Схемы алгоритмов

1.1 Логические схемы алгоритмов

1.1.1 Формальное определение логической схемы

Последовательности элементарных действий, выполняемых в течение одного микротакта синхросерии генератора тактовых импульсов, получили в вычислительной техники название микроопераций (или микроинструкций). Совокупность же микроопераций, выполняемых одновременно в i-том такте, называется микрокомандой. Иногда последовательность микроопераций является неизменной, однако более общим и важным случаем является случай, когда порядок их выполнения существенно зависит от результата выполнения предыдущих действий, то есть от некоторых условий.

Используя аппарат логики, это можно представить так. Предположим, нужно сравнить два числа x и y по абсолютной величине. Тогда запись вида $P(|x| = |y|)$ символизирует условие: если $|x| = |y|$, то $P = 1$, иначе $P = 0$. Это условие — основа для разветвления микропрограммы. Нередко условие в целом зависит от нескольких простых условий $p_1, p_2, ..., p_k$, т.е.: $\alpha_i = \alpha(p_1, p_2, ..., p_k)$.

Вполне понятно, что в этом случае выполнение микропрограммы является однозначно заданной функцией от k заданных условий. В работах А.А. Ляпунова подобная функция записывается в виде конечной строки, включающей символы $A_1, A_2, ..., A_n$ (микрооперации), логические функции $\alpha_1, \alpha_2, ..., \alpha_m$ и символов $\uparrow^i$ (начало) и $\downarrow^i$ (конец) стрелок, где $i = 1, 2, 3, ...$

В дальнейшем символы $\mathrm{A}_i (i = 1, 2, ..., n)$ будем называть «операторами». Строка $A_1\alpha \uparrow^1 A_2 ... \downarrow^1 A_i ... A_n$ в этом случае означает, что после выполнения оператора A_1 может быть два случая.

- $\alpha = 1$, тогда выполняется оператор A_2;
- $\alpha = 0$, тогда выполняется оператор A_i, непосредственно справа от конца стрелки $\downarrow^1$.

При этом всегда будем считать, что функция α может принимать лишь два значения — 0 или 1.

Условимся логическую функцию $\alpha \uparrow^i$ называть «логическим условием».

Определение. *Логической схемой алгоритма (ЛСА) называют конечную строку из символов* $A_1, A_2, ..., A_n$, *логических условий* $\alpha_1, \alpha_2, ..., \alpha_m$ *и концов стрелок* $\downarrow^1, ..., \downarrow^m$ *такую, что для каждого начала стрелки* $\uparrow^i$ *найдется один и только один конец стрелки с тем же индексом.*

Будем считать, что в ЛСА каждый оператор A_i встречается только один раз (в случае, если это не так, можно ввести переобозначение).

Назовем символы операторов и логических условий (ЛУ) «элементарным выражением», и всякую строку, состоящую из таких выражений и концов стрелок, так что для каждого индекса i найдется не более одного начала и одного конца стрелки, «выражением».

Итак, ЛСА по вышеизложенному определению задает порядок выполнения операторов в зависимости от значений ЛУ.

Рассмотрим ЛСА $U(p_1, ..., p_m, A_1, ..., A_n)$, где p_i — ЛУ, являющиеся независимыми логическими переменными. По условию каждая переменная может принимать значение 0 или 1.

Обозначим всевозможные наборы значений переменных $p_1, ..., p_m$ через $\Delta = \Delta_1, \Delta_2, ..., \Delta_{2^m}$.

Тогда процесс реализации ЛСА для произвольной последовательности Δ может быть определен следующим образом:

- На первом шаге задаем значения переменных в виде Δ_{s1}.
- На втором — отмечаем самое левое элементарное выражение схемы алгоритма U.
- ...
- После нескольких последовательно выполняемых шагов получаем: $A_{i1}, A_{i2}, ..., A_{i(L-1)}$.
- Проанализируем элементарное выражение, следующего шага.
  - Если это оператор A_{iL}, то приписываем его к полученной строке справа и задаем значения набора переменных в виде $\Delta_{s(L+1)}$, а затем переходим к следующему элементарному выражению.
  - Если же это ЛУ $\alpha_j(p_1, ..., p_m) \uparrow^i$, то действуем уже описанным способом, то есть при $\alpha_j = 1$ переходим к следующему элемен-

тарному выражению справа, иначе — к ближайшему справа от конца $\downarrow^i$.

- Процесс реализации ЛСА заканчивается, если выполнен заключительный оператор A_n.

Полученную строку операторов назовем значением ЛСА U для последовательности наборов Δ. При этом заметим, что значения ЛУ могут изменяться только во время выполнения операторов.

Введем некоторые понятия и определения, предложенные в работах Ю.И. Янова.

Пусть задана ЛСА U и два множества $\Phi = \{A_1, A_2, ..., A_n\}$ и $\Psi = \{p_1, ..., p_m\}$, принадлежащие U. Каждому оператору $A_i (i = 1, 2, ..., n) \in A$ поставим в соответствие некоторое множество ЛУ $\Psi_i \subseteq \Psi$, которые могут изменять свои значения на обратные во время выполнения оператора A_i. Ю.И. Янов назвал это соответствие «распределением сдвигов» : $A_i - \Psi_i$.

Среди всевозможных вариантов распределений сдвигов различают следующие.

1. Пустое, когда оператору поставлено в соответствие пустое множество $\Psi_i = \lambda$.

2. Универсальное, когда оператору поставлено в соответствие множество всех логических переменных $\Psi_i = \Psi$.

3. Нормальное, когда $\Psi_i \subset \Psi$.

Положим, ЛСА для фиксированной последовательности наборов логических условий $\Delta_{s1}, \Delta_{s2}, ..., \Delta_{st}, \Delta_{s(t+1)}, ...$ реализуется в виде цепочки операторов $A_{i1}, A_{i2}, ..., A_{is_t}, A_{is_{(t+1)}}, ...$

Последовательность наборов рассмотренного вида называют допустимой для схемы алгоритма при заданном распределении сдвигов, если всякий набор логических переменных $\Delta_{s(t+1)}$ либо совпадает с набором Δ_{st}, либо отличается от него значениями переменных из множества Ψ_{it}.

Говорят, что две ЛСА $U1$ и $U2$ «равносильны» при заданном распределении сдвигов $(U1 = U2)$, если для всякой допустимой последовательности наборов значения этих ЛСА совпадают.

Само по себе это определение не дает эффективного способа для определения равносильности из-за огромного перебора в случае сложных ЛСА, но зато помогает получить ряд формул преобразований ЛСА в равносильную ей.

Равносильное преобразование состоит в замене одних выражений другими и в изменении взаимного расположения выражений. Это оказывается чрезвычайно полезно в процессе создания систем обработки информации, когда алгоритмы функционирования представлены в терминах ЛСА, кроме того при оптимизации ПО.

Будем говорить, что в ЛСА $U(p_1, ..., p_m, A_1, ..., A_n)$ элементарное выражение D_i при заданном распределении сдвигов «подчинено» логической функции $\alpha(p_1, ..., p_m)$, и записывать этот факт как $D_i \prec \alpha(p_1, ..., p_m)$, если всякий набор Δ_r, при котором должно выполняться D_i, обращает функцию α в единицу.

Например, $U = A_1\alpha(p_1, ..., p_m) \uparrow^1 A_2A_3A_4 \downarrow^1 A_5$. Здесь оператор $A_2 \prec \alpha(p_1, ..., p_m)$, так как после выполнения A_1 оператор A_2 выполняется лишь в случае, если $\alpha = 1$. Оператор A_3 не подчинен α, так как оператор A_2 при распределении сдвигов A_2–$\Psi_2 \subset \Psi$ может изменить значения некоторых переменных таким образом, что функция α при выполнении A_3 уже не будет равна 1. Тем не менее, в работе В.Ф. Дъяченко показано, что оператор A_3 подчинен некоторой логической функции $\beta(p_1, ..., p_m)$, которая является $\beta = max\alpha(p_1, ..., p_m)$ на всех наборах значений переменных множества Ψ_2.

Свойства функции $\beta = max(\Psi_i)\alpha$ таковы, что если

- $\alpha(\Delta) = 1$, то $\beta(\Delta) = \beta(\Delta') = 1$,
- $\alpha(\Delta) = 0$ и $\alpha(\Delta') = 0$, то $\beta(\Delta) = \beta(\Delta') = 0$,
- $\alpha(\Delta) = 0$ и $\alpha(\Delta') = 1$, то $\beta(\Delta) = \beta(\Delta') = 0$,

где Δ' — набор значений переменных $p_1, ..., p_m$, который отличается от набора Δ значениями тех переменных, которые принадлежат Ψ_i. Отсюда следует, что A_4, например, подчинен исходной функции $A_4 \prec max(\Psi_3)\beta = max(\Psi_3)(max(\Psi_2)\alpha)$.

Назовем конъюнкцию логических переменных «элементарной», если в ней не содержится повторяющихся ЛУ. Чтобы по заданной логической функции найти ее max по всем наборам переменных из множества Ψ_i, следует.

1. Представить α в ДНФ.

2. В каждой элементарной конъюнкции все переменные, входящие в Ψ (в том числе и с отрицанием), заменить единицей.

Рассмотрим пример. Дана функция $\alpha = p_1\neg p_2 \vee p_3(\neg p_1 p_2 \vee p_4)$, тогда $\beta = max(\{p_1, p_4\})[p_1\neg p_2 \vee p_3(\neg p_1 p_2 \vee p_4)] = max(\{p_1, p_4\})[p_1\neg p_2 \vee \neg p_1 p_2 p_3 \vee p_3 p_4] = 1\neg p_2 \vee 1 p_2 p_3 \vee p_3 1 = \neg p_2 \vee p_3$

1.1.2 Полная система преобразований Янова

Ю.И.Янов в своих работах предложил специальное ассоциативное исчисление, где формулы представлены как $B = G$, где B и G — выражения. Во всякой ЛСА $U(B)$ можно провести операцию подстановки $B \to G$. Это дает возможность получить ЛСА с минимумом ЛУ, но путем перебора всевозможных ЛСА, равносильных заданной. Сложность подобных преобразований чрезвычайно высока, ввиду чего были предприняты усилия, направленные на обхождение указанной проблемы.

1.1.3 Представление ЛСА системой формул перехода

В работах В.Г. Лазарева и В.Ф. Дъяченко предлагается применять специальные формализмы для особой записи ЛСА, которая позволяет применять уже известные апробированные методы, развитые в теории релейно-контактных схем.

Если $\alpha, \beta, \gamma, ...$ — логические функции от переменных $p_1, ..., p_n, \mathrm{A}_i \in A$, где A — множество операторов $\mathrm{A} = \{A_1, ..., A_m\}$, то запись вида $\alpha\mathrm{A}_i$ (или $p_j\mathrm{A}_i$) назовем «отмеченной функцией» (отмеченной переменной).

$$\alpha A_i = \begin{cases} A_i & , \alpha = 1, \\ 0 & , \alpha = 0. \end{cases}$$

В работах П.С. Новикова показано, что над логическими функциями, входящими в состав отмеченных, можно производить все преобразования алгебры логики.

Для них справедливы следующие правила, сокращенно называемые системой $C1$.

1. $(\alpha \vee \beta)A_i = \alpha A_i \vee \beta A_i$;

2. $\alpha\beta A_i \vee \alpha\gamma A_j = \alpha(\beta A_i \vee \gamma A_j)$;

3. $\alpha = \beta \to \alpha A_i = \beta A_i$.

Указанную систему правил совместно с полной системой аксиом алгебры логики следует рассматривать как систему правил тождественных преобразований отмеченных функций.

Запись вида «$\mathrm{A}_i \rightharpoonup \mathrm{A}_j$» следует читать как «за A_i следует A_j», то есть $U = ...\mathrm{A}_i\mathrm{A}_j....$

Между всеми операторами $\mathrm{A}_i \in \mathrm{A}$ в ЛСА существует различная возможность взаимопереходов, что может быть формально описано следующим образом: $\mathrm{A}_i \rightharpoonup \alpha_{i1}A_i \vee \alpha_{i2}A_2 \vee ... \vee \alpha_{in}A_n$, где $\mathrm{A}_i \in \mathrm{A}$ — операторы, $\alpha_{ij}(p_1, ..., p_m)$ — логические функции от независимых переменных, $\alpha_{ij}\mathrm{A}_j$ — отмеченная функция.

Очевидно, что после выполнения оператора A_i может следовать лишь один из операторов A_1 или A_2, или ..., поэтому множество функций α_{ij} должно обладать двумя свойствами.

1. Ортогональность $\alpha_{ij}\alpha_{ik} \equiv 0, (j, k = 1, ..., n)$;

2. Полнота $\vee(j = 1, ..., n)[\alpha_{ij}] = 1$.

Полное ортогональное множество функций называется «нормальным». Ортогональным будет также и множество отличных функций $\{\alpha_{ij}\mathrm{A}_j\}$. Выражение рассмотренного вида называется формулой перехода для оператора A_i.

Конструктивный способ построения формул перехода заключается в следующем. Необходимо выписать из заданной ЛСА всевозможные последовательности логических переменных (конъюнкций), которые следуют за оператором A_i, после чего построить дизъюнкцию отмеченных логических функций.

В общем виде сказанное может быть представлено как система переходов $S1$.

$$\begin{cases} A_1 \rightharpoonup \vee(j=1,...,n)[\alpha_{1j}A_j] \\ ... \\ A_i \rightharpoonup \vee(j=1,...,n)[\alpha_{ij}A_j] \\ ... \\ A_n \rightharpoonup \vee(j=1,...,n)[\alpha_{nj}A_j] \end{cases}$$

Определим процесс выполнения системы формул перехода $S1$ для последовательности наборов Δ.

- Задать значения переменных Δ_{i1} и определить значение функции $\alpha_{1j}(j=1,...,n)$ в формуле перехода для оператора A_1; если тождественно $\alpha_{1j}(\Delta_{i1})=1$, то выполняется переход к оператору A_j.
- Задать значения переменных Δ_{ij} и определить значение $\alpha_{ij}=1$, при этом найдя оператор, к которому выполняется очередной переход.
- Процесс продолжается до тех пор, пока не выполнится оператор, символизирующий окончание алгоритма.

В результате будет получена строка $\mathrm{A}_{i1}\mathrm{A}_{i2}\mathrm{A}_{i3}...\mathrm{A}_{ik}\mathrm{A}_{i(k+1)}...$, являющаяся значением системы формул переходов для последовательности наборов Δ. Аналогичным образом назовем последовательность наборов вида Δ допустимой, если значение $\Delta_{i(k+1)}$ совпадает со значением Δ_{ik} или отличается от него только теми переменными, которые входят в подмножество, изменяемое оператором A_{ik} во время его выполнения.

Две формулы перехода называются «равносильными», если их значения на данном наборе из любой допустимой последовательности совпадают. Назовем функцию α_{ij} приведенной по переменной p_s, если она может быть представлена как $\alpha_{ij}=p_s(\alpha'_{ij})\vee\neg p_s(\alpha''_{ij})$.

Это свойство распространяется также и на отмеченные функции.

Поясним вышесказанное на примере.

$$A_1 \rightharpoonup p_2p_3A_1 \vee p_1\neg p_2A_2 \vee \neg p_1\neg p_3A_3 \vee (p_1p_2\neg p_3 \vee \neg p_1\neg p_2p_3)A_4$$

Пусть требуется реализовать операцию приведения по переменной p_1. Первый член дизъюнкции не содержит переменной p_1, но если умножить

его на $1 \equiv p_1 \vee \neg p_1$, то получим $(p_1 \vee \neg p_1)p_2p_3A_1 = p_1p_2p_3A_1 \vee \neg p_1p_2p_3A_1$. Затем вынесем за скобки p_1 и $\neg p_1$, что дает

$$p_1(p_2p_3A_1 \vee \neg p_2A_2 \vee p_2\neg p_3A_4) \vee \neg p_1(p_2p_3A_1 \vee \neg p_3A_3 \vee \neg p_2p_3A_4).$$

Таким же образом можно выполнить операции приведения по переменной p_2, p_3. Например, для рассматриваемого примера получим

$$A_1 \rightharpoonup p_1[p_2(p_3A_1 \vee \neg p_3A_4) \vee \neg p_2A_2] \vee \neg p_1[p_3(p_2A_1 \vee \neg p_2A_4)\neg p_3A_3].$$

В соответствии с введенной В.Ф. Дьяченко терминологией назовем данное выражение «скобочной» формулой перехода, а все формулы перехода U — «системой скобочных формул перехода» $S2$. При получении системы скобочных формул полезно использовать некоторые очевидные равносильные соотношения.

$$\begin{cases} p_iA_j \vee p_iA_j = p_iA_j \\ p_iA_j \vee \neg p_iA_j = A_j \\ p_iA_j \vee p_ip_sA_j = p_iA_j \\ A_j \vee p_iA_j = A_j \end{cases}$$

При этом порядок вынесения переменных за скобки не задается, а выбирается из соображений целесообразности.

Таким образом, любые ЛСА могут быть представлены системой формул перехода $S1$, а также системой скобочных формул перехода $S2$. Однако, существует и третья форма — система схемных формул перехода $S3$, получаемая из $S2$ введением в алфавит ЛСА символа разделительного знака «$*$». Для рассматриваемого примера формулы перехода будут иметь следующий вид

$$A_1 \rightharpoonup p_1 \uparrow^1 p_2 \uparrow^2 p_3 \uparrow^3 A_1* \downarrow^3 A_4* \downarrow^2 A_2* \downarrow^1 A_1 \\ \neg p_3 \uparrow^4 A_3* \downarrow^4 p_2 \uparrow^5 A_1* \downarrow^5 A_4.$$

Условимся называть символы между «$*$» «выражениями», а выражение, стоящее сразу после $\rightharpoonup$ — «начальным выражением».

Заметим, что стрелки $\uparrow^i$ и их концы $\downarrow^i$ в схемной формуле перехода (как и в ЛСА) следует считать знаками операторной связи, а знаки дизъюнкции $\vee$, конъюнкции $\wedge$, отрицания $\neg$ — логической связи. Переход от скобочной к равносильной ей схемной формуле перехода заключается в замене логических связей между элементарными выражениями операторными.

Предположим, дана формула перехода $A_i \rightharpoonup p_{s1}(Q_1) \vee \neg p_{s1}(Q_2)$.

Нетрудно показать, что для нее будут равносильны следующие две схемные формулы.

$$\begin{cases} A_i \rightharpoonup p_{s1} \uparrow^1 Q_1 * \downarrow^1 Q_2 \\ A_i \rightharpoonup \neg p_{s1} \uparrow^1 Q_2 * \downarrow^1 Q_1 \end{cases}$$

Составные выражения Q_1 и Q_2 в свою очередь могут быть представлены как

$$\begin{cases} Q_1 = p_{s2} Q_1' \vee \neg p_{s2} Q_1'' \\ Q_2 = p_{s3} Q_2' \vee \neg p_{s3} Q_2'' \end{cases}$$

Тогда формула перехода может быть записана следующим образом

$$A_i \rightharpoonup p_{s1}(p_{s2} Q_1' \vee \neg p_{s2} Q_1'') \vee \neg p_{s1}(p_{s3} Q_2' \vee \neg p_{s3} Q_2'').$$

Пары переменных p_{s1} и $\neg p_{s1}$, p_{s2} и $\neg p_{s2}$, p_{s3} и $\neg p_{s3}$ называются «связными».

Следовательно, для перехода от системы $S2$ к системе $S3$ следует воспользоваться следующим набором правил.

1. Опустить все скобки формул перехода системы $S2$.
2. Заменить знаки «$\vee$» равносильными знаками «*».
3. Заменить все пары связанных переменных p_s и $\neg p_s$ парами $p_s \uparrow^i$ и $\downarrow^i$.

Порядок выполнения правил не играет роли.

Полученная система $S3$ будет равносильна $S2$, а число формул не изменится.

Определение. *Две системы $S3'$ и $S3''$ называются равносильными, если для любой последовательности наборов логических операторов их значения совпадают.*

1.1.4 Тождественные преобразования схемных формул перехода

Определение. *Оператор или часть схемной формулы перехода, которая соответствует заключенной в скобки части скобочной формулы перехода, назовем «подформулой».*

Другими словами, подформулой является такой набор выражений, где для каждого начала стрелки с индексом $\uparrow^i$ найдется конец $\downarrow^i$ с тем же индексом. Например.

$$A_i \rightharpoonup p_1(p_2A_1 \vee \neg p_2A_2) \vee \neg p_1[p_3A_3 \vee \neg p_3(p_2A_1 \vee \neg p_2A_2)]$$
$$A_i \rightharpoonup p_1 \uparrow^1 p_2 \uparrow^2 A_1* \downarrow^2 A_2* \downarrow^1 p_3 \uparrow^3 A_3* \downarrow^3 p_2 \uparrow^4 A_1* \downarrow^4 A_2$$

Тогда подформулами будут являться следующие последовательности символов.

- $N_1 = p_2 \uparrow^2 A_1* \downarrow^2 A_2$
- $N_2 = p_3 \uparrow^3 A_3* \downarrow^3 p_2 \uparrow^4 A_1* \downarrow^4 A_2$
- $N_3 = A_1$
- $N_4 = A_2$
- $N_5 = A_3$

Можно заметить, что N_1 входит в N_2, а N_3, N_4, N_5 входят в N_1 и N_2. Теоретически, всю формулу перехода можно считать подформулой. Ясно, что выделение подформул из формулы перехода является неоднозначным процессом. Например, схемная формула перехода может быть записана как $A_i \rightharpoonup p1 \uparrow^1 N_1* \downarrow^1 N_2$ или $A_i \rightharpoonup p_1 \uparrow^1 N_1* \downarrow^1 p_3 \uparrow^2 N_5* \downarrow^2 N_1$, или же и вовсе в общем виде $A_i \rightharpoonup R_1 \uparrow^1 N_1 * ... * R_i \downarrow^i N_i * W$, где N_i — подформулы, R_i — остаток выражения, не вошедший в подформулу, W — выражение, не вошедшее в подформулы и остатки.

Определение. *Две подформулы N_1 и N_2 называются «равносильными», если на заданном наборе Δ_i логических переменных, входящих в полформулы, выполняется один и тот же оператор A_i (то есть их значения совпадают).*

Рассмотрим систему тождественных преобразований схемных формул перехода $S3$, получившую название системы Лазарева-Дьяченко.

1. (a) $(A_i \rightharpoonup N) = (A_i \rightharpoonup 1 \uparrow^1 N \downarrow^1)$

   (b) $(A_i \rightharpoonup N) = (A_i \rightharpoonup\downarrow^1 N\ 1 \uparrow^1)$

2. (a) $(A_i \rightharpoonup R_{i1} \downarrow^{s1} N * R_{i2} \downarrow^{s2} N * W) = (A_i \rightharpoonup R_{i1} \downarrow^i\downarrow^{s1}\downarrow^{s2} N * R_{i2}\omega \uparrow^i N * W) = (A_i \rightharpoonup R_{i1}\omega \uparrow^i * R_{i2} \downarrow^i\downarrow^{s1}\downarrow^{s2} N * W)$

   (b) $(A_i \rightharpoonup P_1 p_i \uparrow^i P_2 A_m* \downarrow^i P_3 A_l) = (A_i \rightharpoonup P_1\neg p_i \uparrow^i P_3 A_l* \downarrow^i P_2 A_m)$

   При этом $\omega \uparrow^i$ — безусловный переход, P_1, P_2, P_3 — некоторые (возможно, пустые) последовательности логических условий $p_{s1} \uparrow^1 p_{s2} \uparrow^2 p_{s3} \uparrow^3 ... p_{sn} \uparrow^n$

3. $$\begin{cases} A_{i1} \rightharpoonup W_{i1}* \downarrow^{s1} N \\ ... \\ A_{ik} \rightharpoonup W_{ik} * R \downarrow^{s2} N \end{cases} = \begin{cases} A_{i1} \rightharpoonup W_{i1} \\ ... \\ A_{ik} \rightharpoonup W_{ik} * R \downarrow^{s1}\downarrow^{s2} N \end{cases}$$

4. $(A_i \rightharpoonup P_1 p_2 \uparrow^i P_2 N_k* \downarrow^i p_2 \uparrow^j N_q* \downarrow^j P_3 N_2 * W) = (A_i \rightharpoonup P_1 p_2 \uparrow^i P_2 N_k* \downarrow^i P_3 N_2 * W)$

   N_q — любая подформула

5. $(A_i \rightharpoonup R_p \uparrow^i N* \downarrow^i\downarrow^s \lambda * W) = (A_i \rightharpoonup R \downarrow^i\downarrow^s p \uparrow^i N * W)$

   λ — пустая подстановка

6. $\downarrow^i\downarrow^s = \downarrow^s\downarrow^i$

7. $p \uparrow^i ... \downarrow^i = p \uparrow^s ... \downarrow^s$

8. $(\alpha \uparrow^i \prec \beta(p_1, ..., p_n)) \to (\alpha \uparrow^i = (\alpha \vee \neg\beta) \uparrow^i)$

9. $(N_1 = N_2 * N_3) \to (RN_1 = RN_2 * N_3)$

10. $(N_n = N_m; \Phi'(N_n) = \Phi'') \to (\Phi'(N_m) = \Phi'')$

Все указанные соотношения принято называть системой $C2$ и вместе с системой $C1$ они составляют полную систему тождественных преобразований схемных формул. Таким образом, для любой ЛСА U_1 через конечное число применений $\{C1, C2\}$ можно получить равносильную $U_2 = U_1$, оптимальную по некоторым параметрам (например, по числу ЛУ).

1.1.5 Алгоритм перехода от системы $S3$ к ЛСА

После цепочки преобразований ЛСА $U1 \to S1 \to S2 \to S3$ предоставляется возможность произвести оптимизацию схемных формул перехода при помощи полной системы тождественных преобразований $\{C1, C2\}$. Затем возникает проблема обратного перехода от $S3$ к ЛСА $U2 = U1$.

Алгоритм перехода изображен на рис. 1. Необходимо оговориться, что представленный алгоритм основывается не предположении, что каждый из операторов A_i встречается в системе $S3$ лишь однажды.

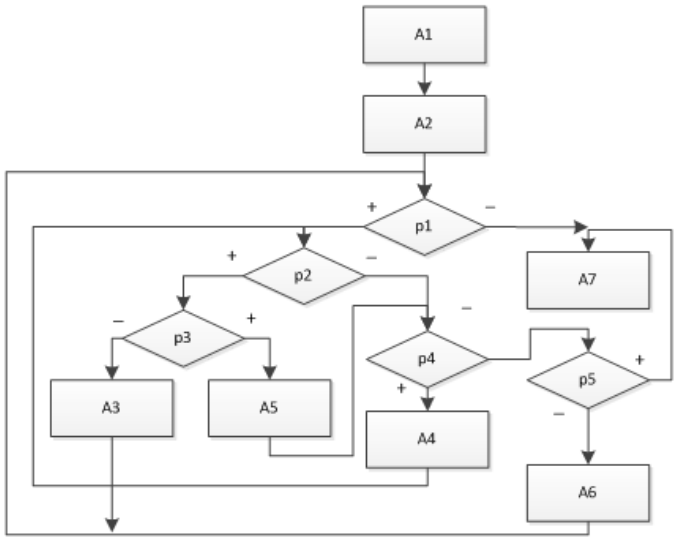

Рис. 1 — ГСА

Значение операторов.

- $A1$ — выписать оператор $A0$ вначале строки.
- $A2$ — выписать справа от $A0$ начальное выражение из схемной формулы $A0$, вычеркнув его.
- $A3$ — выписать начальное выражение из схемной формулы $A_i (i = 1, 2, ...)$,вычеркнув его.
- $A4$ — выписать любое не вычеркнутое не начальное выражение, если оно есть.
- $A5$ — заменить в выписанном выражении оператор A_k со всеми концами стрелок слева от него на $\omega \uparrow^s$ (s — целое положительное число); конец $\downarrow^s$ и оператор A_k переместить в конец строки ЛСА.

- $A6$ — выписать любой еще не встречавшийся в строке ЛСА оператор, в схемной формуле которого не вычеркнуто начальное выражение.
- A7 — останов.

Значение логических переменных.

- $p1$ — 1: в системе $S3$ есть невычеркнутые выражения; 0: все выражения вычеркнуты.
- $p2$ — 1: строка оканчивается оператором A_i; 0: строка оканчивается выражением $\omega \uparrow^i$.
- $p3$ — 1: строка оканчивается оператором A_k; 0: строка оканчивается выражением A_i.
- $p4$ — 1: в системе $S3$ есть невычеркнутые неначальные выражения; 0: все неначальные выражения вычеркнуты.
- $p5$ — 1: в строке имеются все операторы ЛСА; 0: есть неиспользуемые операторы.

1.1.6 Пример оптимизации ЛСА

$$U_1 = A_0 \downarrow^4 A_1 p_1 \uparrow^1 \downarrow^3 \downarrow^9 A_2 \downarrow^2 p_3 \uparrow^5 \omega \uparrow^4 \downarrow^1 p_2 \uparrow^2 \omega \uparrow^{10} \downarrow^5 \\ \neg p_4 \uparrow^6 \omega \uparrow^3 \downarrow^7 \neg p_4 \uparrow^8 \omega \uparrow^9 \downarrow^6 \downarrow^8 A_3 \neg p_1 \uparrow^7 \downarrow^{10} A_4 A_k$$

Построим систему $S1$.

$$\begin{cases} A_0 \rightharpoonup A_1 \\ A_1 \rightharpoonup p_1 A_2 \vee \neg p_1 p_2 A_4 \vee \neg p_1 \neg p_2 p_3 A_1 \vee \neg p_1 \neg p_2 \neg p_3 \neg p_4 A_2 \vee \\ \vee \neg p_1 \neg p_2 \neg p_3 p_4 A_3 \\ A_2 \rightharpoonup p_3 A_1 \vee \neg p_3 \neg p_4 A_2 \vee \neg p_3 p_4 A_3 \\ A_3 \rightharpoonup \neg p_1 A_4 \vee p_1 \neg p_4 A_2 \vee p_1 p_4 A_3 \\ A_4 \rightharpoonup A_k \end{cases}$$

Построим систему $S2$.

$$\begin{cases} A_0 \rightharpoonup A_1 \\ A_1 \rightharpoonup p_1 A_2 \vee \neg p_1 \{p_2 A_4 \vee \neg p_2 [p_3 A_1 \vee \neg p_3 (\neg p_4 A_2 \vee p_4 A_3)]\} \\ A_2 \rightharpoonup p_3 A_1 \vee \neg p_3 (\neg p_4 A_2 \vee p_4 A_3) \\ A_3 \rightharpoonup \neg p_1 A_4 \vee p_1 (\neg p_4 A_2 \vee p_4 A_3) \\ A_4 \rightharpoonup A_k \end{cases}$$

Построим систему $S3$.

$$\begin{cases} A_0 \rightharpoonup A_1 \\ A_1 \rightharpoonup p_1 \uparrow^1 A_2 * \downarrow^1 p_2 \uparrow^2 A_4 * \downarrow^2 p_3 \uparrow^3 A_1 * \downarrow^3 \neg p_4 \uparrow^4 A_2 * \downarrow^4 A_3 \\ A_2 \rightharpoonup p_3 \uparrow^5 A_1 * \downarrow^5 \neg p_4 \uparrow^6 A_2 * \downarrow^6 A_3 \\ A_3 \rightharpoonup \neg p_1 \uparrow^7 A_4 * \downarrow^7 \neg p_4 \uparrow^8 A_2 * \downarrow^8 A_3 \\ A_4 \rightharpoonup A_k \end{cases}$$

Анализирую формулы перехода A_1, A_2, A_3, заметим, что они содержат совпадающие (равносильные) подформулы (за исключением индексов стрелок).

Однако, правило 7 позволяет заменять индексы произвольным образом. Произведем замену индексов.

$$A_2 \rightharpoonup p_3 \uparrow^3 A_1 * \downarrow^3 \neg p_4 \uparrow^4 A_2 * \downarrow^4 A_3$$

$$A_3 \rightharpoonup \neg p_1 \uparrow^7 A_4 * \downarrow^7 \neg p_4 \uparrow^4 A_2 * \downarrow^4 A_3$$

Затем перенесем совпадающие подформулы в формулу перехода A_1.

$$\begin{cases} A_0 \rightharpoonup A_1 \\ A_1 \rightharpoonup p_1 \uparrow^1 A_2 * \downarrow^1 p_2 \uparrow^2 A_4 * \downarrow^2 \downarrow^9 p_3 \uparrow^3 A_1 * \downarrow^7 \downarrow^3 \neg p_4 \uparrow^4 A_2 * \downarrow^4 A_3 \\ A_2 \rightharpoonup \omega \uparrow^9 \\ A_3 \rightharpoonup \neg p_1 \uparrow^7 A_4 \\ A_4 \rightharpoonup A_k \end{cases}$$

После этого применим к формуле перехода A_1 правило 2, то есть проинвертируем пары связных переменных. Перенесем операторы A_4 в формулу A_3 и A_1 — в A_0.

$$\begin{cases} A_0 \rightharpoonup\downarrow^3 A_1 \\ A_1 \rightharpoonup \neg p_1 \uparrow^1 \neg p_2 \uparrow^2\downarrow^9 \neg p_3 \uparrow^3\downarrow^7 \neg p_4 \uparrow^4\downarrow^1 A_2* \downarrow^4 A_3 \\ A_2 \rightharpoonup \omega \uparrow^9 \\ A_3 \rightharpoonup \neg p_1 \uparrow^7\downarrow^2 A_4 \\ A_4 \rightharpoonup A_k \end{cases}$$

Воспользуемся вышеизложенным алгоритмом возврата от системы $S3$ к ЛСА.

$$U_2 = A_0 \downarrow^3 A_1 \neg p_1 \uparrow^1 \neg p_2 \uparrow^2\downarrow^9 \neg p_3 \uparrow^3\downarrow^7 \neg p_4 \uparrow^4\downarrow^1 A_2\omega \uparrow^9\downarrow^4 A_3 \neg p_1 \uparrow^7\downarrow^2 A_4 A_k$$

Полученная ЛСА U_2 равносильна U_1, но имеет меньшее число элементарных выражений ($U1/U2$).

- Число операторов — 6/6;
- Число логических условий — 6/5;
- Число тождественно ложных переходов ω — 4/1.

Очевидно, что если число операторов в исходной ЛСА было минимально, то оно останется таким же. Сокращение числа ЛУ и безусловных переходов ω зависит как от наличия равносильных подформул, так и от способа вынесения за скобки логических переменных. Следовательно, порядок вынесения за скобки переменных должен быть таким, чтобы увеличить число равносильных подформул, которые можно объединить.

В качестве заключения следует выделить некоторые практические рекомендации вынесения за скобки логических переменных в формулах перехода.

1. Начинать вынесение за скобки следует с той переменной, по которой приведено множество отмеченных элементарных конъюнкций.

2. Если множество приведено по нескольким логическим переменным, то можно начинать с любой переменной.

3. Если указанное множество отмечено по всем логическим переменным, то порядок безразличен при условии, что в формуле нет пары или более отмеченных элементарных конъюнкций с одинаковыми операторами (в противном случае будет получено увеличение числа ЛУ).