

ПРИНЦИПЫ ПОСТРОЕНИЯ БАЗ ДАННЫХ В ПРОИЗВОДСТВЕННЫХ ОБЪЕДИНЕНИЯХ

А. КВЕДЕРАВИЧЮС, Р. КЕВЛИЧЮС

Непрерывное увеличение объемов научной, технической и нормативной информации, интеграция и дифференциация научных исследований, разработка и производство новых сложных изделий требуют создания специализированных систем сбора, хранения и переработки информации, обеспечивающих представление любому потребителю точной, исчерпывающей информации в соответствии с его запросами, — используя сети ЭВМ, терминалы и технические средства специального назначения.

Одной из важных функций специализированных систем обработки информации, использующих сети ЭВМ, является организация и ведение информационного фонда, с одной стороны, и стратегии поиска — с другой (2). Для этого необходимо создать специализированные банки данных. Созданию, исследованию и эксплуатации банков данных посвящены многочисленные работы. Основной отличительной чертой созданных до сих пор банков данных является централизация ранее созданных массивов информации и обеспечение централизованного доступа ко всей базе данных (6; 5; 4). Однако в настоящее время имеет место важная тенденция создания распределенных баз данных как более надежных, удобных, децентрализованных и экономичных (1; 3). Основная идея распределенных баз данных заключается в том, что поисковая информация находится в разных узлах, в том числе отдаленных друг от друга на определенное расстояние. В распределенных базах данных различают расчлененные (каждый узел объединяет множество данных, не имеющих копий в других узлах), полностью дублированные (все узлы содержат одинаковые данные) и частично дублированные базы данных.

Анализ потоков документации и фактографической информации, подлежащей автоматическому поиску в проектно-производственных объединениях «Сигма», «Эльфа» и других, разнообразие форм представления информации, целей использования информации пользователями, экономико-технических параметров показал, что целесообразно создавать специфические системы баз данных, которые нельзя целиком отнести к вышеуказанным. Такие базы данных должны быть гибридными, т. е. комплексом централизованных, иерархических и распределенных баз данных. Условно назовем их иерархически-распределенными. Отличительной чертой таких баз данных является то, что данные в них распределены не только по узлам, отдаленным друг от друга, но и в конкретном узле в зависимости от потребительских групп или отдельных потребителей, частоты обращения, новизны, полезности, экономико-технических и других параметров.

Один из узлов в таких базах данных является центральным, в котором находятся данные, общие для всей системы. Это в основном данные, имеющие низкую частоту обращения и критические значения других параметров. Часть данных в узлах, кроме центрального, в зависимости от значения некоторых параметров может быть дублирована. Это

в основном данные, имеющие высокую частоту обращения. Аналогично данные в узле также могут быть распределены по потребителям, частоте обращения и другим параметрам. Дело в том, что большая часть потребителей специализированных систем интересуется сравнительно узкой областью информации, а время ответа не должно превышать 2—3 секунд, т. е. система должна работать в реальном масштабе времени, причем часто в диалоговом режиме. Именно распределение данных в узлах по некоторым параметрам сокращает время поиска.

Для обеспечения сокращения времени поиска целесообразно выделить два типа запросов: постоянные — для определенного круга потребителей (такие запросы могут иметь определенную форму и структуру с заранее подготовленными программами и указанными данными); полусвободные или свободные, сформированные на естественном языке (такие запросы необходимо дополнительно обрабатывать, т. е. придавать им необходимую форму и структуру). Формализованный запрос является исходной точкой для автоматической генерации программы, реализующей заданный запрос. Как правило, такие запросы не имеют заранее подготовленных программ и стратегии поиска необходимой информации. Они должны быть сформированы автоматически в диалоговом режиме.

Неотъемлемой частью иерархически-распределенных баз данных являются «управляющий» базой, ответственный за различные функции контроля, перераспределение данных между узлами и внутри узлов, обеспечение консультации, ведение учета параметров, оказание помощи пользователям, связание любого потребителя с любым узлом и т. д.

Обобщенная структура иерархически-распределенной базы данных такова (см. рис. 1):

V — центральный узел. В центральном узле находятся данные, критерий пользования которыми сравнительно низки. Этими данными может пользоваться любой потребитель любого узла. В случае увеличения критериев пользования в некоторых узлах соответствующие данные пересылаются в соответствующие узлы, а в случае уменьшения критериев пользования до критического уровня — определенные данные выводятся в архив. Вместе с тем в центральном узле находятся центральный «справочник» и система управления базой данных, выполняющие функции перераспределения данных между узлами, связывания потребителей одного узла с данными другого и т. п. (1; 6);

V_i — i -й узел. В нем находятся «справочник» узла и система управления базой данных этого узла. Сама база данных разбита на отдельные участки $V_{i1}, V_{i2}, \dots, V_{ij}, \dots, V_{im}$ (m — число участков в узле), имеющих постоянных потребителей. В узле V_i также находятся данные, общие для всех потребителей данного узла, в том числе и случайных. По необходимости данные в узле V_i и участке V_{im} могут быть разделены по частоте их использования или другим параметрам на группы. Система управления данными узла в зависимости от критериев пользования данными перераспределяет их между участками, внутри узла V_i между группами, пересылает в центральный узел или с разрешения последнего посылает в архив;

P_{ijk} — k -й потребитель j -го узла i -го участка. Потребитель может быть групповой или одиночный;

R_{ii} — i -й потребитель i -го узла, не имеющий определенного участка данных. Потребитель может быть групповой или одиночный;

B_i — плановый ввод данных, коррекция и вывод «мусора» i -го участка;

B — плановый ввод данных, коррекция и вывод «мусора» центрального участка;

$x_i - x_i(V - V_i)$ — двусторонняя связь между центральным узлом и узлом V_i ;

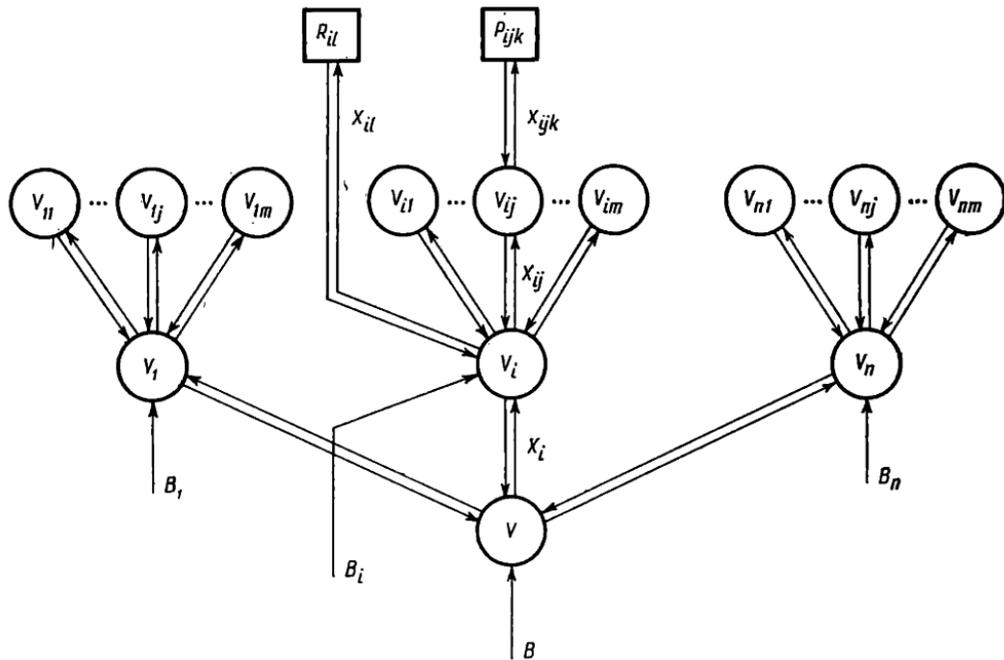


Рис. 1. Обобщенная структура иерархически-распределенного банка данных

$x_{ij} - x_{ij}(V_i - V_{ij})$ — двусторонняя связь между узлом V_i и участком V_{ij} этого узла;

$x_{ijh} - x_{ijh}(V_{ij} - P_{ijh})$ — двусторонняя связь между узлом V_i участка V_{ij} и потребителем P_{ijh} , имеющим собственный участок;

$x_{ii} - x_{ii}(V_i - R_{ii})$ — двусторонняя связь между узлом V_i и потребителем R_{ii} , не имеющим собственного участка данных.

Обобщенная стратегия поиска иерархически-распределенных баз данных показана на рис. 2.

Особое место в иерархически-распределенных базах данных занимает функция перераспределения данных. Она сводится к тому, что данные автоматически время от времени пересылаются из одного узла в другой, из одного участка в другой, из участка в узел и т. д.

При этом необходимо учитывать ряд экономических и технических параметров программных средств, параметры использования данных, таких как частота обращения, важность и др.

Следовательно, в определенный момент времени элементарная часть или группа данных будет характеризоваться вышеперечисленными параметрами, т. е.

$$C^i = \gamma(\gamma_1, \gamma_2, \dots, \gamma_j, \dots, \gamma_n), \quad (1)$$

где C^i — состояние данных класса i ;

$\gamma(\gamma_1, \gamma_2, \dots, \gamma_j, \dots, \gamma_n)$ — множество параметров, определяющих состояние;

γ_j — элемент множества, т. е. некоторый параметр, численное значение которого меняется в определенных пределах;

j — текущий номер параметра;

n — общее число параметров.

Если для каждого параметра γ_j в зависимости от его важности (веса) среди других параметров ввести пороговое значение Z_j , то каждый параметр можно представить в бинарном виде:

$$x_j = \begin{cases} 1, & \text{если } \gamma_j \geq Z_j, \\ 0, & \text{если } \gamma_j < Z_j. \end{cases} \quad (2)$$

Следовательно, данные класса i в каждый момент времени характеризуются совокупностью бинарных переменных x :

$$C^i = x(x_1, x_2, \dots, x_j, \dots, x_n); \quad (3)$$

Совокупность переменных x каким-то образом будет определять состояние данных класса i , которое зависит не только от того, какие значения принимают переменные в данный момент, но и от того, какими их значения были раньше. Из этого вытекает, что данные характеризуются не только переменной x , но и состоянием данных перед тем, как изменялась переменная x . Обозначим совокупность возможных состояний данных через переменную типа A .

$$A = a(a_1, a_2, \dots, a_i, \dots, a_m), \quad (4)$$

где A — совокупность возможных состояний данных класса;

a_i — элемент множества, т. е. некоторое определенное состояние;

i — текущий номер состояния;

m — общее число состояний.

Новое состояние данных будет зависеть от старого его состояния и от настоящего значения переменной x . Назовем эту зависимость состояний функцией переходов

$$a(t) = f(a(t-1), x(t)), \quad (5)$$

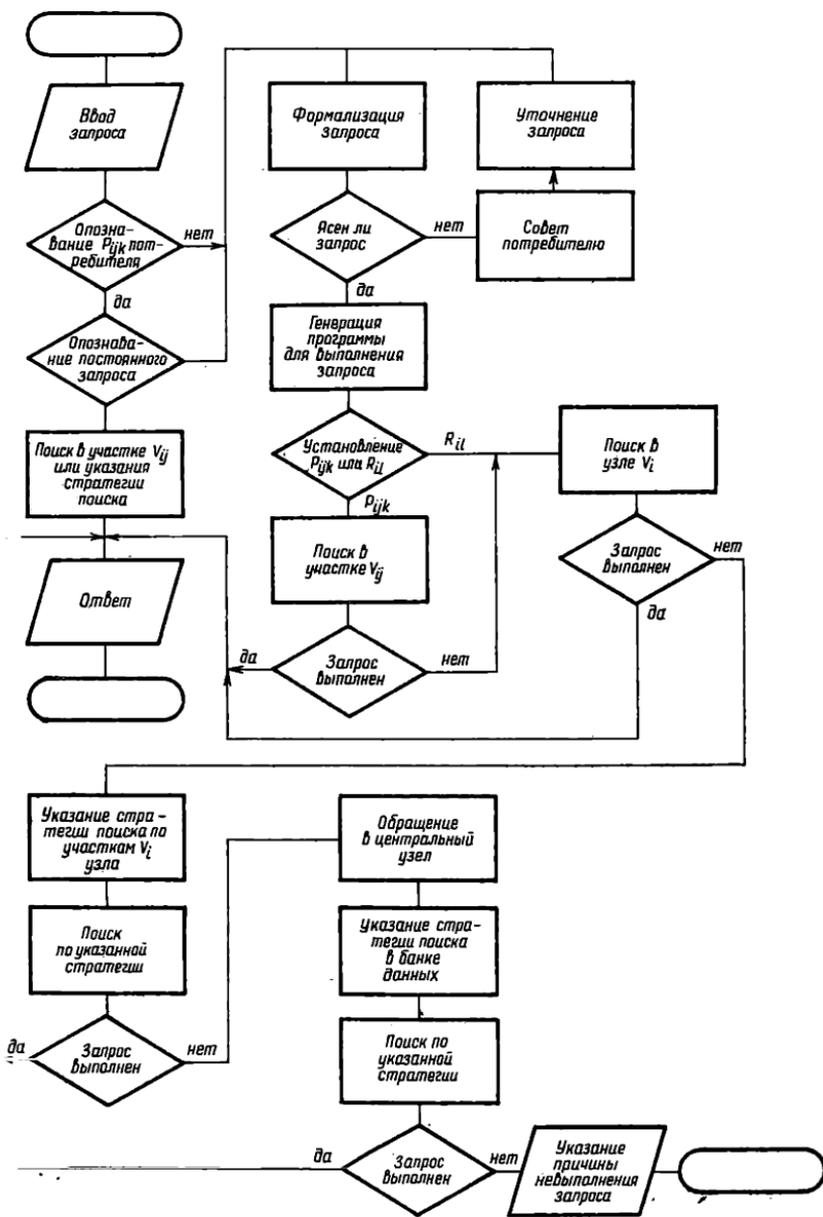


Рис. 2. Обобщенная стратегия поиска

связывающей состояние $a(t)$, в которое переходят данные, с прежним его состоянием $a(t-1)$ и переменной $x(t)$.

При изменении состояний данных система должна быть информирована совокупностью переменных типа y

$$y = y(y_1, y_2, \dots, y_k, \dots, y_p). \quad (6)$$

Новое значение переменной y будет зависеть от старого значения состояния данных и от настоящего значения переменной x . Назовем эту зависимость функцией выходов

$$y(t) = \lambda(a(t-1), x(t)), \quad (7)$$

связывающей выходную величину $y(t)$ с $a(t-1)$ и $x(t)$.

В результате такого обозначения мы получаем входной алфавит x^i данных класса i , состоящий из конечного количества входных букв; x_j^i — бинарные значения параметров данных; алфавит A^i , состоящий из конечного количества букв; a_j^i — обязательное место нахождения данных, переход в новое состояние совпадает с новым участком, в который должны пересылаться данные после изменения значения входного алфавита; алфавит y^i , состоящий из конечного количества букв; y_j^i — команды, определяющие автоматическое состояние модульной программы для физического перемещения данных из одного узла в другой, с одного участка в другой и функционирование конечного дискретного автомата, описанного системой уравнения

$$\begin{cases} a(t) = f(a(t-1), x(t)) \\ y(t) = \lambda(a(t-1), x(t)). \end{cases} \quad (8)$$

Следовательно, «управляющим» пересылания данных из одного узла V_i (участка) в другой V_j является конечный дискретный автомат.

Программная реализация такого автомата вполне возможна. Таким образом, любая диспетчерская программа в автоматизированных системах управления может быть интерпретирована как конечный дискретный автомат, функционирование которого можно задавать при помощи регулярных выражений, таблиц или ориентированного графа.

Вильнюсский университет
им. В. Капсукаса
Кафедра экономической
информации

Редколлегия вручено
в октябре 1982 г.

ЛИТЕРАТУРА

1. Глушков В. М., Сточный А. А., Базилевич И. А. Средства работы со структурированными банками в сетях ЭВМ: Обзор.— Управляющие системы и машины, 1980, № 6.
2. Кведеравичюс А.-К. В. Организация информационной базы ЛИПС в проектно-производственных объединениях.— В кн.: Проблемы и направления оптимизации и технизации учебного процесса. Вильнюс: ВГУ, 1979.
3. Сети ЭВМ / Под ред. В. М. Глушкова. М.: Связь, 1977.
4. Champagne G. A. Current trends in data base systems.— Computer, 1979, v. 12, No 5.
5. Goepel M. Zur Gestaltung einer Datenbank in betrieblichen Systemen der automatisierten Informationsverarbeitung.— In: Rectechnik Datenverarbeitung., 1971, t. 8, No 2.
6. Reference model of open systems architecture (version 3).— ISO/TC 97/SC, 1978.