4. Баскаков С. С. Алгоритм автоматического выбора опорных узлов в беспроводных сенсорных сетях // Тр. конф. “Информационные технологии и системы” (ИТиС-2007). – Звенигород: 2007. – С. 2–8.

Статья поступила в редакцию 21.01.2008

Сергей Сергеевич Баскаков родился в 1981 г., окончил МГТУ им. Н.Э. Баумана в 2006 г. Аспирант кафедры “Информационные системы и телекоммуникации” МГТУ им. Н.Э. Баумана. Автор 3 научных работ в области беспроводных сетей и систем связи.

S.S. Baskakov (b. 1981) graduated from the Bauman Moscow State Technical University in 2006. Post-graduate of “Information Systems and Telecommunications” department of the Bauman Moscow State Technical University. Author of 3 publications in the field of wireless networks and communication systems.

---

УДК 004.422.6

К. А. Пасечников, Г. С. Иванова

# СИНТЕЗ ОПТИМАЛЬНЫХ СТРУКТУР ДАННЫХ ДЛЯ РЕШЕНИЯ ЗАДАЧ НА ГРАФАХ

*Предложена модель, позволяющая адекватно отобразить характеристические особенности базовых структур данных, используемых для представления графовых моделей. Формально определена операция объединения базовых структур данных, что позволило автоматизировать расчет временны́х и емкостных параметров полученных комбинированных структур данных. Предложена формальная постановка задачи синтеза оптимальной (с точки зрения минимимизации вычислительной сложности выполнения заданного набора операций) одноуровневой комбинированной структуры данных при условии допустимой емкостной сложности этой структуры.*

При решении задач структурного анализа и синтеза на графах применяется большое число различных структур данных. Они используются для хранения не только основной (информации о графе), но и различной вспомогательной информации. Как было показано в работах [1–4], способ организации элементов структуры данных существенно влияет на вычислительную и емкостную сложности программ решения задач на графах. В работах [2, 3] показано, что применение лишь базовых структур данных для решения задач на графах неэффективно в силу отсутствия такой базовой структуры данных, которая обеспечила бы минимальную вычислительную сложность для любой операции. Следовательно, необходим синтез комбинированных структур данных и выбор из них оптимальной с точки зрения вычислительной сложности.

Для решения задачи автоматического синтеза оптимальных структур данных необходимо разработать такую модель структуры данных, которая отражает структурные особенности различных способов организации данных, существенные с точки зрения вычислительной сложности выполняемых над структурой данных операций, а также ее емкостной сложности.

Любая структура данных представляет собой описание одного из возможных способов размещения конечного множества данных $D$ в памяти. При этом каждому элементу множества данных ставится во взаимно-однозначное соответствие $P$ адрес из множества доступных адресов памяти $A \leftrightarrow D$, $P \subset A \times D$.

Структура данных кроме полезной информации в виде элементов данных обычно хранит служебную информацию о связях элементов между собой. Элемент данных вместе со служебной информацией будем называть *компонентом* структуры данных. Таким образом, можно говорить о наличии множества $C$ компонентов структуры данных и множества $Z$, определяющего взаимно-однозначное соответствие размещенных элементов данных компонентам структуры $P \leftrightarrow C$, $Z \subset P \times C$. Связи $F$ между компонентами структуры данных отобразим, определив однозначное соответствие $C \to C$, $F \subset C \times C$.

Для построения моделей конкретных структур данных рассмотрим основные способы организации данных, лежащие в основе сложных структур данных — векторный и списковый, и определим основные компоненты этих структур, отношения компонентов, их характеристики и свойства.

**Векторный способ организации** предполагает последовательное размещение однотипных (одинаковых по размеру) элементов в непрерывной области памяти. Подобное размещение позволяет определить адрес любого элемента вектора по его номеру, адресу первого элемента и размеру. При этом адрес текущего элемента можно определить по адресу любого элемента, находящегося слева или справа от него. Элементом вектора может быть элемент данных, указатель на данные или структуру данных, ключ для получения данных (в этом случае они хранятся отдельно), признак или флаг (характеристические векторы). Компонентом вектора будет являться непосредственно элемент данных, а множество связей компонентов $F$ будет представлять собой декартово произведение множества $C$ самого на себя, т.е. $F = C \times C$.

При переходе от объекта к модели поставим во взаимно-однозначное соответствие множеству $D$ множество вершин графа $X_D$, а множеству адресов $A$ — множество вершин графа $X_A$. Поскольку множество $P$ описывает взаимно-однозначное соответствие между множествами $A$ и $D$, каждой паре элементов множества $P$ сопоставляем ребро

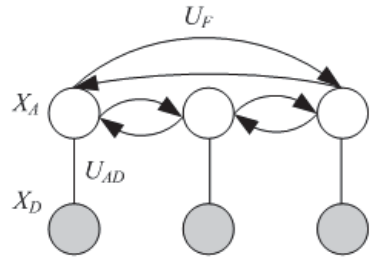


**Рис. 1. Модель вектора:**
● — вершина данных; ○ — вершина адреса; ← — дуга, связывающая вершины адреса; — — ребро, связывающее вершину адреса и вершину данных

$u_{AD} \in U_{AD}$. Таким образом, каждому компоненту $c \in C$ вектора будет взаимно-однозначно соответствовать двухвершинный подграф с одним ребром $g(< x_A,\, x_D >,\, u_{AD}) \in G$, где $G$ — граф, описывающий организацию элементов структуры данных. Каждой связи компонентов из множества $F$ сопоставим дугу из множества $U_F$ (рис. 1).

Рассмотрим способ определения емкостной сложности вектора по модели. Поскольку вектор предназначен для хранения данных фиксированной длины, обозначим размер одного элемента через $l^V$. Количество данных, хранимых вектором, определяется мощностью множества $|X_A| = n$. Сопоставим каждому элементу из этого множества значение веса, равное $l^V$, т.е. $(\forall x \in X_A)\, l^V(x) = l^V$. Каждому элементу множества $X_D$ сопоставим значение веса, равное нулю, т.е. $(\forall x \in X_D)\, l^V(x) = 0$. В результате объем памяти (в байтах), требуемый для хранения вектора, составит

$$E(n) = \sum_{\forall x \in X} l^V(x) = l^V |X_A| = l^V n.$$

При сравнении нескольких структур данных между собой необходимо знать объем памяти $L(n)$, занимаемый вспомогательными элементами каждой структуры данных в зависимости от числа хранимых элементов данных. Далее функцию $L(n)$ будем называть *дополнительной* емкостной сложностью. Очевидно, что функция $L(n)$ будет связана с емкостной сложностью $E(n)$ следующим соотношением:

$$E(n) = L(n) + l^V n,$$

соответственно для вектора $L(n) = 0$.

Можно показать, что вычислительная сложность операций, посчитанная по модели, совпадает с вычислительной сложностью операций над вектором, приведенной в работах [1, 4], что обусловлено адекватностью полученной модели. Для дополнения модели информацией о вычислительной сложности операций определим также множество

функций $Q$ как

$$Q = \{q_i(n) : s_i \to N, i = \overline{1, |S|}\},$$

где $S = \{s_i\}$ — множество допустимых операций над структурой данных; $q_i(n)$ — функция оценки вычислительной сложности $i$-й операции.

Таким образом, модель вектора будет выглядеть следующим образом:

$$M_V = \langle G_V,\, Q,\, L(n)\rangle,$$

где $G_V(\{X_D, \langle X_A, l^V\rangle\}, \{U_{AD}, \overrightarrow{U_F}\})$ — смешанный граф, описывающий организацию элементов; $Q$ — множество функций, определяющих вычислительную сложность операций; $L(n) = 0$ — дополнительная емкостная сложность структуры данных.

**Списковая организация данных** подразумевает размещение элементов списка в произвольных свободных участках памяти. При построении списковых структур с каждым компонентом списка связывается набор вспомогательных полей, обеспечивающий связь отдельных компонентов списка между собой. Так, для линейного односвязного списка компоненты связаны между собой отношением “предыдущий–следующий” за счет хранения в компоненте адреса следующего компонента. Наличие вспомогательных полей требует рассмотрения дополнительных множеств — множества $D'$ вспомогательных элементов данных (указателей) и множества $A'$ — адресов вспомогательных элементов данных. При этом связи между элементами указанных дополнительных множеств представим множеством $P'$, каждый элемент которого определяет соответствие адресов памяти вспомогательным элементам данных, т.е. $A' \leftrightarrow D',\ P' \subset A' \times D'$. Взаимно-однозначное соответствие $P' \leftrightarrow C$ вспомогательных элементов компонентам данных определим как множество $Z'$, такое, что $Z' \subset P' \times C$. Неявные связи между полями в компонентах списка представим как множество $F'$.

Основными структурами со списковой организацией данных, применяемыми для представления графовых моделей и вспомогательных данных, являются линейные односвязные, линейные двусвязные, $n$-связные и древовидные списки [3]. Рассмотрим модель простейшего односвязного списка, когда с каждым элементом списка связан всего один элемент данных. Каждый компонент списка представляет собой запись, а сам список является множеством записей, соединенных между собой.

При переходе от объекта к модели поставим во взаимно-однозначное соответствие множествам $A$, $D$, $A'$, $D'$ соответствующие множества вершин графа $X_A$, $X_D$, $X_{A'}$, $X_{D'}$. Элементы множеств $P$, $P'$

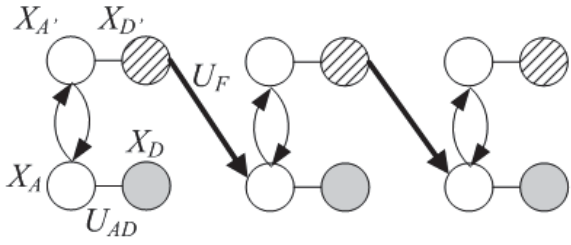


**Рис. 2. Модель односвязного списка:**
○ — вершина данных; ▨ — вспомогательная вершина; ● — вершина адреса; ⟵ — дуга, связывающая вершины адреса; ⟵ — ребро, связывающее вершину адреса и вершину данных; ⟵ — дуга, связывающая компоненты списка между собой

взаимно-однозначно сопоставим с ребрами из множеств $U_{AD}$, $U_{AD'}$. Связи между полями записей, задаваемые множеством $F'$, отобразим на множество дуг $\overrightarrow{U_A}$. Связи компонентов между собой, задаваемые множеством $F$, отобразим на множество дуг $\overrightarrow{U_F}$. Таким образом, множеству компонентов односвязного списка $C$ взаимно-однозначно соответствует множество подграфов $\{g(\langle x_A, x_D, x_{A'}, x_{D'}\rangle, \langle\overrightarrow{u_A}, u_{AD}, u_{AD'}\rangle)\in \in G\}$ (рис. 2).

Выполним оценку емкостной сложности связного списка по модели. Количество данных, хранимых списком, определяется числом элементов множества $X_A$, поэтому сопоставим каждому элементу из этого множества значение веса, равное $l^V$, т.е. $(\forall x \in X_A)\, l^V(x) = l^V$. Число дополнительных указателей, которые содержит список, определяется числом элементов множества $X_{A'}$, поэтому сопоставим каждому элементу из этого множества значение веса, равное $l^A$, т.е. $(\forall x \in X_{A'})\, l^V(x) = l^A$. Каждому элементу множеств $X_D, X_{D'}$ сопоставим значение веса, равное нулю, т.е. $(\forall x \in X_D)\, l^V(x) = 0$, $(\forall x \in X_{D'})\, l^V(x) = 0$. Тогда объем памяти (в байтах), требуемый для хранения списка, составит

$$E(n) = \sum_{\forall x \in X} l^V(x) = l^V|X_A| + l^A|X'_A| = nl^V + nl^A.$$

По аналогии с вектором модель списка будет выглядеть следующим образом:

$$M_{\text{S}} = \langle G_S, Q, L(n)\rangle,$$

где $G_S(\{X_D, X_{D'}, \langle X_A, l^V\rangle, \langle X_{A'}, l^A\rangle\}, \{U_{AD}, U_{AD'}, \overrightarrow{U_A}, \overrightarrow{U_F}\})$ — смешанный граф, описывающий организацию элементов в структуре данных; $Q$ — множество функций, определяющих вычислительную сложность операций; $L(n) = nl^A$ — дополнительная емкостная сложность списка.

Полученные модели базовых структур данных являются более простыми, чем модели, предложенные в работе [2], так как они не содержат избыточной с точки зрения поставленной задачи информации.

**Операция объединения моделей структур данных.** В результате объединения двух и более базовых структур данных получается новая комбинированная структура. Очевидно, что результирующая и исходные структуры должны описывать размещение одних и тех же элементов данных в памяти, т.е. набор данных и для исходных структур, и для структуры-результата должен быть один и тот же.

В качестве примера рассмотрим объединение древовидного списка $M_1$ и вектора $M_2$, которые описываются следующим образом:

$$M_1 = \langle G_1, Q_1, L_1(n)\rangle,$$

$$G_1 = G(X_1, U_1) = \\ = G(\{X_D, X_{D1}, \langle X_A, l^V\rangle, \langle X_{A1}, l^A\rangle\}, \{U_{AD}, U_{AD1}, \overrightarrow{U_{F1}}, U_1'\}),$$

$$M_2 = \langle G_2, Q_2, L_2(n)\rangle,$$

$$G_2 = G(X_2, U_2) = G(\{X_D, \langle X_A, l^V\rangle\}, \{U_{AD}, \overrightarrow{U_{F2}}\}),$$

где $G_1$, $G_2$ — графы организации элементов древовидного списка и вектора; $Q_1$, $Q_2$ — множества функций, определяющих вычислительную сложность операций над структурой данных; $L_1(n)$, $L_2(n)$ — дополнительные объемы памяти для организации древовидного списка и вектора; $X_D$, $X_A$ — множество вершин, соответствующих элементам данных и их адресам; $U_{AD}$ — множество дуг, отражающих взаимнооднозначное соответствие между элементами данных и их адресами; $X_{D1}$, $X_{A1}$, $U_{AD1}$ — множества вершин и дуг, описывающие дополнительные поля в компонентах древовидного списка; $\overrightarrow{U_{F1}}$, $\overrightarrow{U_{F2}}$ — множества дуг, отражающих связи компонентов структуры данных между собой; $U_1'$ — множество дуг, отражающие связи между полями внутри компонента.

Графы $G_1$ и $G_2$ имеют общие вершины $X_A$, $X_D$ и ребра $U_{AD}$, что позволяет выполнить операцию их объединения. Результат объединения представлен на рис. 3.

Подсчитаем вычислительную и емкостную сложности комбинированной структуры данных. В ходе объединения моделей не было введено никаких вспомогательных конструкций, следовательно, дополнительная емкостная сложность комбинированной структуры данных составит

$$L(n) = L_1(n) + L_2(n).$$

При подсчете вычислительной сложности необходимо учитывать, что операции доступа можно выполнять над любой структурой данных, входящих в комбинированную структуру (так как все они описывают расположение одних и тех же данных в памяти). Однако опера-

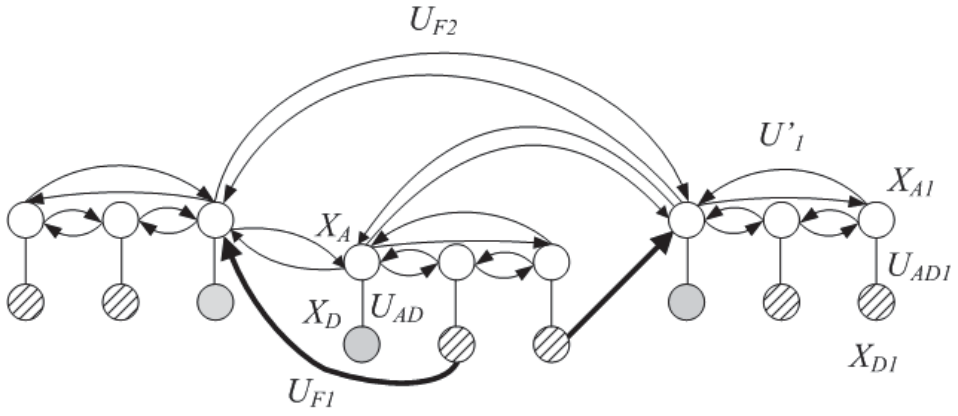


**Рис. 3. Результат объединения моделей древовидного списка и вектора**

ции преобразования необходимо выполнять над всеми составляющими комбинированной структуры данных. С учетом этого представим множество всех возможных операций над структурами данных $S$ как объединение двух подмножеств — подмножества операций доступа $S_1$ и подмножества операций преобразования $S_2$: $S = S_1 \cup S_2$. Тогда вычислительная сложность комбинированной структуры данных имеет вид:

$$Q^* = Q_{S1} \cup Q_{S2};$$

$$Q_{S1} = \{q_i(n) = \min[q_{1_i}(n), q_{2i}(n)]\},$$

$$Q_{S2} = \{q_i(n) = q_{1_i}(n) + q_{2i}(n)\}.$$

Таким образом, комбинированной структурой данных, полученной в результате объединения двух структур данных, задаваемых моделями $M_1$, $M_2$, будет структура данных, описываемая моделью

$$M = \langle G^*, Q^*, L^*(n)\rangle,$$

где

$$G^* = G_1 \cup G_2 =$$

$$= G(\{X_D, \langle X_A, l^V\rangle, X_{D1}, \langle X_{A1}, l^A\rangle\}, \{U_{AD}, U_{AD1}, \overrightarrow{U_{F1}}, U'_1, \overrightarrow{U_{F2}}\}),$$

$$L(n) = L_1(n) + L_2(n);$$

$$Q^* = Q_{S1} \cup Q_{S2};$$

$$Q_{S1} = \{q_i(n) = \min[q_{1_i}(n), q_{2i}(n)]\};$$

$$Q_{S2} = \{q_i(n) = q_{1_i}(n) + q_{2i}(n)\}.$$

Предложенный способ объединения структур данных позволяет синтезировать сложные комбинированные структуры данных, а также

дает возможность автоматически определять их вычислительные и емкостные параметры.

**Формальная постановка задачи синтеза оптимальных комбинированных структур данных.** Исходными данными [2] для синтеза оптимальной с точки зрения вычислительной сложности заданного алгоритма структуры данных являются:

1) описание множества и размер элемента его данных;

2) совокупность базовых структур данных $B = \{b_i = \langle G_i, Q_i, L_i(n)\rangle\}$;

3) совокупность операций, выполняемых над множеством, что позволит определить набор операций над данными структуры;

4) число повторений каждой операции над множеством в алгоритме (множество функций $R = \{r(n)\colon S \to N\}$);

5) допустимый объем памяти $E_{\text{доп}}(n)$, который может занять синтезируемая структура.

При этом операции, выполняемые над множествами, должны быть представлены как базовые, поскольку базовые операции, реализующие основные отношения между элементами множества, независимы с точки зрения их вычислительной сложности.

На основе данных, приведенных в п. 1 и 2, можно сформировать множество вариантов моделей структур данных, которые получаются посредством объединения тех или иных базовых структур, и определить их характеристики. Из полученного множества необходимо выбрать такую структуру данных, которая для заданного набора операций обеспечивает минимальную вычислительную сложность и емкостная сложность которой не превышает допустимого значения. Отсюда получаем формальную постановку задачи синтеза оптимальной структуры данных в следующем виде:

$$\begin{aligned} &M_{\text{opt}} = \langle G^*, Q^*, L^*(n)\rangle \in \text{M};\\ &G^* = \bigcup_{i' \in I'} G_{i'}\colon Q_\Sigma = \sum_{s\in S} r_s(n) * Q_s(n) \to \min;\\ &L^*(n) \leqslant L_{\text{доп}}(n),\\ &L_{\text{доп}}(n) = E_{\text{доп}}(n) - nl^V; \end{aligned} \tag{1}$$

здесь $G^* \in G$; $G$ — множество вариантов структуры, которые реализуют основные отношения, заданные на множестве элементов графа, и, возможно, дополнительные отношения, вытекающие из выполняемых над этими данными операций; $G_{i'}$ — базовая одноуровневая структура данных (см. п. 2); $I' = \{y_i * i\}$ — множество индексов структур, включенных в результирующую структуру; $i \in I$ — множество индексов структур данных — кандидатов на вхождение в комбинированную

структуру; $y_i$ — признак вхождения $i$-й структуры в комбинированную структуру; $r_s(n)$ — функция, определяющая число повторений операции типа $s \in S$ в зависимости от размера входа задачи, $r_s(n) \in R$; $Q_s(n)$ — оценка вычислительной сложности операции типа $s$ над результирующей структурой данных, $Q_s(n) \in Q^*$.

В этом случае задача определения оптимальной структуры данных заключается в нахождении такого множества индексов $I'$, при котором $Q_\Sigma \to \min$ при сохранении истинности отношения $L^*(n) \leqslant L_{\text{доп}}(n)$.

Таким образом, в соответствии с формальной постановкой основными этапами синтеза оптимальных структур данных являются: отображение описания схемы алгоритма в базис операций над структурами данных; определение числа выполнений каждой операции для каждой структуры данных (задание множества $R$); определение такого множества индексов $I'$, для которого значение приведенной целевой функции минимально, т.е. определение такого набора структур данных, объединение которых дает оптимальный результат с точки зрения указанного критерия; реализация объединения базовых структур данных для получения комбинированной структуры данных.

## СПИСОК ЛИТЕРАТУРЫ


1. А х о А., У л ь м а н Д., Х о п к р о ф т Д. Структуры данных и алгоритмы. – М.: Вильямс, 2003. – 384 с.
2. И в а н о в а Г. С. Математические модели структур // Информационные технологии. – 2006. – № 9. – С. 44–52.
3. О в ч и н н и к о в В. А. Алгоритмизация комбинаторно-оптимизационных задач при проектировании ЭВМ и систем. – М.: Изд-во МГТУ им. Н.Э. Баумана, 2001. – 288 с.
4. К о р м е н Т., Л е й з е р с о н Ч., Р и в е с т Р. Алгоритмы: построение и анализ. – М.: МЦНМО, 2004. – 960 с.


Статья поступила в редакцию 28.02.2008


Константин Алексеевич Пасечников родился в 1982 г., окончил МГТУ им. Н.Э. Баумана в 2005 г. Руководитель департамента технических проектов в ООО “Protection Technology” (StarForce). Автор 6 научных работ в области вычислительной техники. Специализируется в области автоматизации проектирования компьютерных систем.

K.A. Pasetchnikov (b. 1982) graduated from Bauman Moscow State Technical University in 2005. Head of Technical Projects Department at limited-liability company Protection Technology (StarForce). Author of 6 publications in field of computer science. Primary specialization is design of program systems.

Галина Сергеевна Иванова родилась в 1954 г., окончила МВТУ им. Н.Э. Баумана в 1978 г. Канд. техн. наук, доцент кафедры “Компьютерные системы и сети” МГТУ им. Н.Э. Баумана. Автор 50 научных работ в области вычислительной техники. Специализируется в области проектирования программных систем.

G.S. Ivanova (b. 1954) graduated from Bauman Moscow Higher Technical School in 1978, PhD (Eng), assoc. professor of “Computers Systems and Networks” Department of Bauman Moscow State Technical University. Author of 50 publications in field of computer science. Specializes in the field of designing program systems.