УДК 004.421 + 519.6

ОПТИМИЗИРУЮЩИЕ ПРЕОБРАЗОВАНИЯ АЛГОРИТМОВ, ИСПОЛЬЗУЮЩИЕ СВОЙСТВА МНОЖЕСТВ, ПРЕДИКАТОВ И ОПЕРАЦИЙ НАД НИМИ

В.А. Овчинников, **Г.С. Иванова**

МГТУ им. Н.Э. Баумана, Москва, Российская Федерация
e-mail: vaovchinnikov@gmail.com; gsivanova@gmail.com

Многие комбинаторно-оптимизационные задачи структурного синтеза относятся к классу NP-полных, время решения которых экспоненциально зависит от размера входа задачи. Большинство задач проектирования структур сложных систем имеют большую размерность входа — миллионы и более компонентов. В связи с этим даже для полиномиальных алгоритмов актуальной является проблема сокращения времени работы за счет снижения их вычислительной сложности. Цель работы — исследование предлагаемых способов снижения вычислительной сложности алгоритмов на графах и множествах, в том числе оценка эффективности этих преобразований. В статье определено понятие "оптимизирующие преобразования алгоритмов". Обоснована актуальность их применения. Выполнена классификация способов снижения вычислительной сложности алгоритмов, использующих свойства множеств, предикатов и операций над ними. Рассмотрено подробно каждое из восьми преобразований и приведена оценка их эффективности.

Ключевые слова: алгоритм, вычислительная сложность, оптимизирующие преобразования, эффективность, множества, предикаты, операции.

OPTIMIZING TRANSFORMATIONS OF ALGORITHMS USING PROPERTIES OF SETS, PREDICATES, AND OPERATIONS OVER THEM

V.A. Ovchinnikov, **G.S. Ivanova**

Bauman Moscow State Technical University, Moscow, Russian Federation
e-mail: vaovchinnikov@gmail.com; gsivanova@gmail.com

Many combinatorial optimization problems of structural synthesis relate to the class of NP-complete problems, the time required to solve them depends exponentially on the problem input size. A majority of problems to design structures of complex systems have a large dimensionality of input: millions and more components. In connection with this, even for polynomial algorithms, the problem to save their computation time for expense of reducing their computational complexity is urgent. This work is aimed at the investigation of the proposed methods for reducing the computational complexity of algorithms for graphs and sets including the estimation of efficiency of these transformations. The notion of optimizing transformations of algorithms is defined. The urgency of their application is substantiated. The methods for reducing the computational complexity of algorithms using the properties of sets, predicates, and operations over them are classified. Each of eight transformations is considered in detail and its efficiency is estimated.

Keywords: algorithm, computational complexity, optimizing transformations, efficiency, sets, predicates. operations.

Классификация способов снижения вычислительной сложности алгоритмов, использующих свойства множеств, предикатов и операций над ними. Под *оптимизирующими преобразованиями* будем понимать совокупность эвристических приемов модификации алгоритмов, которые позволяют в определенных случаях снижать их вычислительную сложность.

Оптимизационный эффект применения способов снижения вычислительной сложности к алгоритму достигается удалением лишних вычислений посредством замены части алгоритма на более эффективную, т.е. имеющую меньшую вычислительную сложность. Эта трансформация должна быть *корректной*, т.е. сохранять эквивалентность исходного и полученного алгоритмов.

Исходный и преобразованный алгоритмы *эквивалентны*, если на всех допустимых наборах входных данных задачи дают одинаковые результаты. *Корректность* изменений подразумевает эквивалентность заменяемого и заменяющего фрагментов и удовлетворение контекстных условий на требуемые свойства объектов алгоритма и/или связи заменяемого фрагмента с остальной частью алгоритма. Такими условиями являются, например, проверки наличия в алгоритме операторов и/или объектов, позволяющих использовать заменяющий фрагмент.

В [1] отмечено, что вычислительная сложность алгоритма зависит от многих факторов. В настоящей статье рассматриваются только способы снижения вычислительной сложности алгоритмов, использующие свойства множеств, предикатов и операций над ними. Классификация этих способов приведена на рис. 1.

Оптимизирующие преобразования, использующие свойства множеств и операций над ними. *Первый способ этой группы преобразований* заключается в реализации операции удаления элемента множества посредством его замены другим, например последним. Способ базируется на основном свойстве множества, т.е. применение его возможно, если элементы множества не упорядочены по какому-либо правилу и оно не участвует в дальнейших преобразованиях в первоначальном виде. Отметим, что в случае необходимости множество можно предварительно копировать. Информацию о возможности реализации способа нетрудно получить из описания алгоритма в терминах операций над графами и множествами.

Оценим эффективность применения способа. Пусть из множества A необходимо выбирать и удалять некоторый элемент. Оценка “в худшем” числа операций для удаления сдвигом равна $n - 1$. Удаление замещением будет выполнено за одну операцию.

Второй способ — это замена выражений алгебры подмножеств логически эквивалентными и требующими меньшего числа операций

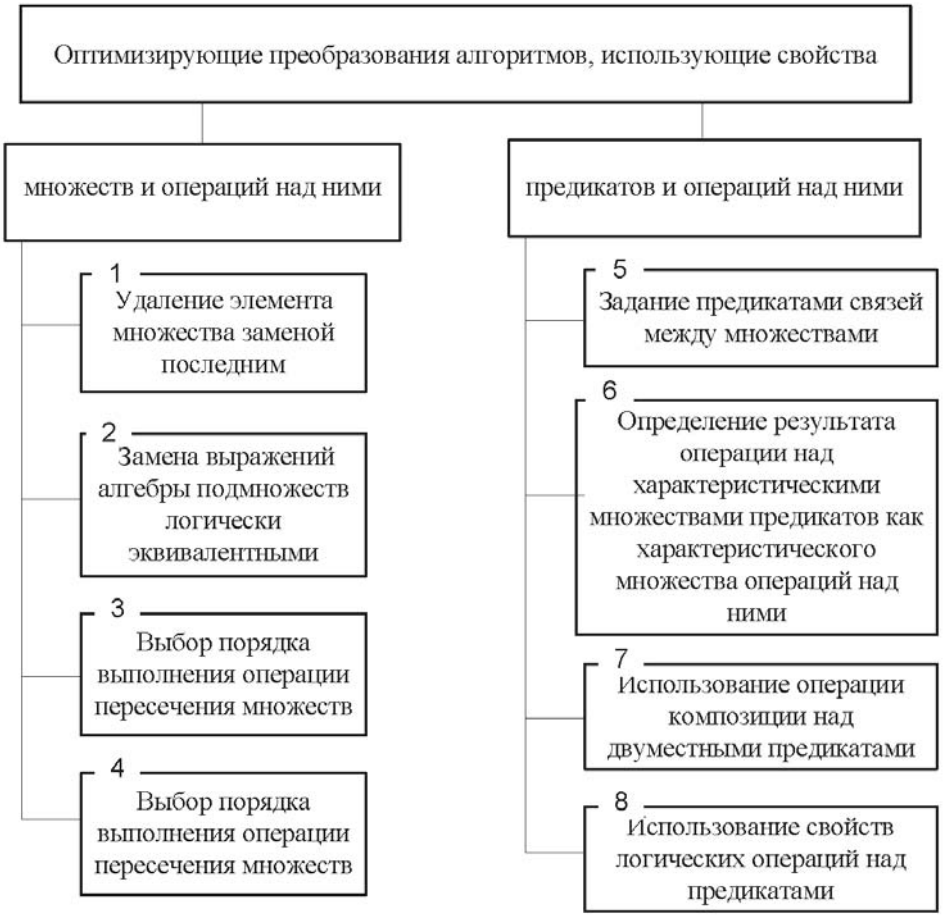

Рис. 1. Классификация способов снижения вычислительной сложности алгоритмов

для их реализации. В этом способе целесообразно выделить две группы:

— заменяемые и заменяющие выражения состоят из одних и тех же подмножеств;

— в заменяемые и заменяющие выражения могут входить и разные подмножества, на отношения между которыми наложены дополнительные условия.

К первой группе относятся, например:

— замена выражения вида $X \subseteq Y \bullet Z$ или $X \subset Y \bullet Z$, где $Z \subseteq X (Z \subset X)$, на конструкцию вида $\{X \backslash Z\} \subseteq Y$ или $\{X \backslash Z\} \subset Y$;

— использование для определения состава некоторого подмножества вместо выражения $X \backslash \{Y \cup Z\}$ выражения $X \backslash Y \backslash Z$, где X, Y, $Z \subset U$ (в частном случае $Y, Z \subset X$).

Эффективность преобразования первой группы рассмотрим на примере операции строгого включения. В эквивалентности выра-

жения $X \subset Y = Z$ выражению $X \backslash Z \subset Y$ нетрудно убедиться. Действительно:

$$X \subset Y = Z \Leftrightarrow (\forall x \in X)(x \in Y \vee x \in Z)$$

и

$$X \backslash Z \subset Y \Leftrightarrow (\forall x \in X \& x \notin Z)(x \in Y).$$

Число сравнений элементов множеств X, Y и Z при реализации выражения $X \subset Y = Z$ равно $k_{op1} = |X| \times (|Y| + |Z|)$. Для выражения $X \backslash Z \subset Y$ при $|X \cap Z| \neq \varnothing$ число операций сравнения $k_{op2} = |X| \times$ $\times |Z| + (|X| - |X \cap Z|) \times |Y|$, откуда $k_{op1} - k_{op2} = |X \cap Z| \times |Y|$.

Докажем эквивалентность выражений $X \backslash \{Y \cup Z\}$ выражению $X \backslash Y \backslash Z$. Известно, что $X \backslash \{Y \cup Z\} = X \cap \rceil \{Y \cup Z\}$. По закону де Моргана $\rceil \{Y \cup Z\} = \rceil Y \cap \rceil Z$. Тогда $X \backslash \{Y \cup Z\} = X \cap \rceil Y \cap \rceil Z$, но $X \cap \rceil Y = X \backslash Y$. Отсюда $X \backslash \{Y \cup Z\} = X \backslash Y \cap \rceil Z$. С учетом $\{X \backslash Y\} \cap \rceil Z = \{X \backslash Y\} \backslash Z$ окончательно получим $X \backslash \{Y \cup Z\} = X \backslash Y \backslash Z$.

Покажем целесообразность замены выражения $X \backslash \{Y \cup Z\}$ на $X \backslash Y \backslash Z$. Число сравнений элементов множеств X, Y и Z при $Y \cap Z = \varnothing$ оценивается по формуле $k_{op1} = |Y| \times |Z| + |X| \times (|Y| + |Z|)$. Число операций сравнения для выражения $X \backslash Y \backslash Z$ будет максимальным в предположении, что $X \cap Y = \varnothing$, $X \cap Z = \varnothing$ и $Y \cap Z = \varnothing - k_{op2} = |X| \times$ $\times |Y| + |X| \times |Z|$. Следовательно, $k_{op1} - k_{op2} = |Y| \times |Z|$.

Рассмотрим случай Y, $Z \subset X$. При $Y \cap Z = \varnothing$ число операций для выражения $X \backslash \{Y \cup Z\}$ равно $k'_{op1} = k_{op1}$ и для выражения $X \backslash Y \backslash Z k'_{op2} = |X| \times |Y| + (|X| - |Y) \times |Z|$, откуда $k'_{op1} - k'_{op2} = (|X| +$ $+ |Y|) \times |Z|$.

Является ли выполненный нами анализ возможных преобразований в данном случае достаточно полным и обеспечивающим тем самым достижение поставленной цели — максимальное снижение вычислительной сложности за счет преобразования данного вида? Вернемся к замене $X \subseteq Y = Z$ на $\{X \backslash Z\} \subseteq Y$. Ответ будет положительным, если не существует других эквивалентных условий. Однако из теории множеств [2] известно, что следующие условия эквивалентны:

$$A \subseteq B \sim A \cap B = A \sim A \cup B = B \sim A \backslash B = \varnothing \sim \rceil A \cup B = U,$$

где U — универсум, под которым будем понимать все элементы данного вида, находящиеся в рассмотрении.

С учетом сказанного список эквивалентных фрагментов, которые могут быть использованы для проверки условия в нашем алгоритме, будет следующим:

1) $X \subseteq Y = Z$, 2) $\{X \backslash Z\} \subseteq Y$, 3) $\{X \backslash Z\} \cap Y = \{X \backslash Z\}$,
4) $\{X \backslash Z\} \cup Y = Y$, 5) $\{X \backslash Z\} \backslash Y = \varnothing$, 6) $\rceil \{X \backslash Z\} \cup Y = U$.

Оценка числа операций сравнения, необходимых для проверки условия по фрагментам 3–6, показывает, что k_{op3}, k_{op4}, $k_{op6} > k_{op2}$ и $k_{op5} = k_{op2}$, где k_{opi} — число сравнений элементов множеств X, Y, Z, необходимых для проверки условия по i-й формуле. Таким образом, фрагменты 1, 3, 4 и 6 являются заменяемыми, а фрагмент 2 или 5 — заменяющим.

Продолжим анализ. Операция $X \backslash Z$ эквивалентна $X \cap \rceil Z$, отсюда получаем следующий набор логически эквивалентных фрагментов:

$$7)\ X \cap \rceil Z \subseteq Y,\ 8)\ \{X \cap \rceil Z\} \cap Y = X \cap \rceil Z,\ 9)\ \{X \cap \rceil Z\} \cup Y = Y,$$
$$10)\ \{X \cap \rceil Z\} \backslash Y = \varnothing,\ 11)\ \{X \cap \rceil Z\} \cap \rceil Y = \varnothing,\ 12)\ \rceil \{X \cap \rceil Z\} \cup Y = U.$$

Все эти фрагменты также являются заменяемыми. Покажем это. Для определения $X \backslash Z$ и $X \cap \rceil Z$ требуется выполнить $|X| \times |Z|$ и $|X| \times$ $\times\, |Z| + (|X| - |Z|) \times |X| = |X|^2$ операций сравнения соответственно. С учетом того, что $Z \subset X$, справедливость утверждения очевидна.

В качестве примера преобразований второй группы можно привести замену выражения $x_i \in X_1$ на $x_i \notin X_2$ и выражения $X_i \subset X_1$ на $X_i \cap X_2 = \varnothing$, $X_i \cap X_1 = \varnothing$ на $X_i \subseteq X_2$, если $X_2 = X \backslash X_1$, $|X_1| > |X|_2$, $x_i \in X$ или $X_i \subset X$. Очевидно, что при использовании этого преобразования число операций сравнения сокращается в $k = |X_1|/|X_2|$ раз. Способ легко формализовать.

Два следующих способа снижения вычислительной сложности основаны на использовании свойств операций над множествами, заданными перечислением.

Третий способ — выбор порядка выполнения операции пересечения более чем двух множеств. Заданы множества $X, Y, Z \subset U$. Обозначим $|X| = n$, $|Y| = m$, $|Z| = k$. Преобразование основано на использовании свойств коммутативности и ассоциативности операции пересечения и предположении, что число сравнений при пересечении более чем двух множеств можно сократить за счет выбора порядка выполнения операции над парой множеств. Напомним, что упомянутые законы для трех множеств имеют соответственно вид

$$X \cap Y \cap Z = Y \cap X \cap Z = \ldots \text{ и } (X \cap Y) \cap Z = X \cap (Y \cap Z) = (X \cap Z) \cap Y.$$

В табл. 1 приведены результаты анализа трех возможных вариантов порядка определения пересечения множеств X, Y, Z.

Положим для определенности, что $n > m > k$. В общем случае второй вариант порядка выполнения операции предпочтительнее первого, если $nm + |X \cap Y|k > mk + |Y \cap Z|n$; третий вариант порядка выполнения операции предпочтительнее первого, если $nm +$ $+\, |X \cap Y|k > nk + |X \cap Z|m$.

Рассмотрим некоторые частные случаи. Минимальное число операций сравнения для первого порядка определения пересечения трех

Таблица 1

Зависимость суммарного числа операций сравнения от порядка нахождения пересечения множеств X, Y, Z

№ варианта	Порядок определения	Число сравнений	Мощность множества	Суммарное число сравнений F_i
1	$D = X \cap Y$, $D \cap Z$	nm $r_1 k$	$r_1 = \lvert X \cap Y \rvert$	$nm + r_1 k$
2	$D = Y \cap Z$, $X \cap D$	mk nr_2	$r_2 = \lvert Y \cap Z \rvert$	$mk + r_2 n$
3	$D = X \cap Z$, $D \cap Y$	nk $r_3 m$	$r_3 = \lvert X \cap Z \rvert$	$nk + r_3 m$

множеств равно nm при $X \cap Y = \varnothing$, максимальное для второго и третьего равно $k(n + m)$ при $Z \subset Y$. *Таким образом, в указанных предположениях преобразование эффективно, если* $nm > k(n + m)$.

В случае, если $Z \subset Y \subset X$, имеем: $r_1 = m$, $r_2 = k$ и $r_3 = k$. Тогда $F_2 = F_3$, $F_1 > F_2$, $F_1/F_2 = (n/k + 1)/(n/m + 1) > 1$, так как $m > k$ и $F_1 - F_2 = n(m - k)$.

При большом числе множеств целесообразно решение задачи выбора оптимального порядка пересечения множеств по критерию минимума числа операций сравнения решать методом динамического программирования.

Пример использования преобразования: для декомпозированной на R подсистем структуры известны цепи, связывающие пары подсистем — ребра гиперграфа $U_{i,j}$, $i, j = 1, R$, $i \neq j$. Необходимо определить цепи, общие для всех подсистем.

Четвертый способ использует свойство дистрибутивности операций над множествами: $(X \cup Y) \cap (X \cup Z) = X \cup (Y \cap Z)$. Для $(X \cup Y) \cap (X \cup Z)$ минимальное число операций сравнения будет, если $Y, Z \subset X$:

$$F_1 = nm + nk + n^2.$$

Максимальное число операций сравнения при определении $X \cup \cup (Y \cap Z)$ будет, если $Y = Z$: $F_2 = m^2 + n \times m$. Тогда $\Delta F_{1,2} = n^2 + + nk - m^2$. Преобразование эффективно, если $n^2 + nk > m^2$.

Пример применения преобразования: в схеме заданы три части $\text{Сх}_1(\text{Э}_1, \text{С}_1)$, $\text{Сх}_2(\text{Э}_2, \text{С}_2)$, $\text{Сх}_3(\text{Э}_3, \text{С}_3)$ — три куска гиперграфа $H_1^{\text{к}}(X, U_1)$, $H_2^{\text{к}}(Y, U_2)$, $H_3^{\text{к}}(Z, U_3)$. Необходимо определить множество элементов, общих для соединения Сх_1 с Сх_2 и Сх_1 с Сх_3.

Оптимизирующие преобразования, использующие свойства предикатов и операций над ними. *Пятый способ заключается в задании предикатами или соответствующими им отношениями таких связей между множествами, которые позволяют снизить вычислительную сложность операций над этими множествами.* Примером

может служить использование вектора специальных признаков для определения принадлежности элементов множества тому или иному подмножеству его разбиения, отношения D "координата элемента в записи множества B соответствует координате ассоциированного с ним элемента в записи множества C".

Выполним обоснование и оценку эффективности использования предиката на примере определения принадлежности элемента некоторому подмножеству разбиения множества. Имеется разбиение множества $X = \{x_i/i = 1,n\}$ на m непересекающихся подмножеств $Y = \{Y1, Y2, \ldots, Ym\}$, $Yj \subseteq X$. Напомним, что для всех $j, r \in J = 1,m$

$$|Yj|, |Yr| \geq 1,\ Yj \cap Yr = \varnothing \ \text{ и } \ \bigcup_{j \in J} Yj = X.$$

В том случае, когда множество Y задано перечислением, определение подмножества, которому принадлежит некоторый элемент x_i множества X, может потребовать в худшем случае n операций его сравнения с $x_t \in Yj$ для всех $j \in J$. Таким образом, асимптотическая сложность выполнения этой операции в худшем равна $O(n)$. Из свойств разбиения множества следует, что каждый элемент множества может принадлежать только одному подмножеству и подмножество может включать более одной вершины. Следовательно, разбиение Y множества X — это сюръективное отношение из X в Y. Отсюда принадлежность элементов x_i подмножествам разбиения Y может быть задана образом PX множества X относительно соответствующего этому отношению предиката $P(X, Y)$:

$$PX = \{Px_i/x_i \in X\},\ Px_i = \{Yj \in Y : Px_i(Yj) = \text{"и"}\},\ \text{где } Yj \subseteq X.$$

Для разбиения $Y = \{Y1, Y2, Y3\}$, где $Y1 = \{x_1, x_3, x_5\}$, $Y2 = \{x_4, x_6\}$, $Y3 = \{x_2\}$ множества $X = \{x_1, x_2, x_3, x_4, x_5, x_6\}$ образ этого множества будет

$$PX = \{Y1, Y3, Y1, Y2, Y1, Y2\}.$$

Экономнее задать сюръективное отношение из множества X в множество индексов подмножеств разбиения Y. Тогда получим множество признаков — номеров подмножеств разбиения

$$B = \{b_i/i = 1,n\}, \ \text{ где } \ b_i = j, \ \text{ если } \ x_i \in Yj.$$

Если множество PX или B представлено вектором, то вычислительная сложность операции определения подмножества, которому принадлежит элемент x_i, равна $O(1)$.

Данный способ был использован в алгоритме Краскала [3] и в дальнейшем для построения процедур абстрактного типа данных — "множество" [4]. Применение этого способа в алгоритмах, выполняющих

соединение компонент связности добавлением ребер, кроме проверки условия соединения разных компонент требует коррекции имен или номеров компонент в PX или B соответственно.

Вычислительная сложность операции коррекции равна $O(n)$. Покажем это. Пусть рассматривается операция добавления в неориентированный граф, состоящий из нескольких компонент связности, ребра u_j, у которого $\Gamma u_j = \{x_3, x_6\}$, $\Gamma u_j \in \Gamma U$. Определив, что $Px_3 \neq Px_6$, мы установили возможность соединения компонент связности, множества вершин которых $Y1 = \{x_1, x_3, x_5\}$ и $Y2 = \{x_4, x_6\}$ соответственно. Если подмножеству вершин новой компоненты связности мы хотим дать имя $Y1$, то образы Px_i необходимо переопределить:

$$Px_i = Y1 : Px_i = Y2/x_i \in X.$$

Таким образом, вычислительная сложность слияния n компонент, каждая из которых содержала по одному элементу, будет равна $O(n^2)$.

Для уменьшения числа операций просмотра элементов образа PX нужно иметь возможность за одну операцию определять первый элемент каждого подмножества и непосредственно переходить от текущего элемента подмножества к следующему. Для достижения этой цели можно задать два предиката. Первый из них должен позволять с вычислительной сложностью $O(1)$ находить начальный элемент подмножества, а второй — для каждого элемента подмножества указывать следующий.

Обозначим эти предикаты $F(Y, \{Yj_1\})$ и $N(X, X)$. Истинность $F(Yj, x_k)$ означает, что первым элементом подмножества Yj при задании его перечислением элементов является элемент x_k. Значение $N(x_i, x_l)$ будет истинным, если для элемента x_i следующим при перечислении элементов подмножества будет элемент x_l. Для нашего примера образом множества Y относительно предиката $F(Y, \{Yj_1\})$ будет $FY = \{x_1, x_4, x_2\}$, а образом множества X относительно предиката $N(X, X)$ будет $NX = \{x_3, 0, x_5, x_6, 0, 0\}$.

Число операций слияния подмножеств зависит от соотношения мощностей подмножеств и порядка их переименования. Если последовательно выполнять слияние двух подмножеств, первое из которых состоит из одного элемента, а второе из i элементов, где $i = 1, n-1$, то число операций переименования будет равно $n(n-1)/2$. Число этих операций можно существенно сократить, если при слиянии двух подмножеств переименовывать подмножество меньшей мощности. Тогда при соединении двух компонент связности ребром u_j, у которого $\Gamma u_j = \{x_l, x_k\}$ и $Px_l \neq Px_k$, необходимо проверить $|Yt| > |Yr|$, где $Yt = Px_l$ и $Yr = Px_k$, и при удовлетворении этого условия выполнить процедуру

$$(\forall x_i \in Yr) Px_i := Px_l,$$

а в противном случае — процедуру

$$(\forall x_i \in Yt)Px_i := Px_k.$$

Как показано в [5], в этом случае любой элемент может перейти в любое подмножество не более чем за $1 + \log_2 n$ шагов. Теперь вычислительная сложность всех слияний будет $O(n \log_2 n)$ — сравните с полученной ранее оценкой, равной $O(n^2)$. Формализация данного способа и автоматизация его применения представляются сложными, но разрешимыми задачами.

Шестой способ — определение результата операции над характеристическими множествами одноместных предикатов как характеристического множества результата операции над ними. При задании множеств коллективизирующим свойством (характеристическим предикатом) объединение, пересечение, дополнение, симметрическую разность их характеристических множеств целесообразно определять как характеристические множества соответствующих логических операций над этими предикатами. Рассмотрим этот способ на примере операции объединения.

Пусть множества P и Q заданы предикатами $P(X)$ и $Q(X)$. Необходимо определить множество $R = P \cup Q$. Проанализируем два варианта определения множества R.

1. Находим множества P и Q как характеристические множества предикатов в соответствии с выражениями

$$P = \{x_i : P(x_i) = \text{“и”}/x_i \in X\} \text{ и } Q = \{x_i : Q(x_i) = \text{“и”}/x_i \in X\}.$$

Затем выполняем операцию их объединения: $R = P \cup Q$.

Число операций сравнения, необходимых для определения множества R, равно

$$F_1 = (k_1 + k_2)|X| + |P| \times |Q|,$$

где k_1 и k_2 — число операций, необходимых для определения истинности $P(x_i)$ и $Q(x_i)$ соответственно.

2. Определяем характеристическое множество предиката $R(X) = P(X) \vee Q(X)$:

$$R = \{x_i : P(x_i) = \text{“и”} \vee Q(x_i) = \text{“и”}/x_i \in X\}.$$

В этом случае число операций сравнения, необходимых для определения множества R, равно

$$F_2 = (k_1 + k_2 + 1)|X|.$$

Очевидно, что применение данного преобразования целесообразно при $|P| \times |Q| > |X|$.

В табл. 2 приведены логические операции над предикатами, характеристические множества которых равны результатам операций над множествами, указанным в первом столбце.

Таблица 2

Соответствие операций над множествами операциям над предикатами

Операции над множествами	Логические операции над предикатами
$P \cup Q$	$P(X) \vee Q(X)$
$P \cap Q$	$P(X)\&Q(X)$
$P \backslash Q$	$P(X)\&\neg Q(X)$
$P \Delta Q$	$P(X)\Delta Q(X) = (P(X) \vee Q(X))\&\neg(P(X)\&Q(X))$

Пример использования такого преобразования в алгоритме структурного синтеза. Решается задача декомпозиции структуры системы на две подсистемы с равным числом компонентов. Моделью структуры системы является гиперграф, представленный в форме $H(X, U, \Gamma X, \Gamma U)$, $|X| = n$, $|U| = m$. В результате работы алгоритма разрезания гиперграфа на два куска $H_1^{\text{к}}(X_1, U_1)$ и $H_2^{\text{к}}(X_2, U_2)$ получены множества X_1 и X_2 ($|X_1| = |X_2| = n/2$). Рассмотрим только определение множества связей между подсистемами. Напомним, что по определению куска графа

$$U_1 = \{U_{1,1}, U_{1,2}\}, \quad U_2 = \{U_{2,2}, U_{2,1}\},$$

$$U_{1,1} \cap U_{1,2} = \varnothing, \quad U_{2,2} \cap U_{2,1} = \varnothing, \quad U_{1,2} = U_{2,1} = U_1 \cap U_2,$$

где $U_{1,1}$ и $U_{2,2}$ — внутренние ребра кусков $H_1^{\text{к}}$ и $H_2^{\text{к}}$; $U_{1,2}$ и $U_{2,1}$ — внешние ребра этих кусков.

Свойство, определяющее принадлежность ребра u_j множеству U_1 (U_2), можно сформулировать, например, так: “хотя бы одна вершина множества $X_1(X_2)$ и ребро u_j инцидентны”. При представлении множеств $X, U, \Gamma X, \Gamma U$ перечислением предикаты $P_1(U)$ и $P_2(U)$, характеристическими множествами которых являются U_1 и U_2 соответственно, запишем как

$$P_1(u) = \text{“}\Gamma u \cap X_1 \neq \varnothing\text{”} \text{ и } P_2(u) = \text{“}\Gamma u \cap X_2 \neq \varnothing\text{”}.$$

Рассмотрим 1-й и 2-й варианты определения множеств U_1, U_2 и $U_{1,2}$:

$$U_1 = \{u_j : \Gamma u_j \cap X_1 \neq \varnothing / u_j \in U, \quad \Gamma u_j \in \Gamma U\};$$

$$U_2 = \{u_j : \Gamma u_j \cap X_2 \neq \varnothing / u_j \in U, \quad \Gamma u_j \in \Gamma U\}, \quad U_{1,2} = U_1 \cap U_2.$$

Число операций сравнения, необходимых для определения множества $U_{1,2}$, равно

$$F_1 = (k_1 + k_2)|U| + |U_1| \times |U_2|, \text{ где } k_1 = |\Gamma u_j| \times |X_1| \text{ и } k_2 = |\Gamma u_j| \times |X_2|.$$

Для разрезания на куски справедливо: $|U_{1,1}| + |U_{2,2}| + |U_{1,2}| = |U|$, $|U_1| = |U_{1,1}| + |U_{1,2}|$ и $|U_2| = |U_{2,2}| + |U_{2,1}|$. На основании закона Рента имеем $|U_{1,2}| = |\Gamma x| \times |X_1|^{0,5}$. Принимая $|\Gamma u| = A$, $|\Gamma x| = \rho$ и $|U_1| = |U_2|$,

получаем $k_1 = k_2 = A(n/2)$, $|U_{1,2}| = \rho(n/2)^{0,5}$ и $|U_1| = |U_2| = (m + \rho(n/2)^{0,5})/2$. Окончательно имеем

$$F_1 = Anm + (m + \rho(n/2)^{0,5})^2/4.$$

Предикат, задающий принадлежность ребра u как множеству U_1, так и U_2, будет иметь вид

$$P_{1,2}(u) = \text{“}\Gamma u \cap X_1 \neq \varnothing\text{”} \& \text{“}\Gamma u \cap X_2 \neq \varnothing\text{”}.$$

Его характеристическое множество определяется по формуле

$$U_{1,2} = \{u_j : \Gamma u_j \cap X_1 \neq \varnothing \& \Gamma u_j \cap X_2 \neq \varnothing / u_j \in U, \ \ \Gamma u_j \in \Gamma U\}.$$

Число операций сравнения, необходимых для определения множества $U_{1,2}$, равно $F_2 = Anm$.

Нетрудно видеть, что число операций сравнения в этом случае меньше на $\Delta F = (m + \rho(n/2)^{0,5})^2/4$.

Седьмой способ основан на использовании операции композиции над двуместными предикатами. Заданы двуместные предикаты $\Gamma_1(X, U)$ и $\Gamma_2(U, Z)$. Необходимо определить образы элементов x относительно предиката $F_1(X, Z) = \Gamma_1(X, U) \bullet \Gamma_2(U, Z)$; здесь $\bullet$ — символ операции композиции, где $F_1(x, z) =$ “и” : $\exists u(\Gamma_1(x, u) =$ “и” $\& \Gamma_2(u, z) =$ “и”).

Пусть $X = \{x_1, x_2, x_3\}$, $U = \{u_1, u_2, u_3, u_4\}$, $Z = \{z_1, z_2, z_3\}$ и предикаты принимают значение “истина” на парах

$$\Gamma_1(X, U) : (x_1, u_1), \ (x_1, u_2), \ (x_2, u_3), \ (x_3, u_4);$$

$$\Gamma_2(U, Z) : \ (u_1, z_2), \ (u_2, z_2), \ (u_2, z_3), \ (u_3, z_3), \ (u_4, z_1), (u_4, z_2).$$

На рис. 2 показана геометрическая интерпретация предикатов $\Gamma_1(X, U)$, $\Gamma_2(U, Z)$ и $F_1(X, Z)$.

Образ элемента $x_i \in X$ относительно предиката $F_1(X, Z)$ будет определяться по правилу

$$F_1 x_i = \{z_k : \exists u_j(\Gamma_1(x_i, u_j) = \text{“и”} \& \ \Gamma_2(u_j, z_k) = \text{“и”}) / u_j \in U, \ z_k \in Z\}.$$

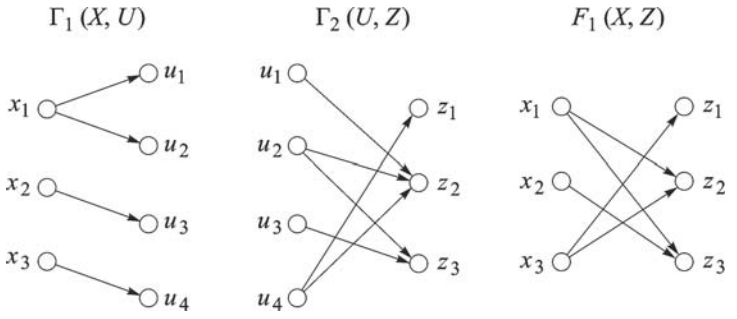

Рис. 2. Геометрическая интерпретация предикатов $\Gamma_1(X, U)$, $\Gamma_2(U, Z)$ и их композиции $F_1(X, Z)$

Для перехода к моделям задач структурного синтеза приравняем множество Z к множеству X. Множества X и U определим как вершины и ребра графа соответственно. Тогда композиция предикатов $\Gamma_1(X,U)$ и $\Gamma_2(U,X)$ — это предикат смежности графа $F_1(X,X) = \Gamma_1(X,U) \bullet \Gamma_2(U,X)$.

Переходя к образам элементов множества X и U относительно предикатов $\Gamma_1(X,U)$ и $\Gamma_2(U,X)$ соответственно, получаем, что смежность вершины x_i определяется как

$$F_1 x_i = \{x_k : \exists u_j (u_j \in \Gamma_1 x_i \& x_k \in \Gamma_2 u_j)/u_j \in U,\ x_k \in X\}.$$

Логически эквивалентным выражению $\exists u_j (u_j \in \Gamma_1 x_i \& x_k \in \Gamma_2 u_j)$ будет

$$\Gamma_1 x_i \cap \Gamma_2^{-1} x_k \neq \varnothing.$$

Теперь $F_1 x_i(X) = \{x_k : \Gamma_1 x_i \cap \Gamma_2^{-1} x_k \neq \varnothing / x_k \in X\}$.

Оценим число операций сравнения, необходимых для определения образа вершины x_i относительно предиката смежности двумя способами — как объединение подмножеств вершин, инцидентных ребрам $u_j \in \Gamma_1 x_i$, и как характеристическое множество предиката $F_1 x_i(X)$.

В первом случае образ определяется по формуле

$$F_1 x_i = \bigcup_{u_j \in \Gamma_1 x_i} \Gamma_2 u_j,$$

Число операций сравнения элементов множеств $\Gamma_2 u_j$ будет равно

$$K_1 = |\Gamma_2 u_j|^2 \sum_{i=2}^{|\Gamma_1 x_i|} [1 + c(i-2)],$$

где $0 \le c < 1$ — коэффициент, учитывающий возможность того, что у подмножеств $\Gamma_2 u_j$ и $\Gamma_2 u_r$, принадлежащих $\Gamma_1 x_i$, могут быть общие вершины. При $|\Gamma_1 x_i| = \rho$ и $|\Gamma_2 u_j| = A$ число операций сравнения $K_1 = f(A^2\rho^2)$. Для определения $\Gamma_2 u_j$ и $\Gamma_1 x_i$ потребуется менее, чем $(n+m)$ операций проверки $\Gamma_2 u_j(x_i) =$ “и” и $\Gamma_1 x_i(u_j) =$ “и". Тогда $K_{1\Sigma} < (n + m + f(A^2\rho^2))$.

Если $|\Gamma_1 x_i|$ и $|\Gamma_2 u_j|$ ограничены m и n соответственно, то асимптотическая оценка будет $O_1(m^2 n^2)$.

Определение $F_1 x_i$, как характеристического множества предиката $F_1 x_i(X)$, потребует $K_2 = |\Gamma_1 x_i| \times |\Gamma_2^{-1} x_k|$ n операций сравнения. При $|\Gamma_1 x_i| = |\Gamma_2^{-1} x_k| = \rho$ число операций сравнения $K_2 = \rho^2 n$. Если $|\Gamma_1 x_i|$ ограничена m, то асимптотическая оценка будет $O_2(m^2 n)$.

Восьмой способ — использование свойств логических операций над предикатами. Рассмотрим использование свойства коммутативности при определении характеристических множеств. Пусть необходимо определить характеристические множества предикатов

$$T_1(X) = P(X) \& Q(X),\ \ T_2(X) = P(X) \backslash Q(X) = P(X) \& \rceil Q(X)$$

и

$$T_3(X) = P(X) \vee Q(X).$$

Их характеристические множества имеют вид

$$T_1 = \{x_i : P(x_i) = \text{“и”} \& Q(x_i) = \text{“и”} / x_i \in X\},$$

$$T_2 = \{x_i : P(x_i) = \text{“и”} \& \rceil Q(x_i) = \text{“и”} / x_i \in X\}$$

и

$$T_3 = \{x_i : P(x_i) = \text{“и”} \vee Q(x_i) = \text{“и”} / x_i \in X\}.$$

Операции & и $\vee$ обладают свойством коммутативности:

$$P(X)\&Q(X) = Q(X)\&P(X),\ P(X)\&\rceil Q(X) = \rceil Q(X)\&P(X)$$

и

$$P(X) \vee Q(X) = Q(X) \vee P(X).$$

Очевидно, что при определении T_1 или T_2, если $P(x_i) =$ “л”, то значение $Q(x_i)$ или $\rceil Q(x_i)$ нет необходимости проверять, а при определении T_3 нет необходимости проверять значение $Q(x_i)$, если $P(x_i) =$ “и”. Отсюда следует, что использование свойства коммутативности приведет к сокращению числа операций сравнения, если при определении T_1 или T_2 предикаты $P(x_i)$ и $Q(x_i)$ или $P(x_i)$ и $\rceil Q(x_i)$ соответственно располагать по возрастанию вероятности исхода “истина”, а при определении T_3 предикаты $P(x_i)$ и $Q(x_i)$ располагать по убыванию того же параметра.

Отметим, что при выполнении операций над более чем двумя предикатами их ранжирование правомерно на основании законов ассоциативности и коммутативности. Использование данного преобразования возможно, если существуют или могут быть получены за приемлемое время оценки вероятности исхода “истина” соответствующих предикатов.

Заключение. Выполненные в работе исследования показали высокую эффективность большинства оптимизирующих преобразований данного класса. Это обусловливает целесообразность их широкого использования при разработке алгоритмов на графах и множествах.

ЛИТЕРАТУРА

1. *Овчинников В.А.* Алгоритмизация комбинаторно-оптимизационных задач при проектировании ЭВМ и систем: М.: Изд-во МГТУ им. Н.Э. Баумана, 2001. 288 с.
2. *Судоплатов С.В.*, *Овчинникова Е.В.* Элементы дискретной математики: учебник. М.: ИНФРА-М, Новосибирск: Изд-во НГТУ, 2002. 280 с.
3. *Ахо А.В.*, *Хопкрофт Д.Э.*, *Ульман Д.Д.* Построение и анализ вычислительных алгоритмов: пер. с англ. М.: Мир, 1979. 535 с.
4. *Ахо А.В.*, *Хопкрофт Д.Э.*, *Ульман Д.Д.* Структуры данных и алгоритмы: пер. с англ. М.: Вильямс, 2001. 384 с.

5. *Кормен Т.*, *Лейзерсон Ч.*, *Риверст Р.* Алгоритмы: построение и анализ. М.: МЦНМО, 2000. 960 с.

REFERENCES

[1] Ovchinnikov V.A. Algoritmizatsiya kombinatorno-optimizatsionnykh zadach pri proektirovanii EVM i system [Construction of algorithms for combinatorial optimization problems in computer and system design]. Moscow, MGTU im. N.E. Baumana Publ., 2001. 288 p.

[2] Sudoplatov S.V., Ovchinnikova E.V. Elementy diskretnoy matematiki [Elements of discrete mathematics]. Moscow, INFRA-M Publ., 2002. 280 p.

[3] Aho A.V., Hopcroft J.E., Ullman J.D. The design and analysis of computer algorithms. Addison-Wesley, 1974. (Russ. ed.: Akho A.V., Khopkroft D.E., Ul'man D.D. Postroenie i analiz vychislitel'nykh algoritmov. Moscow, Mir Publ., 1979. 535 p.).

[4] Aho A.V., Hopcroft J.E., Ullman J.D. Data structures and algorithms. Addison-Wesley, 1983. (Russ. ed.: Akho A.V., Khopkroft D.E., Ul'man D.D. Struktury dannykh i algoritmy. Moscow, Vil'yams Publ., 2001. 384 p.).

[5] Cormen T.H., Leiserson C.E., Rivest R.L. Introduction to algorithms. MIT Press, 1990, 1296 p. (Russ. ed.: Kormen T., Leyzerson Ch., Riverst R. Algoritmy: postroenie i analiz. Moscow. MTsNMO Publ., 2000. 960 p.).

Статья поступила в редакцию 25.02.2012

Владимир Анатольевич Овчинников — д-р техн. наук, профессор кафедры "Компьютерные системы и сети" МГТУ им. Н.Э. Баумана. Автор свыше 120 научных работ в области вычислительной техники и проектирования компьютерных систем.
МГТУ им. Н.Э. Баумана, Российская Федерация, 105005, Москва, 2-я Бауманская ул., д. 5.

V.A. Ovchinnikov — Dr. Sci. (Eng.), professor of "Computer Systems and Networks" department of the Bauman Moscow State Technical University. Author of more than 120 publications in the field of computational technology and computer system design.
Bauman Moscow State Technical University, Vtoraya Baumanskaya ul., 5, Moscow, 105005 Russian Federation.

Галина Сергеевна Иванова — д-р техн. наук, профессор кафедры "Компьютерные системы и сети" МГТУ им. Н.Э. Баумана. Автор свыше 50 научных работ в области вычислительной техники и проектирования программных систем.
МГТУ им. Н.Э. Баумана, Российская Федерация, 105005, Москва, 2-я Бауманская ул., д. 5.

G.S. Ivanova — Dr. Sci. (Eng.), professor of "Computer Systems and Networks" department of the Bauman Moscow State Technical University. Author of more than 50 publications in the field of computational technology and software system design.
Bauman Moscow State Technical University, Vtoraya Baumanskaya ul., 5, Moscow, 105005 Russian Federation.