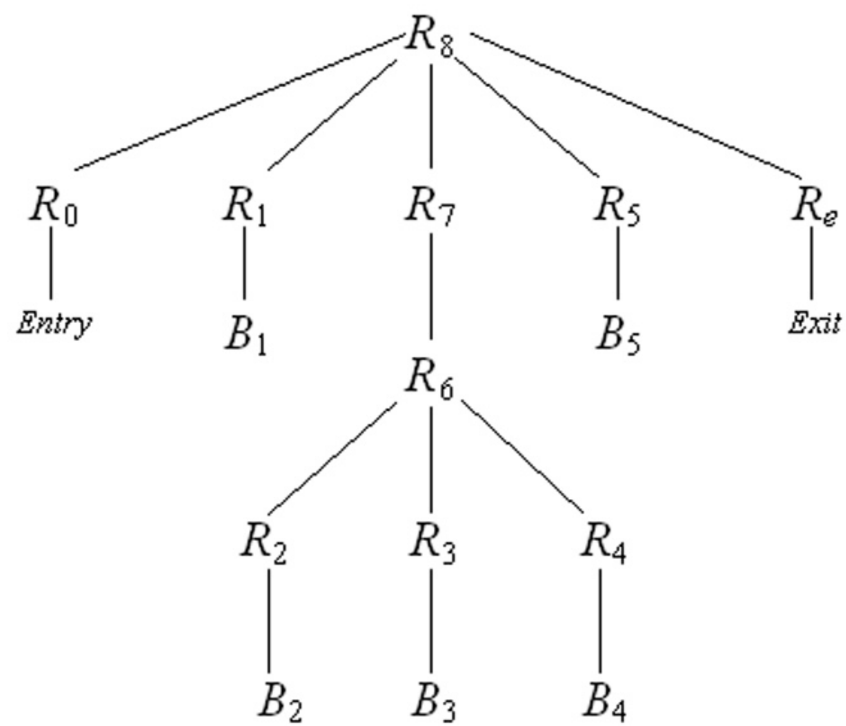
9.1 Структурный анализ графа потока управления

9.1.5. Построение иерархии областей

◊ **Пример**.



Исходный ГПУ



Дерево управления

9.1 Структурный анализ графа потока управления

9.1.6. Алгоритм построения восходящего порядка областей

◊ **Вход**: приводимый ГПУ G.

◊ **Выход**: список областей графа G, который может использоваться в задачах анализа потоков данных на основе областей.

◊ **Метод**: выполняем следующие действия.

1) Составляем список областей-листьев, состоящих из отдельных блоков в произвольном порядке.

2) Выбираем очередной естественный цикл L, такой, что все области, соответствующие естественным циклам, содержащимся в L, уже внесены в список. Сначала добавляем в список область-тело для L, а затем – область-цикл L.

3) Если весь граф G является естественным циклом, добавляем в конец списка область, состоящую из всего графа потока целиком.

9.1 Структурный анализ графа потока управления

9.1.7. Скорость сходимости итерационных алгоритмов

◊ Построение областей и дерева управления позволяет ускорить обход графа потока управления во время выполнения различных алгоритмов анализа потока данных.

◊ В алгоритмах анализа потока данных, в которых в качестве сбора используется объединение, удается, заменив каждый цикл узлом типа область-цикл, оставить только ациклические пути для распространения атрибутов.

Это позволяет сократить время выполнения соответствующего анализа потока данных.

9.2. Анализ потока данных на основе областей

9.2.1. Схема анализа потока данных на основе областей

- ◊ На каждом уровне иерархии областей:
  - Для каждой области R и для каждой подобласти $R' \in R$ вычисляется передаточная функция $f_{R,In[R']}$, суммирующая влияние всех возможных путей в R, ведущих от входа в R ко входу в R'.

9.2. *Анализ потока данных на основе областей*

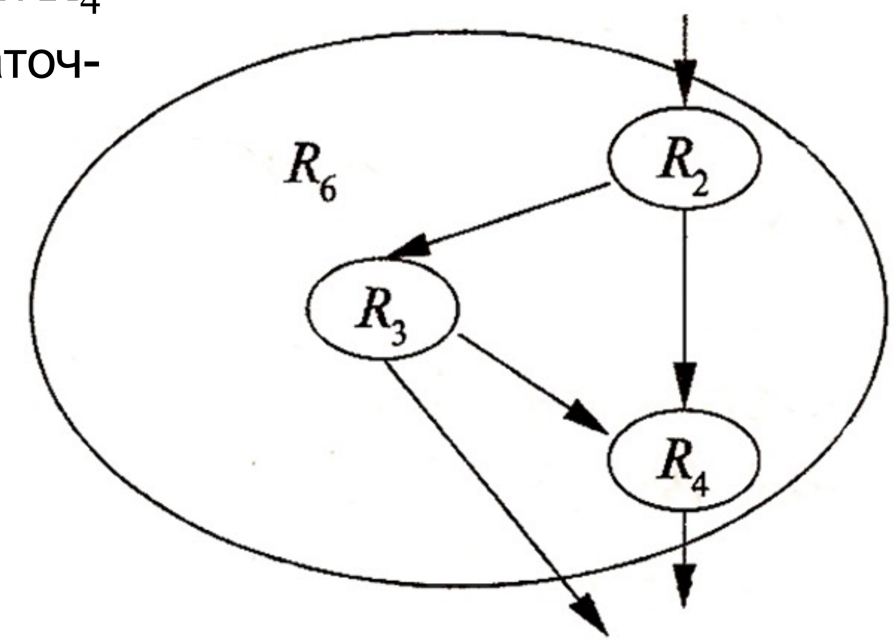
9.2.1. Схема анализа потока данных на основе областей

- На каждом уровне иерархии областей:
  - Для каждой области R и для каждой подобласти $R' \in R$ вычисляется передаточная функция $f_{R,In[R']}$, суммирующая влияние всех возможных путей в R, ведущих от входа в R ко входу в R'.
  - Например, для области R_6 и её подобластей R_2, R_3 и R_4 будут вычислены передаточные функции

$$f_{R_6,In[R_2]}$$

$$f_{R_6,In[R_3]}$$

$$f_{R_6,In[R_4]}$$



9.2. Анализ потока данных на основе областей

9.2.1.1. Направление анализа потока данных на основе областей

◊ Мы рассматриваем анализ только в прямом направлении (сверху вниз). Для задач потока данных в обратном направлении необходимы неочевидные изменения в алгоритме, и в данном курсе они подробно не рассматриваются.

◊ В самом простом случае можно предложить следующее решение для выполнения анализа в обратном направлении:

1) Построение областей выполняется на обратном графе потока управления;
2) Передаточные функции областей для анализа в обратном направлении строятся так же как и для прямого, но только на обратном графе;
3) Обратный граф потока управления обязательно должен быть *приводимый*, в противном случае данный метод неприменим. Это накладывает ограничение в виде отсутствия `break` в исходной программе (иначе в обратном графе будет вход внутрь региона).

9.2. Анализ потока данных на основе областей

9.2.1. Схема анализа потока данных на основе областей

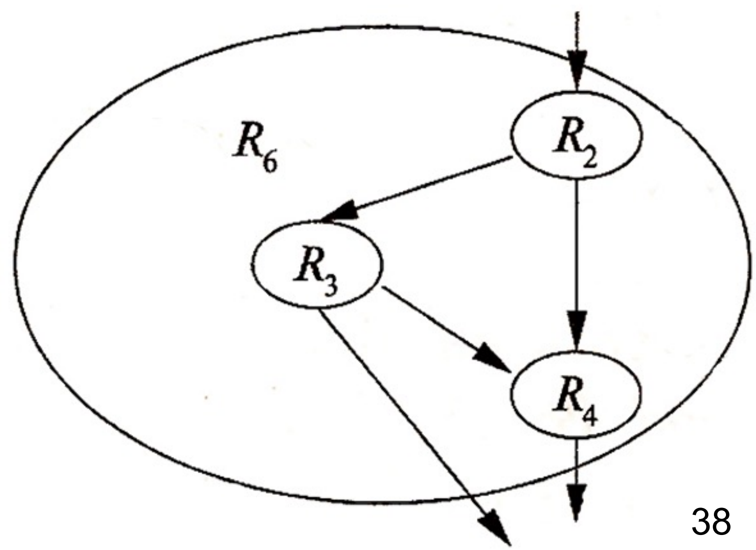
- На каждом уровне иерархии областей:
  - Область $R' \in R$ называется *выходной* подобластью области R, если у R' есть выходное ребро к некоторой области, не принадлежащей области R.
  - Для каждой выходной области $R' \in R$ вычисляется передаточная функция $f_{R,Out[R']}$, суммирующая влияние всех возможных путей в R, ведущих от входа в R к выходу из R'.

9.2. Анализ потока данных на основе областей

9.2.1. Схема анализа потока данных на основе областей

◊ На каждом уровне иерархии областей:

- Область $R' \in R$ называется *выходной* подобластью области R, если у R' есть выходное ребро к некоторой области, не принадлежащей области R.
- Для каждой выходной области $R' \in R$ вычисляется передаточная функция $f_{R,Out[R']}$, суммирующая влияние всех возможных путей в R, ведущих от входа в R к выходу из R'.
- Например, для области R_6 и её выходных подобластей R_3 и R_4 будут вычислены передаточные функции $f_{R_6,Out[R_3]}$ $f_{R_6,Out[R_4]}$



9.2. Анализ потока данных на основе областей

9.2.1. Схема анализа потока данных на основе областей

- ◊ **<u>Первый этап</u>. Построение передаточных функций всех областей (от листьев к корню дерева управления).**

  - Сначала обрабатываются области-листья (отдельные блоки):

    для каждой области-листа R, состоящей из блока B,

    $f_{R,In[B]} = I$ (тождественная функция)

    $f_{R,Out[B]} = f_B$ (передаточная функция блока B).

9.2. Анализ потока данных на основе областей

9.2.1. Схема анализа потока данных на основе областей

- ◊ **<u>Первый этап</u>. Построение передаточных функций всех областей (от листьев к корню дерева управления).**

  - Сначала обрабатываются области-листья (отдельные блоки):

    для каждой области-листа R, состоящей из блока B,

    $f_{R,In[B]} = I$ (тождественная функция)

    $f_{R,Out[B]} = f_B$ (передаточная функция блока B).

  - Перемещение вверх по иерархии:

    - ***R*** **– область-тело**: ребра, принадлежащие R, образуют ациклический граф на подобластях R, что позволяет при вычислении передаточных функций использовать топологический порядок областей.

    - ***R*** **– область-цикл**: учитывается только влияние обратных ребер, ведущих к заголовку R.

9.2. Анализ потока данных на основе областей

9.2.1. Схема анализа потока данных на основе областей

- **<u>Первый этап</u>. Построение передаточных функций всех областей (от листьев к корню дерева управления).**
  - Сначала обрабатываются области-листья (отдельные блоки).
  - Перемещение вверх по иерархии:
    - $\boldsymbol{R}$ **– область-тело**: ребра, принадлежащие R, образуют ациклический граф на подобластях R, что позволяет при вычислении передаточных функций использовать топологический порядок областей.
    - $\boldsymbol{R}$ **– область-цикл**: учитывается только влияние обратных ребер, ведущих к заголовку R.
  - В конце обработки достигается вершина иерархии и вычисляются передаточные функции области R_n, представляющей собой весь граф потока.

9.2. Анализ потока данных на основе областей

9.2.1. Схема анализа потока данных на основе областей

◊ **<u>Второй этап</u>. Анализ иерархии областей (от корня к листьям дерева управления).**

- Области просматриваются в обратном порядке, начиная с области R_n и далее, опускаясь вниз по иерархии. Для каждой области вычисляются значения потока данных на входе.
- Чтобы получить значения потока данных на входе $R \in R_n$ используется передаточная функция $f_{R_n, In[R]}$
- Вычисления повторяются, до тех пор, пока не будут достигнуты области-листья (базовые блоки).

9.2. Анализ потока данных на основе областей

9.2.2. Вычисление передаточных функций

- Для анализа потока данных с использованием итеративного алгоритма в базовых блоках потребовалась замкнутость структуры потока данных относительно операции *композиции* функций.
- Для анализа потока данных на основе областей к передаточным функциям применяется уже не только операция композиции, но еще две операции, позволяющие вычислять передаточные функции для областей по передаточным функциям для их подобластей:
  - операция *сбора* (для входов в компоненты подобластей) и
  - операция *замыкания* (для циклов).
- Для применимости указанных операций требуется, чтобы структура потока данных (множество передаточных функций) была замкнута относительно трех операций: композиции, сбора и замыкания.

9.2. Анализ потока данных на основе областей

9.2.2. Вычисление передаточных функций

◊ **1. Замкнутость относительно композиции**

- *Замкнутость* множества $\mathcal{F}$ *относительно композиции* означает, что композиция двух передаточных функций, принадлежащих множеству $\mathcal{F}$, является передаточной функцией, принадлежащей множеству $\mathcal{F}$.

- **Утверждение.** Множество $\mathcal{GK}$ передаточных функций вида *gen-kill*, замкнуто относительно композиции:

  если $f_1 \in \mathcal{GK}$ $f_2 \in \mathcal{GK}$, а $f = f_1 \circ f_2$, то и $f \in \mathcal{GK}$

  Доказательство.

$$\begin{aligned}
(f_2 \circ f_1)(x) &= gen_2 \cup ((gen_1 \cup (x - kill_1)) - kill_2) = \\
&= gen_2 \cup (gen_1 - kill_2)) \cup (x - (kill_1 \cup kill_2)) = \\
&= gen \cup (x - kill),
\end{aligned}$$

$$gen = gen_2 \cup (gen_1 - kill_2) \qquad kill = kill_1 \cup kill_2.$$

9.2.2. Вычисление передаточных функций

◊ 2. Операция сбора

- Операция *сбора* $\wedge_{\mathcal{F}}$ на множестве передаточных функций $\mathcal{F}$ определяется с помощью операции $\wedge$ сбора значений потока данных следующим образом:

$$(f_1 \wedge_{\mathcal{F}} f_2)\ (x) = f_1(x) \wedge f_2(x),$$

- Множество передаточных функций $\mathcal{F}$ *замкнуто относительно операции сбора* $\wedge_{\mathcal{F}}$, если для любых двух передаточных функций $f_1 \in \mathcal{F}$ и $f_2 \in \mathcal{F}$, их сбор $f = f_1 \wedge_{\mathcal{F}} f_2 \in \mathcal{F}$.

- **Замечание.** Множество передаточных функций $\mathcal{F}$, замкнутое относительно операции сбора $\wedge_{\mathcal{F}}$ является полурешеткой с операцией сбора $\wedge_{\mathcal{F}}$.

9.2. Анализ потока данных на основе областей

9.2.2. Вычисление передаточных функций

◊ **3. Замкнутость относительно операции сбора**

◊ **Утверждение**. Множество $\mathcal{GK}$ передаточных функций вида *gen-kill*, замкнуто относительно операции сбора $\wedge_{\mathcal{GK}}$:

Доказательство:

$$
\begin{aligned}
(f_1 \wedge_{\mathcal{GK}} f_2)(x) \quad &= f_1(x) \wedge f_2(x) = \\
&= gen_1 \cup (x - kill_1) \cup gen_2 \cup (x - kill_2) = \\
&= (gen_1 \cup gen_2) \cup (x - (kill_1 \cap kill_2)) = \\
&= gen \cup (x - kill),
\end{aligned}
$$

$$gen = gen_1 \cup gen_2, kill = kill_1 \cap kill_2.$$

9.2. Анализ потока данных на основе областей

9.2.2. Вычисление передаточных функций

◊ **4. Операция замыкания**

- Пусть f – передаточная функция тела цикла.

  Тогда двум итерациям цикла будет соответствовать функция

  $$f^2(x) = f \wedge_{\mathcal{F}} f,$$

  а n итерациям цикла – функция

  $$f^n = f^{n-1} \wedge_{\mathcal{F}} f.$$

  Если количество итераций цикла неизвестно, его передаточная функция представляется как *замыкание* f.

- Пусть I – тождественная функция. Тогда $f^0 = I$, $f^1 = f$

- *Замыканием* передаточной функции $f \in \mathcal{F}$ называется функция f^*, определяемая формулой

  $$f^* = I \wedge_{\mathcal{F}} \bigwedge_{n>0} f^n$$

9.2. Анализ потока данных на основе областей

9.2.2. Вычисление передаточных функций

◊ **5. Замкнутость относительно операции замыкания**

- Множество передаточных функций $\mathcal{F}$ *замкнуто относительно операции замыкания*.

- **Утверждение**. Множество $\mathcal{GK}$ передаточных функций вида *gen-kill* замкнуто относительно операции замыкания: если $\forall f \in \mathcal{GK}$ то и $f^* \in \mathcal{GK}$.

  Доказательство:

$$f^2(x) = f(f(x)) = gen \cup (gen \cup (x - kill) - kill) =$$
$$= (gen \cup gen) \cup (x - (kill \cup kill)) = gen \cup (x - kill).$$

$f^n(x) = gen \cup (x - kill)$ (по индукции)

$$f^*(x) = I \wedge f^1(x) \wedge f^2(x) \wedge \ldots = x \cup (gen \cup (x - kill)) =$$
$$= gen \cup (x \cup (x - kill)) = gen \cup x$$

$$(x - kill) \subseteq x$$

9.2.2. Вычисление передаточных функций

- **5. Замкнутость относительно операции замыкания**
  - Итак, если $f(x) = (gen \cup (x - kill))$, то $f^*(x) = gen \cup x$

    Следовательно, для $\forall f \in \mathcal{GK}$

$$gen_{f^*} = gen_f$$

$$kill_{f^*} = \varnothing$$

Таким образом, циклы не влияют на анализ достигающих определений

9.2.2. Вычисление передаточных функций

◊ **Итак, формулы для функций вида $\mathcal{GK}$:**

Если $f_1(x) = gen_1 \cup (x - kill_1)$ и $f_2(x) = gen_2 \cup (x - kill_2)$, то

(1) **Композиция:**

$$(f_2 \circ f_1)(x) = gen^\circ \cup (x - kill^\circ),$$

где $gen^\circ = gen_2 \cup (gen_1 - kill_2)$, $kill^\circ = kill_1 \cup kill_2$.

(2) **Сбор:**

$$(f_1 \wedge f_2)(x) = gen^\wedge \cup (x - kill^\wedge),$$

где $gen^\wedge = gen_1 \cup gen_2$, $kill^\wedge = kill_1 \cap kill_2$

Замыкание:

Если $f(x) = (gen \cup (x - kill))$, то $f^*(x) = gen^* \cup (x - kill^*)$,

где $gen^* = gen$, $kill^* = \varnothing$

9.2. Анализ потока данных на основе областей

9.2.3. Алгоритм анализа на основе областей

◊ **Вход**: структура потока данных, замкнутая относительно операций композиции, сбора и замыкания, приводимый граф потока управления G.

◊ **Выход**: значения потока данных $In[B]$ для каждого блока $B \in G$.

◊ **Метод**: выполнить следующие действия:

1) С помощью алгоритма 9.1.3 определить области и их топологический порядок (снизу-вверх)

2) *Восходящий просмотр* для вычисления передаточных функций областей $R_1, R_2, ..., R_n$: (R_n – область самого верхнего уровня).

2а) R – область-лист, соответствующая блоку B:

$$f_{R,In[B]} = I,\ f_{R,Out[B]} = f_B.$$

9.2. Анализ потока данных на основе областей

9.2.3. Алгоритм анализа на основе областей

◊ **Метод**: выполнить следующие действия:

2) *Восходящий просмотр* для вычисления передаточных функций областей $R_1, R_2, \ldots, R_n$:

2b) R – область-тело:

for each $S \in R$ (в топологическом порядке) **{**

$$f_{R,In}[S] = \bigwedge_{B \in Pred(S)} f_{R,Out}[B]$$

/* Если S – заголовок области R, то сбор по «ничему», т.е. $f_{R,In[S]} = I$, если у S один предок – то без сбора, просто $f_{R,In[S]} = f_{S,Out[Pred(S)]}$. */

for each (выходной блок $B \in S$) $f_{R,Out[B]} = f_{S,Out[B]} \circ f_{R,In[S]}$;

}

9.2.3. Алгоритм анализа на основе областей

◊ **Метод**: выполнить следующие действия:

2) *Восходящий просмотр* для вычисления передаточных функций областей $R_1, R_2, \ldots, R_n$:

2c) R – область-цикл:

```
// Пусть S – область тела цикла, непосредственно вложенная в R,
// т.е. S представляет собой R без обратных ребер из R в заголовок R
```

$$f_{R,In[S]} = \left(\bigwedge_{B \in Pred(S)} f_{S,Out[B]} \right)^* ;$$

```
// Сбор выполняется по предшественникам B заголовка S в R,
// т.е. по всем обратным ребрам цикла.
```

for each (выходной блок $B \in S$)

$$f_{R,Out[B]} = f_{S,Out[B]} \circ f_{R,In[S]};$$

9.2. Анализ потока данных на основе областей

9.2.3. Алгоритм анализа на основе областей

◊ **Метод**: выполнить следующие действия:

2) *Восходящий просмотр* для вычисления передаточных функций областей $R_1, R_2, \ldots, R_n$:

2c) R – область-цикл:

```
// Пусть S – область тела цикла, непосредственно вложенная в R,
// т.е. S представляет собой R без обратных ребер из R в заголовок R
```

I) $$f_{R,In[S]} = \left(\bigwedge_{B \in Pred(S)} f_{S,Out[B]} \right)^{*};$$

```
// Сбор выполняется по предшественникам B заголовка S в R,
// т.е. по всем обратным ребрам цикла.
```

II) **for each** (выходной блок $B \in S$)

$$f_{R,Out[B]} = f_{S,Out[B]} \circ f_{R,In[S]};$$

```
// После того, как вычислили In у R, на шаге (I),
// нужно получить еще Out'ы для области-цикла
```

9.2. Анализ потока данных на основе областей

9.2.3. Алгоритм анализа на основе областей

◊ **Метод**: выполнить следующие действия:

3) *Нисходящий просмотр* для вычисления значений $In[R]$ в начале каждой области:

$In[R_n] = Out[Entry]$;

for each region R (в нисходящем порядке)

$In[R] = f_{R',In[R]}(In[R'])$,

где R' – область, непосредственно охватывающая область R

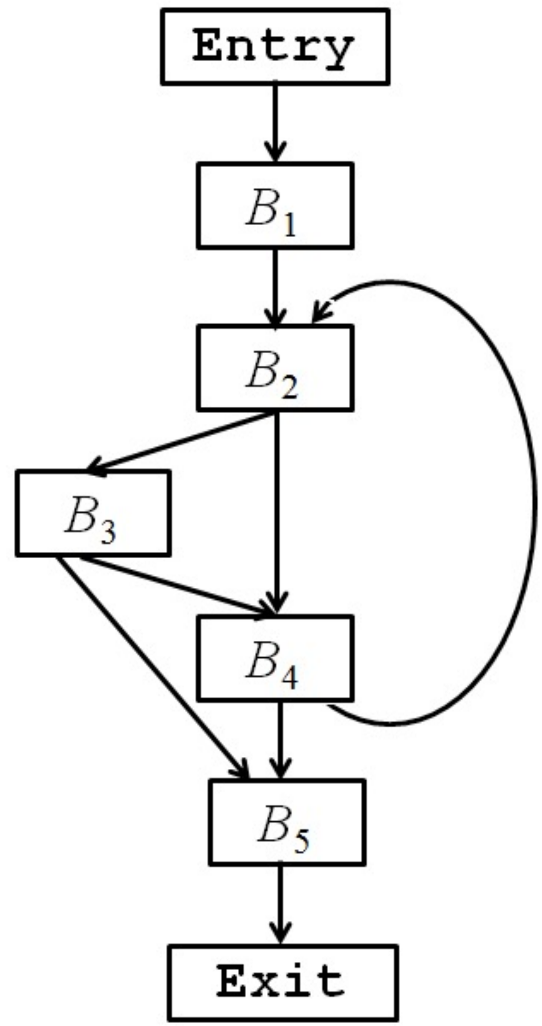
9.2. Анализ потока данных на основе областей

9.2.4. Пример: применение алгоритма 9.2.3

◊ Применим алгоритм 9.2.3 для поиска достигающих определений с помощью анализа на основе областей в рассматриваемой программе.

B_1	`i ← - m, 1`
	`j ← n`
	`a ← u1`
		d_1
		d_2
		d_3

B_2	`i ← + i, 1`	d_4
B_3	`a ← u2`	d_5
B_4	`j ← u3`	d_6
B_5	. . . .	

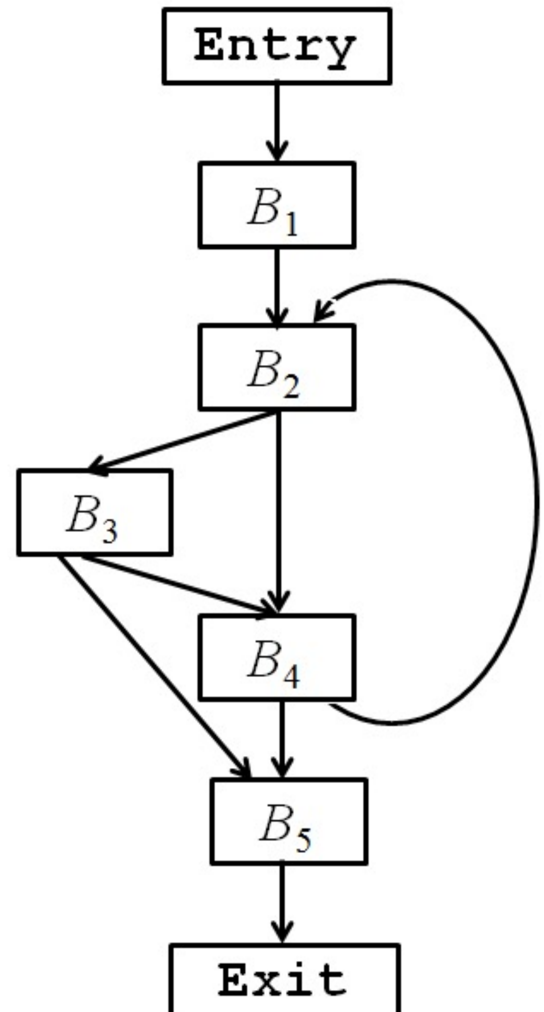


9.2. Анализ потока данных на основе областей

9.2.4. Пример: применение алгоритма 9.2.3

- **Замечание**: поиск достигающих определений обычным способом (с помощью итерационного алгоритма, который изучался в прошлом семестре) дает следующие результаты

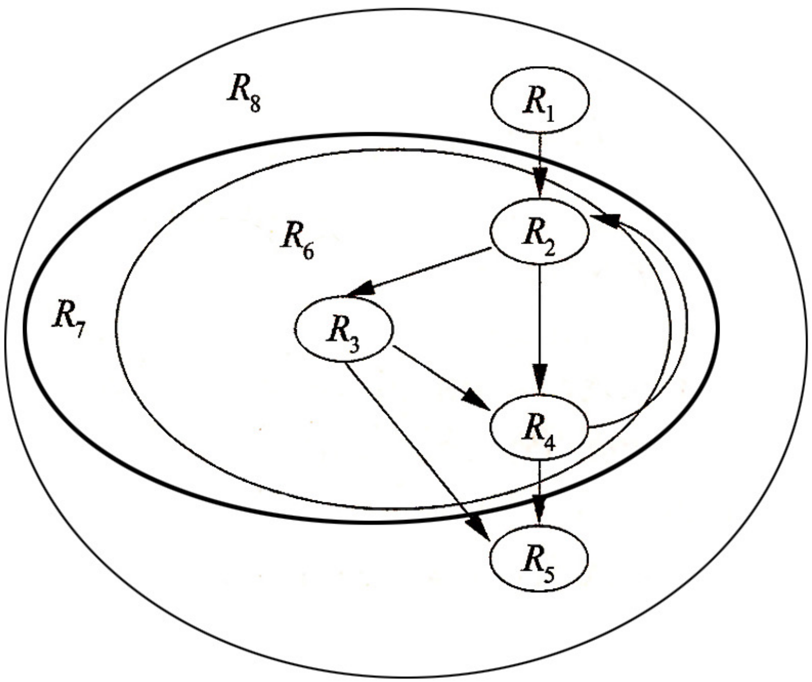
$$
\begin{aligned}
In[B_1] &= \varnothing \\
In[B_2] &= \{d_1, d_2, d_3, d_4, d_5, d_6\} \\
In[B_3] &= \{d_2, d_3, d_4, d_5, d_6\} \\
In[B_4] &= \{d_2, d_3, d_4, d_5, d_6\} \\
In[B_5] &= \{d_2, d_3, d_4, d_5, d_6\}
\end{aligned}
$$

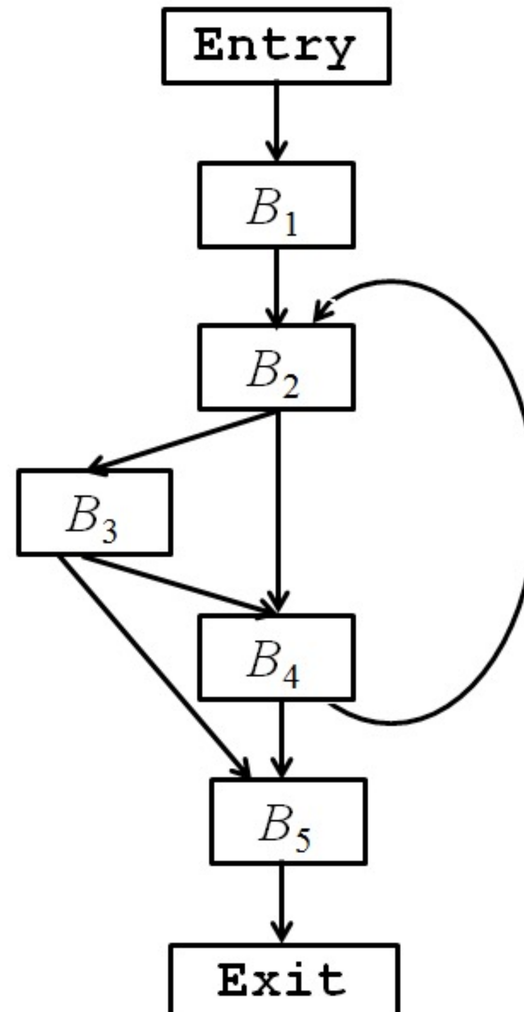


9.2. Анализ потока данных на основе областей

9.2.4. Пример: применение алгоритма 9.2.3

- Применим алгоритм 9.2.3 для поиска достигающих определений с помощью анализа на основе областей в рассматриваемой программе.
- **Шаг 1**: с помощью алгоритма 9.1.3 определим области и пронумеруем их в восходящем топологическом порядке (слайды 20 – 27): получим области $R_1, R_2, R_3, R_4, R_5, R_6, R_7, R_8$



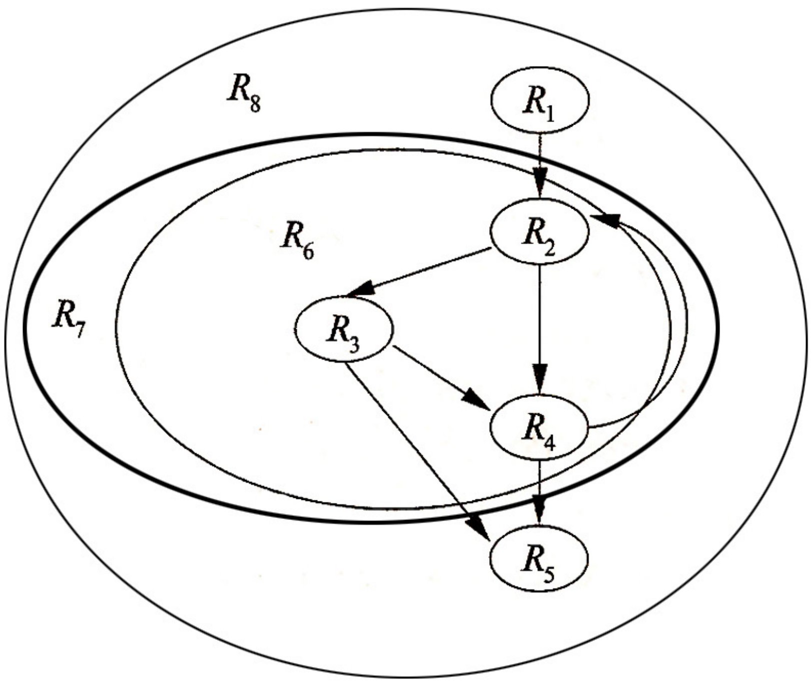


9.2. Анализ потока данных на основе областей

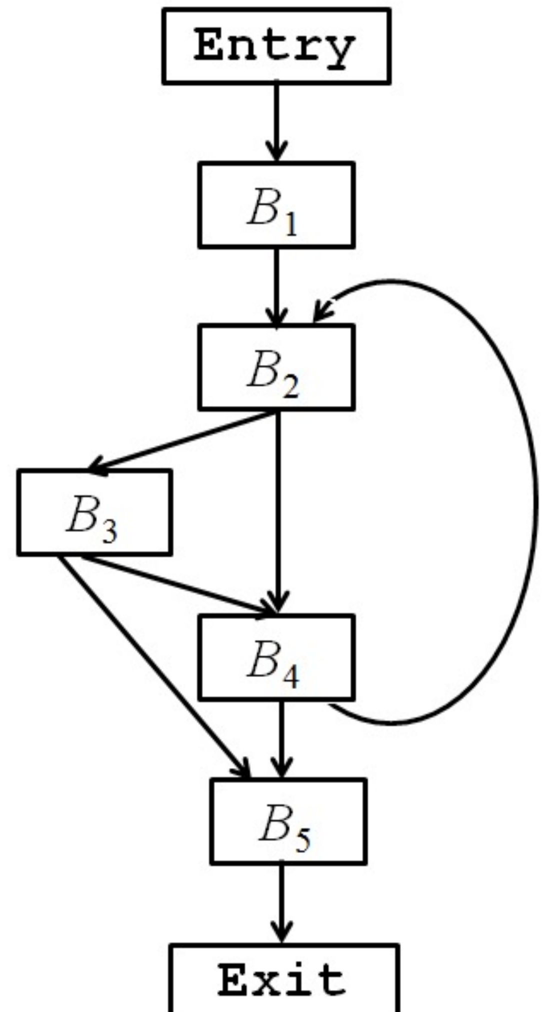
9.2.4. Пример: применение алгоритма 9.2.3

◊ Применим алгоритм 9.2.3 для поиска достигающих определений с помощью анализа на основе областей в рассматриваемой программе.

◊ **Шаг 1**: с помощью алгоритма 9.1.3 определим области и пронумеруем их в восходящем топологическом порядке (слайды 20 – 27): получим области $R_1, R_2, R_3, R_4, R_5, R_6, R_7, R_8$



Области R_1, R_2, R_3, R_4, R_5 являются областями-листьями и соответствуют базовым блокам B_1, B_2, B_3, B_4, B_5



9.2. Анализ потока данных на основе областей

9.2.4. Пример: применение алгоритма 9.2.3

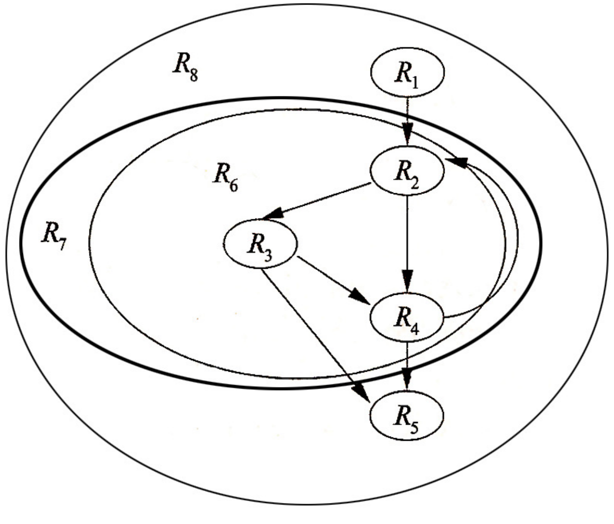
- **Шаг 2**: *Восходящий просмотр* для вычисления передаточных функций областей $R_1, R_2, R_3, R_4, R_5, R_6, R_7, R_8$

Шаг 2а): вычисление передаточных функций областей-листьев R_1, R_2, R_3, R_4, R_5

$$f_{R_i, In[B_i]}(x) = I,$$

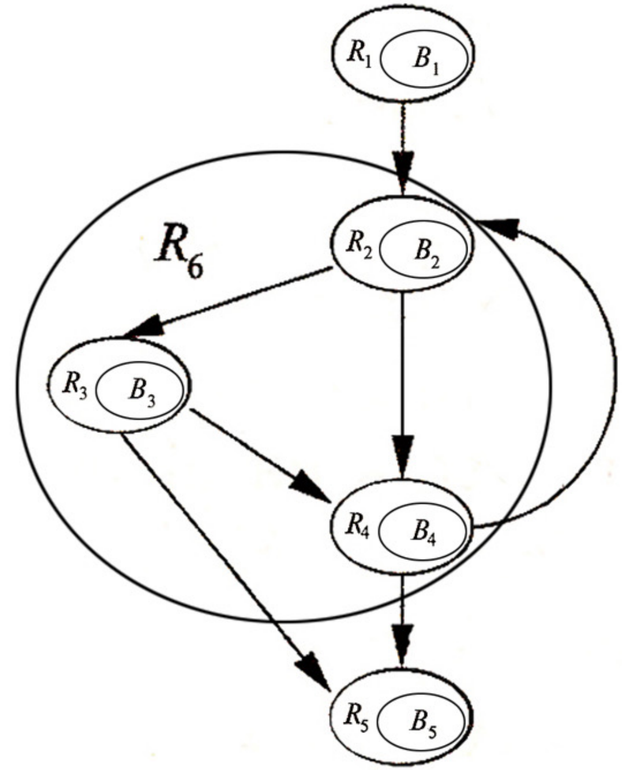
$$f_{R_i, Out[B_i]}(x) = f_{B_i}(x) = gen_{B_i} \cup (x - kill_{B_i})$$

B	B_1	B_2	B_3	B_4	B_5
gen_B	$\{d_1,d_2,d_3\}$	$\{d_4\}$	$\{d_5\}$	$\{d_6\}$	$\varnothing$
$kill_B$	$\{d_4,d_5,d_6\}$	$\{d_1\}$	$\{d_3\}$	$\{d_2\}$	$\varnothing$



9.2. Анализ потока данных на основе областей

9.2.4. Пример: применение алгоритма 9.2.3



- **Шаг 2b):** вычисление передаточной функции области-тела R_6

  Область R_6 состоит из подобластей R_2, R_3 и R_4. Эти подобласти обрабатываются в топологическом порядке, построенном на шаге 1: R_2, R_3, R_4.

  Подобласть R_2: У R_2 нет предшественников в пределах R_6, так как обратное ребро $R_4 \to R_2$ не принадлежит R_6. Следовательно

$$f_{R_6,In[R_2]} = f_{R_2,In[B_2]} = I,$$

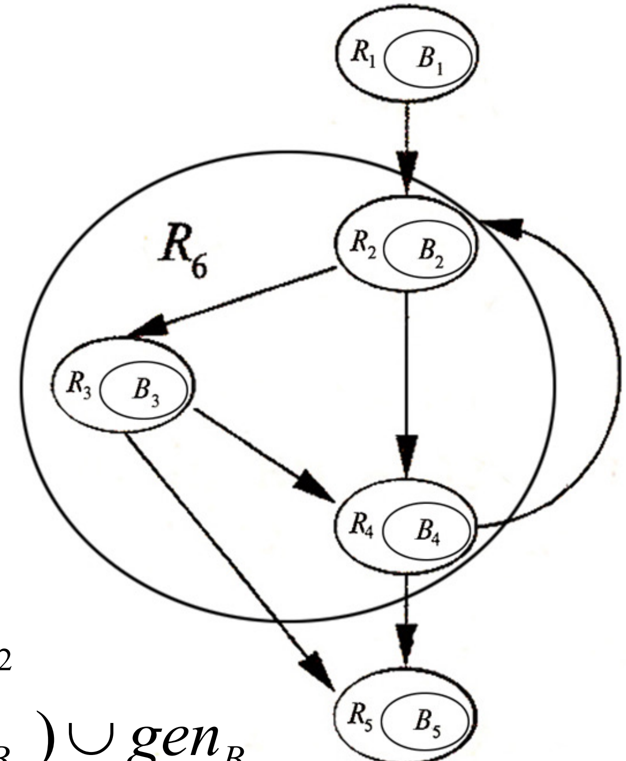
$$f_{R_6,Out[B_2]} = f_{R_2,Out[B_2]} \circ f_{R_6,In[R_2]} = f_{B_2} \circ I = f_{B_2}$$

$$gen_{R_6,In[R_2]} = \varnothing, kill_{R_6,In[R_2]} = \varnothing$$

$$gen_{R_6,Out[R_2]} = gen_{B_2} = \{d_4\}, kill_{R_6,Out[R_2]} = kill_{B_2} = \{d_1\}$$

9.2. Анализ потока данных на основе областей

9.2.4. Пример: применение алгоритма 9.2.3



- **Шаг 2b):** вычисление передаточной функции области-тела R_6

Подобласть R_3: У R_3 в пределах R_6 один предшественник R_2. Следовательно

$$f_{R_6,In[R_3]} = f_{R_6,Out[B_2]} = f_{B_2}$$

$$f_{R_6,Out[B_3]} = f_{R_3,Out[B_3]} \circ f_{R_6,In[R_3]} = f_{B_3} \circ f_{B_2}$$

$$f_{R_6,In[R_3]}(x) = f_{R_6,Out[B_2]}(x) = f_{B_2}(x) = (x - kill_{B_2}) \cup gen_{B_2}$$

$$f_{R_6,Out[B_3]}(x) = f_{R_3,Out[B_3]} \circ f_{R_6,In[R_3]}(x) = (f_{B_3} \circ f_{B_2})(x) =$$

$$(gen_{B_3} \cup (gen_{B_2} - kill_{B_3})) \cup (x - (kill_{B_3} \cup kill_{B_2}))$$

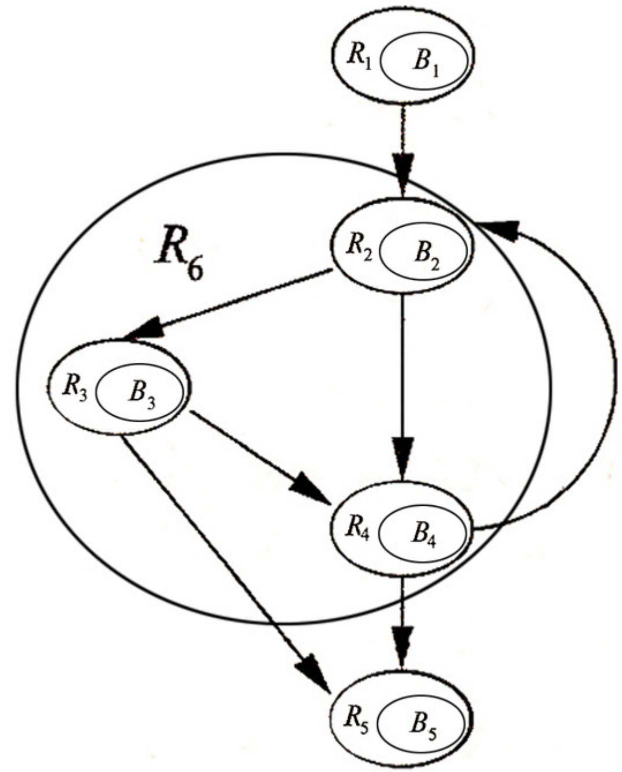
$$gen_{R_6,In[B_3]} = gen_{B_2} = \{d_4\}, kill_{R_6,In[B_3]} = kill_{B_2} = \{d_1\}$$

$$gen_{R_6,Out[B_3]} = gen_{B_3} \cup (gen_{B_2} - kill_{B_3}) = \{d_5\} \cup (\{d_4\} - \{d_3\}) = \{d_4, d_5\}$$

$$kill_{R_6,Out[B_3]} = kill_{B_3} \cup kill_{B_2} = \{d_3\} \cup \{d_1\} = \{d_1, d_3\}$$

9.2. Анализ потока данных на основе областей

9.2.4. Пример: применение алгоритма 9.2.3



- **Шаг 2b):** вычисление передаточной функции области-тела R_6

  Подобласть R_4: У R_4 в пределах R_6 два предшественника R_2 и R_3. Следовательно

$$f_{R_6,In[R_4]} = f_{R_6,Out[B_2]} \wedge f_{R_6,Out[B_3]} =$$

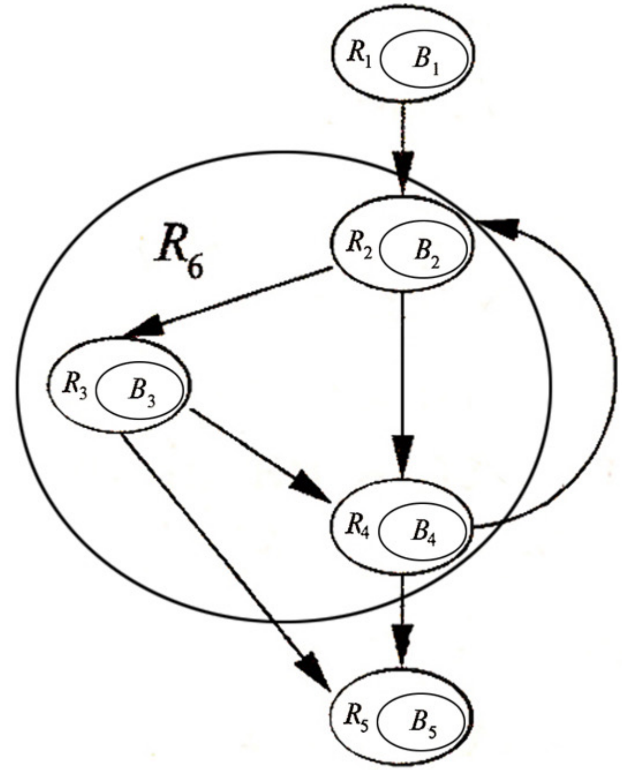
$$= f_{B_2} \wedge f_{R_6,Out[B_3]}$$

$$f_{R_6,Out[B_4]} = f_{R_4,Out[B_4]} \circ f_{R_6,In[R_4]} =$$

$$= f_{B_4} \circ f_{R_6,In[R_4]}$$

9.2. Анализ потока данных на основе областей

9.2.4. Пример: применение алгоритма 9.2.3



- **Шаг 2b):** вычисление передаточной функции области-тела R_6

  Подобласть R_4

$$f_{R_6,\,In[R4]}(x) = (f_{R_6,Out[B_2]} \wedge f_{R_6,Out[B_3]})(x) = \\ = \left(f_{B_2} \wedge f_{R_6,\,Out[B_3]}\right)(x)$$

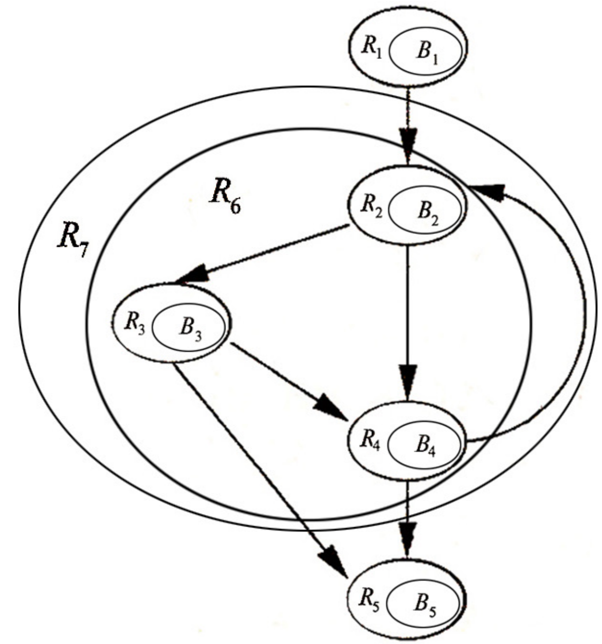
$$f_{R_6,\,Out[B4]}(x) = (f_{R_4,Out[B_4]} \circ f_{R_6,In[R_4]})(x) = \\ = (f_{B_4} \circ f_{R_6,\,In\,[R_4]})(x)$$

$$gen_{R_6,In[R_4]} = gen_{B_2} \cup gen_{R_6,Out[B_3]} = \{d_4\} \cup \{d_4, d_5\} = \{d_4, d_5\}$$
$$kill_{R_6,In[R_4]} = kill_{B_2} \cap kill_{R_6,Out[B_3]} = \{d_1\} \cap \{d_1, d_3\} = \{d_1\}$$

$$gen_{R_6,Out[R_4]} = gen_{B_4} \cup (gen_{R_6,In[R_4]} - kill_{B_4}) \\ = \{d_6\} \cup (\{d_4, d_5\} - \{d_2\} = \{d_4, d_5, d_6\}$$
$$kill_{R_6,Out[R_4]} = kill_{R_6,In[R_4]} \cup kill_{B_4} = \{d_1\} \cup \{d_2\} = \{d_1, d_2\}$$

9.2. Анализ потока данных на основе областей

9.2.4. Пример: применение алгоритма 9.2.3



- **Шаг 2c):** вычисление передаточной функции области-цикла R_7

  Область R_7 содержит только одну подобласть R_6 (тело цикла). Обратному ребру $B_4 \to B_2$ к заголовку R_6 соответствует передаточная функция

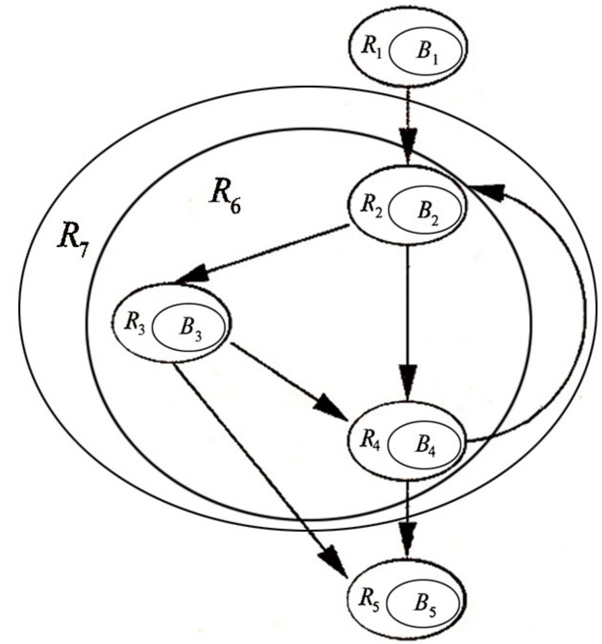
$$f_{R_7,In[R_6]} = f^{*}_{R_6,Out[B_4]}$$

Из области R_7 имеется два выхода – базовые блоки B_3 и B_4**.** Поэтому для получения соответствующих передаточных функций R_7 нужно вычислить композиции $f_{R_7,In[R_6]}$ с $f_{R_6,Out[B_3]}$ и с $f_{R_6,Out[B_4]}$:

$$f_{R_7,Out[B_4]} = f_{R_6,Out[B_4]} \circ f_{R_7,In[R_6]}$$

$$f_{R_7,Out[B_3]} = f_{R_6,Out[B_3]} \circ f_{R_7,In[R_6]}$$

9.2.4. Пример: применение алгоритма 9.2.3



- **Шаг 2c):** вычисление передаточной функции области-цикла R_7

  (a) На входе

  Передаточная функция:

$$f_{R_7,In[R_6]}(x) = f^{*}_{R_6,Out[B_4]}(x)$$

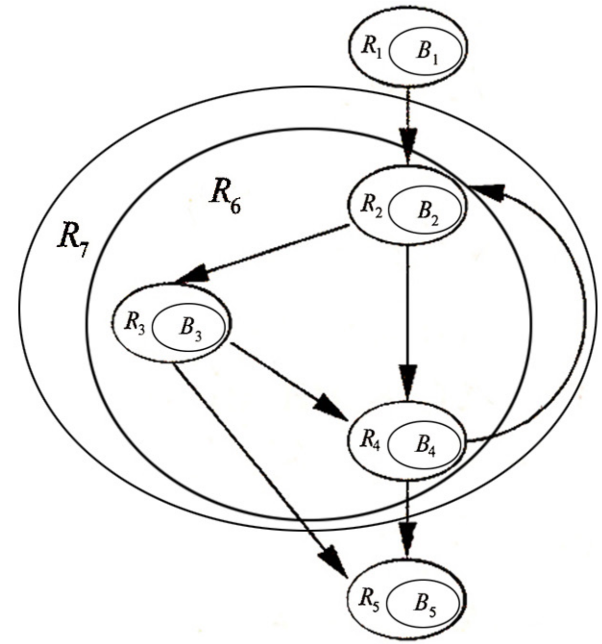
  Множества gen и $kill$

$$gen_{R_7,In[R_6]} = gen_{R_6,Out[B_4]} = \{d_4, d_5, d_6\}$$

$$kill_{R_7,In[R_6]} = \varnothing$$

9.2. Анализ потока данных на основе областей

9.2.4. Пример: применение алгоритма 9.2.3



- **Шаг 2c):** вычисление передаточной функции области-цикла R_7

  (b) На выходах

  Передаточные функции:

$$f_{R_7,Out[B_4]}(x) = (f_{R_6,Out[B_4]} \circ f_{R_7,In[R_6]})(x)$$

$$f_{R_7,Out[B_3]}(x) = (f_{R_6,Out[B_3]} \circ f_{R_7,In[R_6]})(x)$$

  Множества gen и $kill$

$$\begin{aligned} gen_{R_7,Out[B_4]} &= gen_{R_6,Out[B_4]} \cup (gen_{R_7,In[R_6]} - kill_{R_6,Out[B_4]}) \\ &= \{d_4, d_5, d_6\} \cup (\{d_4, d_5, d_6\} - \{d_1, d_2\}) = \{d_4, d_5, d_6\} \end{aligned}$$

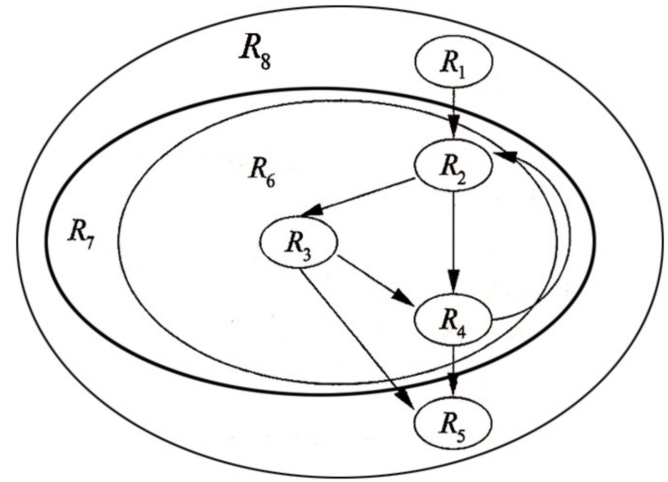
$$kill_{R_7,Out[B_4]} = kill_{R_7,In[R_6]} \cup kill_{R_6,Out[B_4]} = \emptyset \cup \{d_1, d_2\} = \{d_1, d_2\}$$

$$\begin{aligned} gen_{R_7,Out[B_3]} &= gen_{R_6,Out[B_3]} \cup (gen_{R_7,In[R_6]} - kill_{R_6,Out[B_3]}) \\ &= \{d_4, d_5\} \cup (\{d_4, d_5, d_6\} - \{d_1, d_3\}) = \{d_4, d_5, d_6\} \end{aligned}$$

$$kill_{R_7,Out[B_3]} = kill_{R_7,In[R_6]} \cup kill_{R_6,Out[B_3]} = \emptyset \cup \{d_1, d_3\} = \{d_1, d_3\}$$

9.2. Анализ потока данных на основе областей

9.2.4. Пример: применение алгоритма 9.2.3



- **Шаг 2d):** вычисление передаточной функции области-тела R_8

Подобластями области R_8 (весь граф потока) являются (в топологическом порядке) R_1, R_7 и R_5.

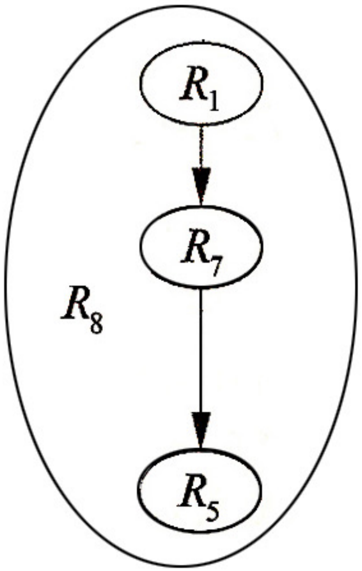
Передаточные функции:

1) для R_1:

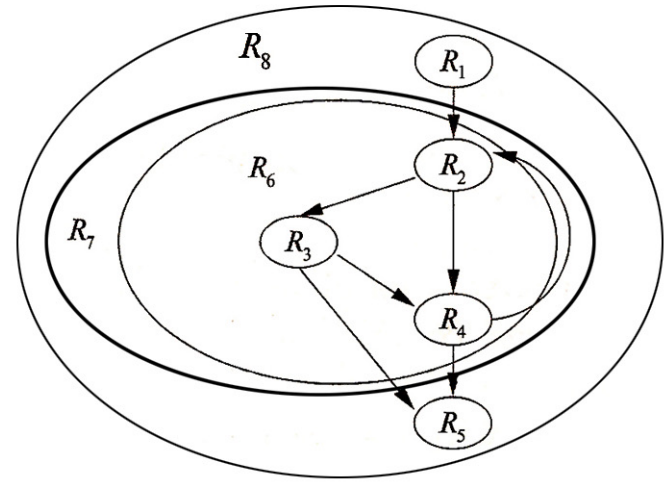
$$f_{R_8,In[R_1]}(x) = I(x) = x$$

$$f_{R_8,Out[B_1]}(x) = f_{B_1}(x)$$

$$gen_{R_8,In[R_1]} = kill_{R_8,In[R_1]} = \varnothing$$



9.2.4. Пример: применение алгоритма 9.2.3



- **Шаг 2d):** вычисление передаточной функции области-цикла R_8

2) для R_7:

Заголовок R_7 (блок B_2) имеет единственного предшественника, B_1.

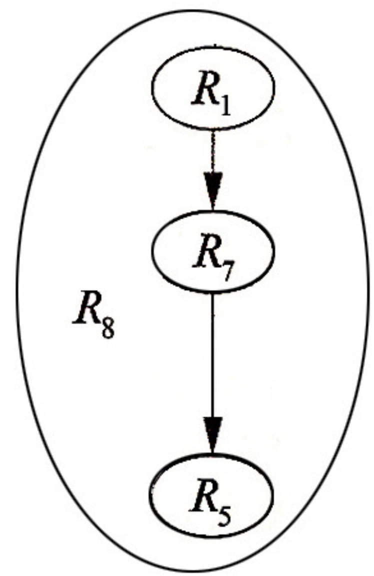
У R_7 два выходных ребра (в блоках B_3 и B_4).

Передаточные функции:

$$f_{R_8,In[R_7]} = f_{R_8,Out[B_1]} = f_{R_1,Out[B_1]} = f_{B_1}$$

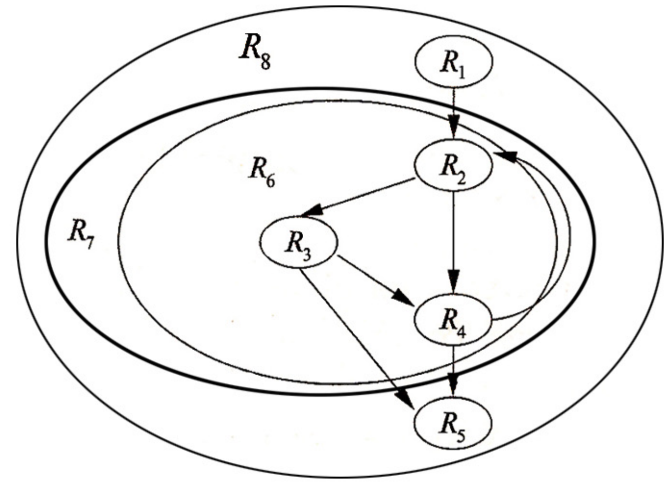
$$f_{R_8,Out[B_3]} = f_{R_7,Out[B_3]} \circ f_{R_8,In[R_7]} = f_{R_7,Out[B_3]} \circ f_{B_1}$$

$$f_{R_8,Out[B_4]} = f_{R_7,Out[B_4]} \circ f_{R_8,In[R_7]} = f_{R_7,Out[B_4]} \circ f_{B_1}$$



9.2. Анализ потока данных на основе областей

9.2.4. Пример: применение алгоритма 9.2.3



- **Шаг 2d):** вычисление передаточной функции области-цикла R_8

2) для R_7:

Множества gen и $kill$

$$gen_{R_8,In[R_7]} = gen_{B_1} = \{d_1, d_2, d_3\}$$

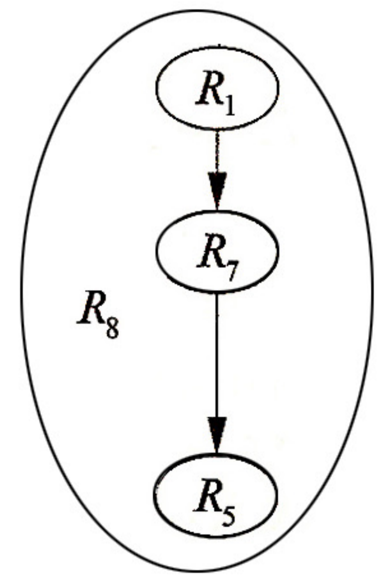
$$kill_{R_8,In[R_7]} = kill_{B_1} = \{d_4, d_5, d_6\}$$

$$gen_{R_8,Out[B_3]} = gen_{R_7,Out[B_3]} \cup (gen_{B_1} - kill_{R_7,Out[B_3]}) =$$

$$= \{d_4, d_5, d_6\} \cup (\{d_1, d_2, d_3\} - \{d_1, d_3\}) = \{d_2, d_4, d_5, d_6\}$$

$$kill_{R_8,Out[B_3]} = kill_{R_7,Out[B_3]} \cup kill_{B_1} =$$

$$= \{d_1, d_3\} \cup \{d_4, d_5, d_6\} = \{d_1, d_3, d_4, d_5, d_6\}$$

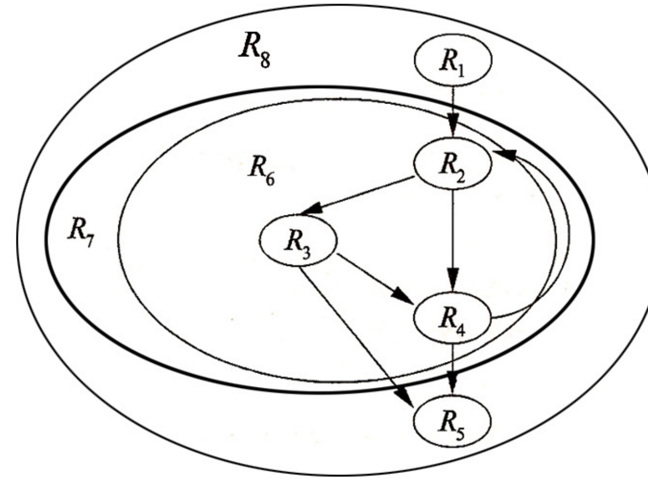


9.2. Анализ потока данных на основе областей
9.2.4. Пример: применение алгоритма 9.2.3

◊ **Шаг 2d):** вычисление передаточной функции области-цикла R_8

3) для R_5: У заголовка R_5 (блок B_5) два предшественника (B_3 и B_4), причем $f_{B_5} = I$.

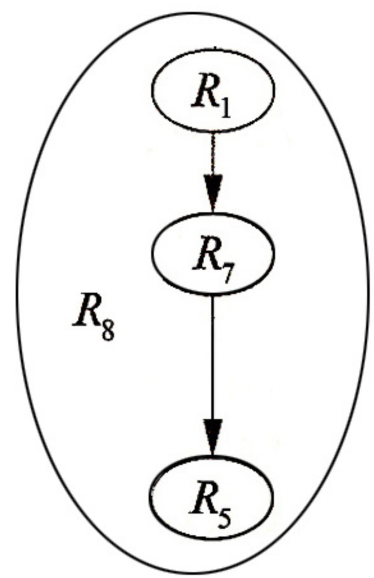
Передаточные функции для R_5:



$$f_{R_8,In[B_5]}(x) = \left(f_{R_8,Out[B_3]} \wedge f_{R_8,Out[B_4]}\right)(x)$$

$$f_{R_8,Out[B_5]}(x) = f_{R_8,In[B_5]}(x)$$

Множества gen и $kill$ (см 9.2.2.3)



$$gen_{R_8,In[B_5]} = gen_{R_8,Out[B_3]} \cup gen_{R_8,Out[B_4]} =$$

$$= \{d_2, d_4, d_5, d_6\} \cup \{d_3, d_4, d_5, d_6\} = \{d_2, d_3, d_4, d_5, d_6\}$$

$$kill_{R_8,In[B_5]} = kill_{R_8,Out[B_3]} \cap kill_{R_8,Out[B_4]} =$$

$$= \{d_1, d_3, d_4, d_5, d_6\} \cap \{d_1, d_2, d_4, d_5, d_6\} = \{d_1, d_4, d_5, d_6\}$$

$$gen_{R_8,Out[B_5]} = gen_{R_8,In[B_5]} = \{d_2, d_3, d_4, d_5, d_6\}$$

$$kill_{R_8,Out[B_5]} = kill_{R_8,In[B_5]} = \{d_1, d_4, d_5, d_6\}$$

9.2. Анализ потока данных на основе областей

9.2.4. Пример: применение алгоритма 9.2.3

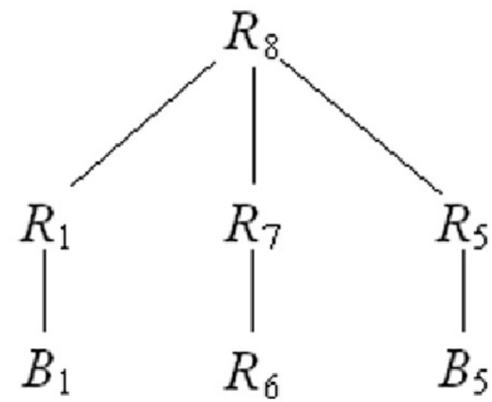
◊ **Результаты шага 2**

	Передаточная функция	gen	$kill$
R_6	$f_{R_6,\text{IN}[R_2]} = I$	$\emptyset$	$\emptyset$
	$f_{R_6,\text{OUT}[B_2]} = f_{R_2,\text{OUT}[B_2]} \circ f_{R_6,\text{IN}[R_2]}$	$\{d_4\}$	$\{d_1\}$
	$f_{R_6,\text{IN}[R_3]} = f_{R_6,\text{OUT}[B_2]}$	$\{d_4\}$	$\{d_1\}$
	$f_{R_6,\text{OUT}[B_3]} = f_{R_3,\text{OUT}[B_3]} \circ f_{R_6,\text{IN}[R_3]}$	$\{d_4, d_5\}$	$\{d_1, d_3\}$
	$f_{R_6,\text{IN}[R_4]} = f_{R_6,\text{OUT}[B_2]} \wedge f_{R_6,\text{OUT}[B_3]}$	$\{d_4, d_5\}$	$\{d_1\}$
	$f_{R_6,\text{OUT}[B_4]} = f_{R_4,\text{OUT}[B_4]} \circ f_{R_6,\text{IN}[R_4]}$	$\{d_4, d_5, d_6\}$	$\{d_1, d_2\}$
R_7	$f_{R_7,\text{IN}[R_6]} = f^*_{R_6,\text{OUT}[B_4]}$	$\{d_4, d_5, d_6\}$	$\emptyset$
	$f_{R_7,\text{OUT}[B_3]} = f_{R_6,\text{OUT}[B_3]} \circ f_{R_7,\text{IN}[R_6]}$	$\{d_4, d_5, d_6\}$	$\{d_1, d_3\}$
	$f_{R_7,\text{OUT}[B_4]} = f_{R_6,\text{OUT}[B_4]} \circ f_{R_7,\text{IN}[R_6]}$	$\{d_4, d_5, d_6\}$	$\{d_1, d_2\}$
R_8	$f_{R_8,\text{IN}[R_1]} = I$	$\emptyset$	$\emptyset$
	$f_{R_8,\text{OUT}[B_1]} = f_{R_1,\text{OUT}[B_1]}$	$\{d_1, d_2, d_3\}$	$\{d_4, d_5, d_6\}$
	$f_{R_8,\text{IN}[R_7]} = f_{R_8,\text{OUT}[B_1]}$	$\{d_1, d_2, d_3\}$	$\{d_4, d_5, d_6\}$
	$f_{R_8,\text{OUT}[B_3]} = f_{R_7,\text{OUT}[B_3]} \circ f_{R_8,\text{IN}[R_7]}$	$\{d_2, d_4, d_5, d_6\}$	$\{d_1, d_3, d_4, d_5, d_6\}$
	$f_{R_8,\text{OUT}[B_4]} = f_{R_7,\text{OUT}[B_4]} \circ f_{R_8,\text{IN}[R_7]}$	$\{d_3, d_4, d_5, d_6\}$	$\{d_1, d_2, d_4, d_5, d_6\}$
	$f_{R_8,\text{IN}[R_5]} = f_{R_8,\text{OUT}[B_3]} \wedge f_{R_8,\text{OUT}[B_4]}$	$\{d_2, d_3, d_4, d_5, d_6\}$	$\{d_1, d_4, d_5, d_6\}$
	$f_{R_8,\text{OUT}[B_5]} = f_{R_5,\text{OUT}[B_5]} \circ f_{R_8,\text{IN}[R_5]}$	$\{d_2, d_3, d_4, d_5, d_6\}$	$\{d_1, d_4, d_5, d_6\}$

9.2. Анализ потока данных на основе областей

9.2.4. Пример: применение алгоритма 9.2.3

◊ **Шаг 3:** *Нисходящий просмотр* для вычисления значений $In[R]$ в начале каждой области:



1) Для области R_8

$In[R_8] = \varnothing$, (граничное условие).

2) Для подобластей области R_8 :

2.1) область R_1

$$In[R_1] = f_{R_8, In[R_1]}(In[R_8]) = I(In[R_8]) = In[R_8] = \varnothing$$

2.2) область R_7

$$In[R_7] = f_{R_8, In[R_7]}(In[R_8]) = f_{B_1}(In[R_8]) = gen_{B_1} \cup (In[R_8] - kill_{B_1}) =$$

$$= (\varnothing - \{d_4, d_5, d_6\}) \cup \{d_1, d_2, d_3\} = \{d_1, d_2, d_3\}$$

2.3) область R_5

$$In[R_5] = f_{R_8, In[R_5]}(In[R_8]) = gen_{R_8, In[R_5]} \cup (In[R_8] - kill_{R_8, In[R_5]})$$

$$= \{d_2, d_3, d_4, d_5, d_6\} \cup (\varnothing - \{d_1\}) = \{d_2, d_3, d_4, d_5, d_6\}$$

9.2. Анализ потока данных на основе областей

9.2.4. Пример: применение алгоритма 9.2.3

- **Шаг 3:** *Нисходящий просмотр* для вычисления значений $In[R]$ в начале каждой области:

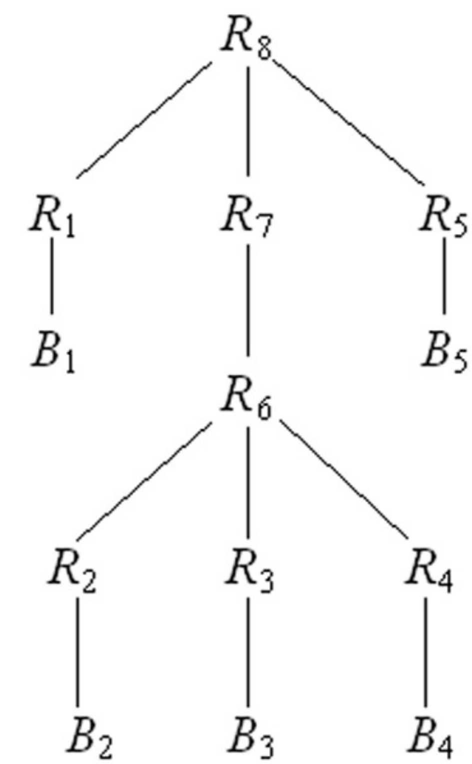
3) Для подобласти R_6 области R_7 :

$$In[R_6] = f_{R_7, In[R_6]}(In[R_7]) =$$

$$= gen_{R_7, In[R_6]} \cup (In[R_7] - kill_{R_7, In[R_6]}) =$$

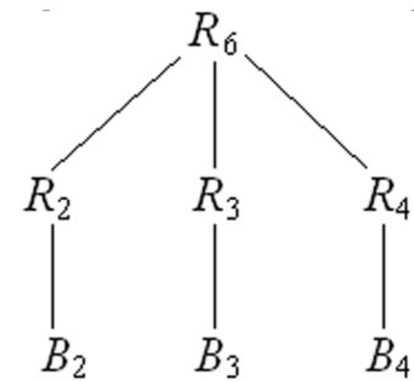
$$= \{d_4, d_5, d_6\} \cup (\{d_1, d_2, d_3\} - \varnothing) =$$

$$= \{d_1, d_2, d_3, d_4, d_5, d_6\}$$



9.2. Анализ потока данных на основе областей

9.2.4. Пример: применение алгоритма 9.2.3



- **Шаг 3:** *Нисходящий просмотр* для вычисления значений $In[R]$ в начале каждой области:

4) Для подобластей области R_6:

4.1) область R_4

$$In[R_4] = f_{R_6, In[R_4]}(In[R_6]) = gen_{R_6, In[R_4]} \cup (In[R_6] - kill_{R_6, In[R_4]})$$
$$= \{d_4, d_5\} \cup (\{d_1, d_2, d_3, d_4, d_5, d_6\} - \{d_1\}) = \{d_2, d_3, d_4, d_5, d_6\}$$

4.2) область R_3

$$In[R_3] = f_{R_6, In[R_3]}(In[R_6]) = gen_{R_6, In[R_3]} \cup (In[R_6] - kill_{R_6, In[R_3]})$$
$$= \{d_4, d_5\} \cup (\{d_1, d_2, d_3, d_4, d_5, d_6\} - \{d_1\}) = \{d_2, d_3, d_4, d_5, d_6\}$$

4.3) область R_2

$$In[R_2] = f_{R_6, In[R_2]}(In[R_6]) = gen_{R_6, In[R_2]} \cup (In[R_6] - kill_{R_6, In[R_2]})$$
$$= \varnothing \cup (\{d_1, d_2, d_3, d_4, d_5, d_6\} - \varnothing) = \{d_1, d_2, d_3, d_4, d_5, d_6\}$$

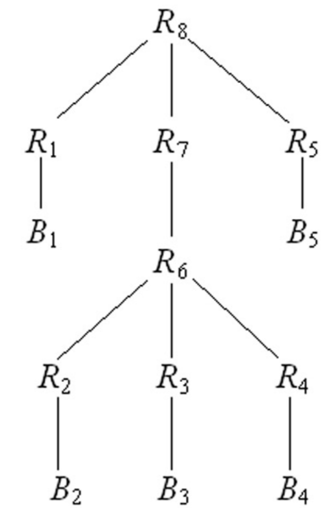
9.2. Анализ потока данных на основе областей

9.2.4. Пример: применение алгоритма 9.2.3

- **Результаты шага 3**

Значения потока данных для достигающих определений:

$$In[R_8] = \varnothing$$
$$In[R_1] = f_{R_8, In[R_1]}(In[R_8]) = \varnothing$$
$$In[R_7] = f_{R_8, In[R_7]}(In[R_8]) = \{d_1, d_2, d_3\}$$
$$In[R_5] = f_{R_8, In[R_5]}(In[R_8]) = \{d_2, d_3, d_4, d_5, d_6\}$$
$$In[R_6] = f_{R_7, In[R_6]}(In[R_7]) = \{d_1, d_2, d_3, d_4, d_5, d_6\}$$
$$In[R_4] = f_{R_6, In[R_4]}(In[R_6]) = \{d_2, d_3, d_4, d_5, d_6\}$$
$$In[R_3] = f_{R_6, In[R_3]}(In[R_6]) = \{d_2, d_3, d_4, d_5, d_6\}$$
$$In[R_2] = f_{R_6, In[R_2]}(In[R_6]) = \{d_1, d_2, d_3, d_4, d_5, d_6\}$$



9.2. Анализ потока данных на основе областей

9.2.4. Пример: применение алгоритма 9.2.3

◊ Сравнение результатов

Множества определений, достигающих входов в блоки B_1, B_2, B_3, B_4, B_5, вычисленные с помощью алгоритма 9.2.3:

$$In[B_1] = In[R_1] = \varnothing$$
$$In[B_2] = In[R_2] = \{d_1, d_2, d_3, d_4, d_5, d_6\}$$
$$In[B_3] = In[R_3] = \{d_2, d_3, d_4, d_5, d_6\}$$
$$In[B_4] = In[R_4] = \{d_2, d_3, d_4, d_5, d_6\}$$
$$In[B_5] = In[R_5] = \{d_2, d_3, d_4, d_5, d_6\}$$

Множества определений, достигающих входов в блоки B_1, B_2, B_3, B_4, B_5, вычисленные без выделения областей (слайд 49):

$$In[B_1] = \varnothing$$
$$In[B_2] = \{d_1, d_2, d_3, d_4, d_5, d_6\}$$
$$In[B_3] = \{d_2, d_3, d_4, d_5, d_6\}$$
$$In[B_4] = \{d_2, d_3, d_4, d_5, d_6\}$$
$$In[B_5] = \{d_2, d_3, d_4, d_5, d_6\}$$