Внутренняя модель данных

Базы данных

Виноградова М.В.
МГТУ им. Н.Э. Баумана, ИУ5

Хранение базы данных

- Логическая структура данных:
  - Таблица (двумерный или многомерный массив данных)
  - Древовидные иерархические структуры
  - Сетевые структуры
- Структура в оперативной памяти:
  - Адресация по месту (логический или физический указатель)
  - Адресация по содержимому (по ключу)

Адресная функция:

- отображает логическую структуру данных на физическую структуру хранения

Таблица как список записей

- Списковая структура:

$$X = (X[1], X[2], X[3], \ldots , X[n])$$

- X [i] – i-ый элемент линейного списка (запись/кортеж таблицы)

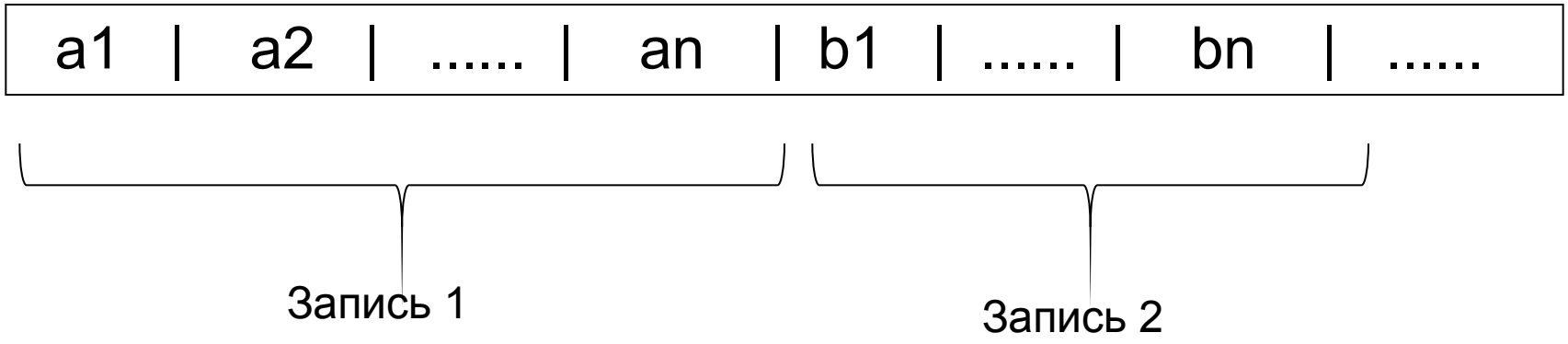
- Типы записей:
  - Фиксированной длины
  - Переменной длины
  - Неопределенной длины

Записи фиксированной длины

- S (структура) записи = const
- M (размер) записи = const
- Mi поля = const

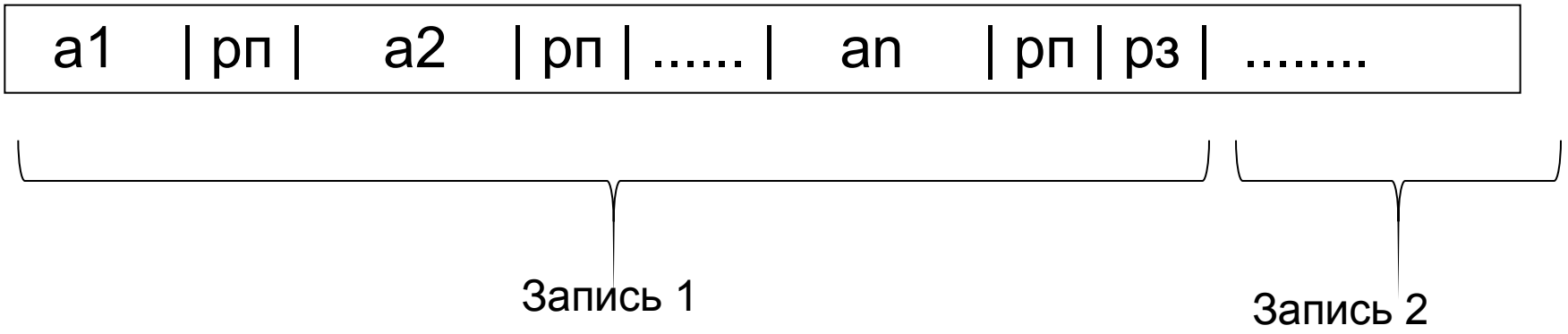
(+) хорошая скорость обработки

(-) большой расход памяти



Записи переменной длины

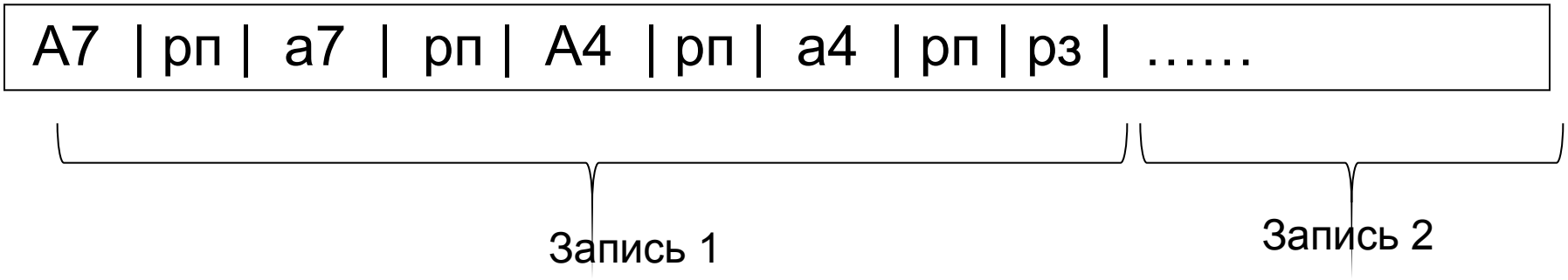
- S (структура) записи = const
- M (размер) записи = var
- Mi поля = var
- Разделитель полей (рп)
- Разделитель записей (рз)



Записи неопределенной длины

- S (структура) записи = var
- M (размер) записи = var
- Mi поля = var
- Разделитель полей (рп)
- Разделитель записей (рз)

Ai (имя) | ai (значение)



A1	A2	A3	A4	A5	A6	A7
NULL	NULL	NULL	a4	NULL	NULL	a7

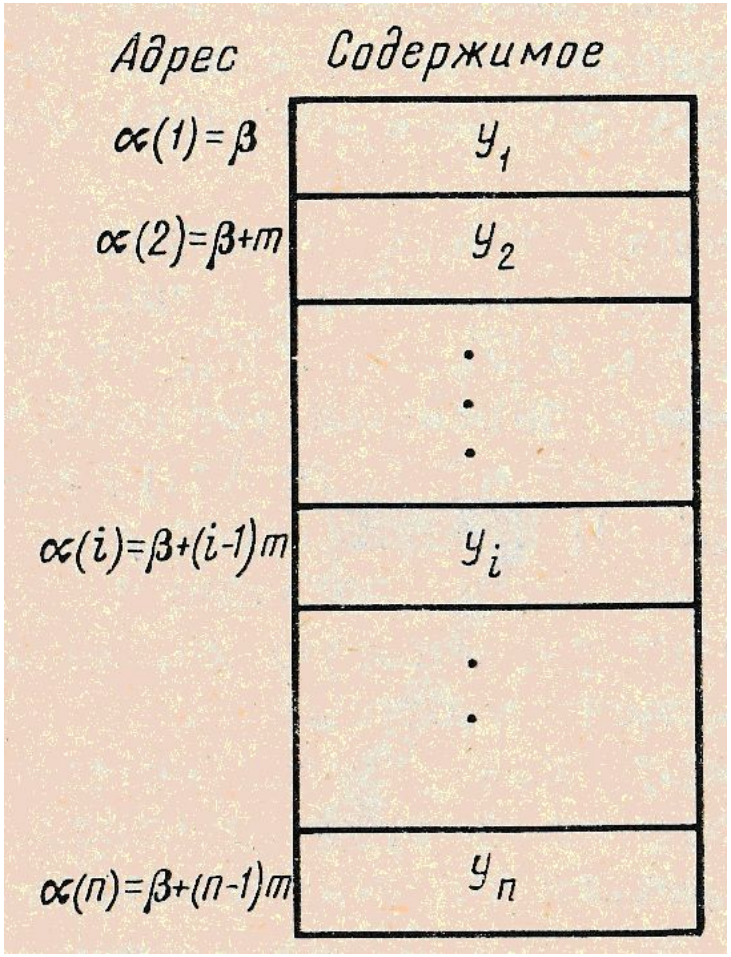
Хранение списковых структур в оперативной памяти

Оперативная память как последовательность байт:

a	v	f	g	k	l	h	k	c	d	h	m	r	s	…….

- Реализация адресной функции:
  - Последовательное распределение памяти
  - Связанное распределение памяти

Последовательное распределение памяти



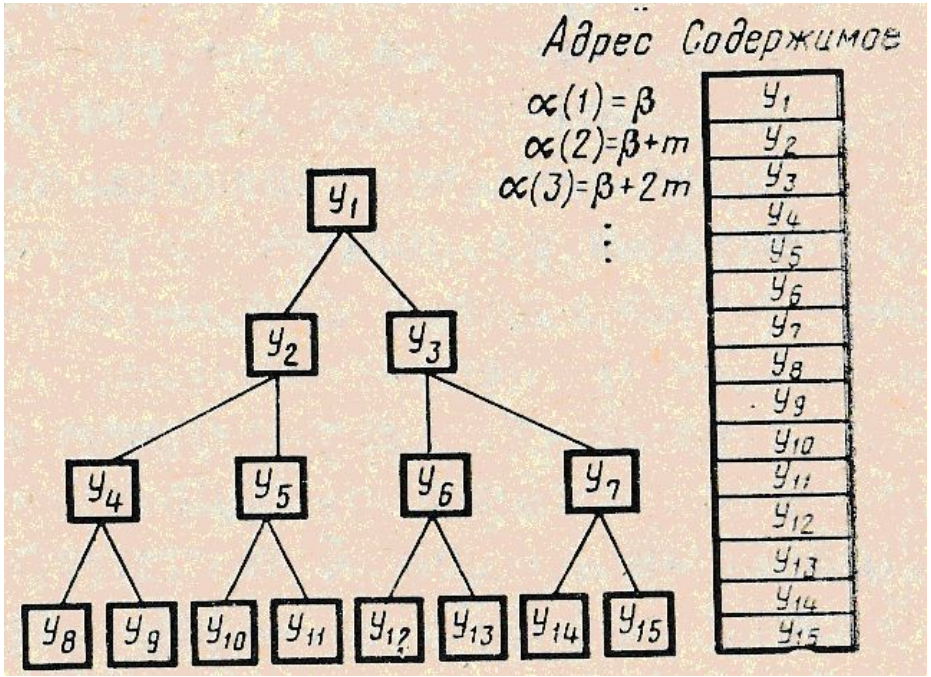
- Последовательный список:
  - Yi – элемент списка
  - N – количество элементов массива
  - m – размер элемента списка
  - B – адрес начала списка (база)

Адресная функция:

a(i) = B + (i-1)*m

Позволяет хранить древовидные структуры

Последовательный список для регулярного двоичного дерева



Адресная функция

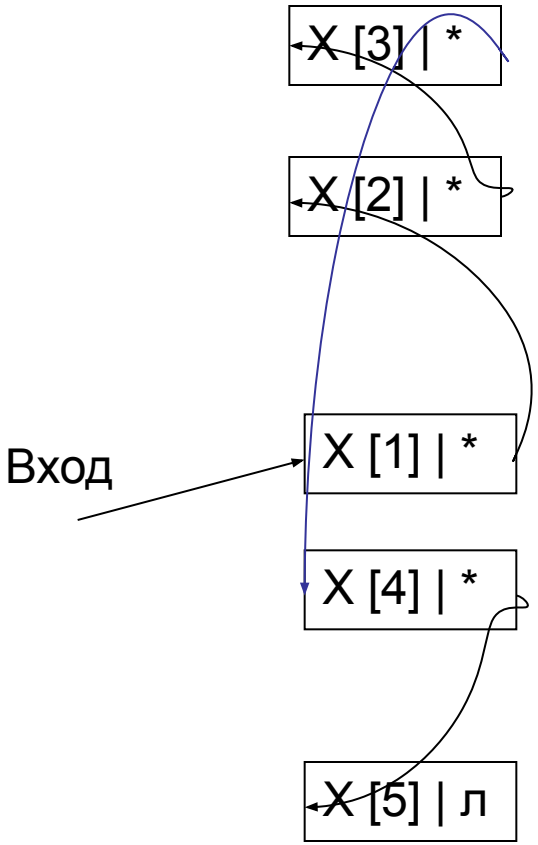
a(k) = B+(k-1)*m

Связанное распределение памяти

- Связанный список
- Цепная структура/Цепь

Особенности:

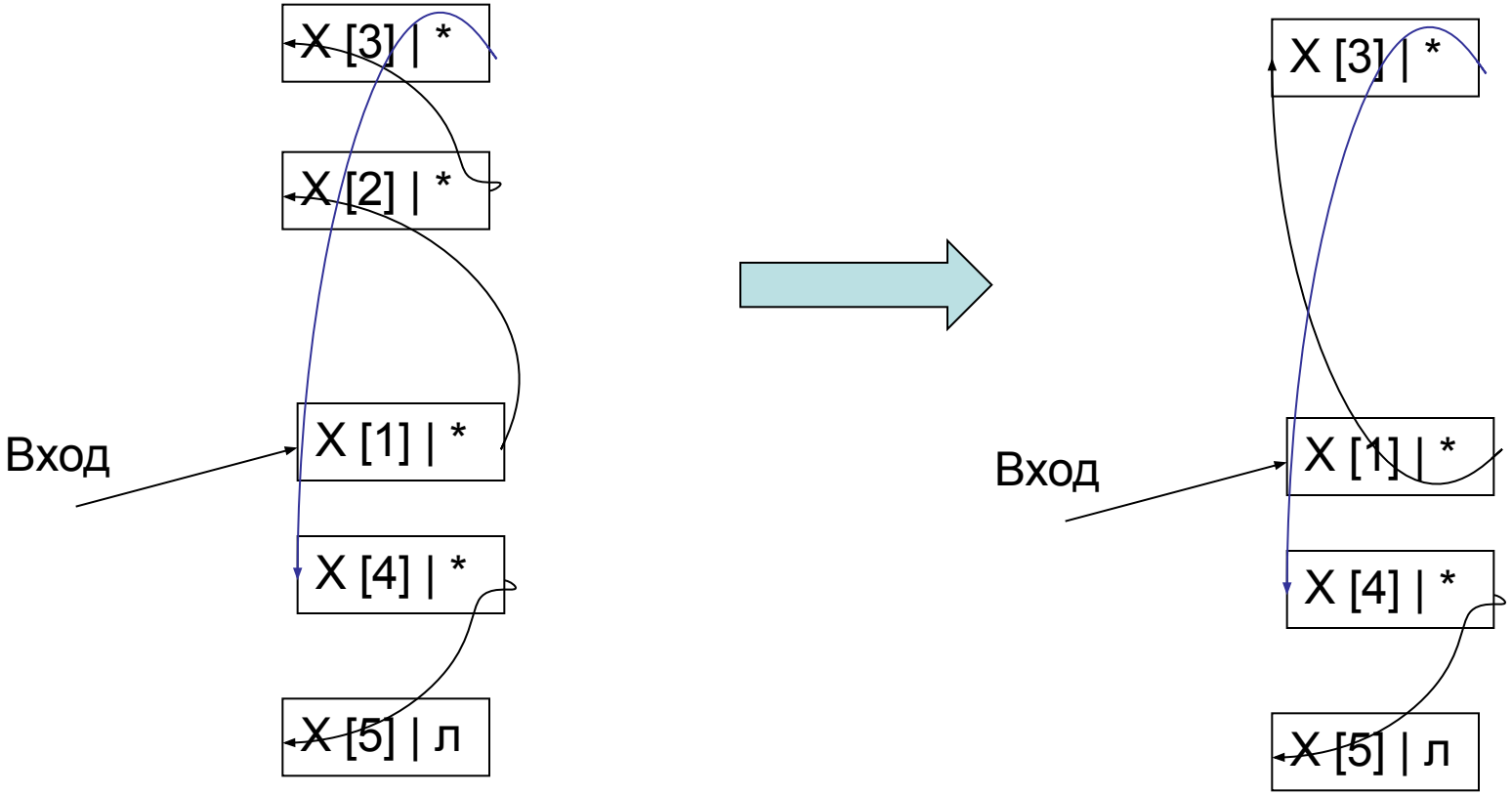
- Элементы хранят в любом месте ОП
- Последовательность элементов задают указатели
- В инвентарных таблицах БД хранят указатель на начало списка (голову)
- Л – конец списка (спец. символ)



Использование связаных списков

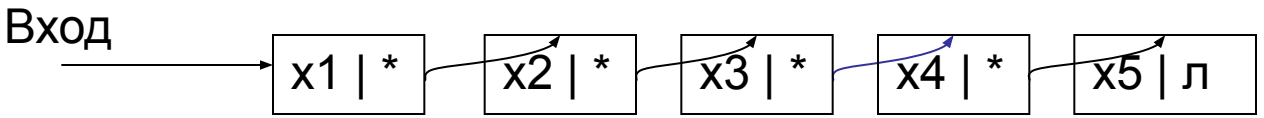
(+) гибкость, изменение структуры без переноса данных

(-) больший объем хранения

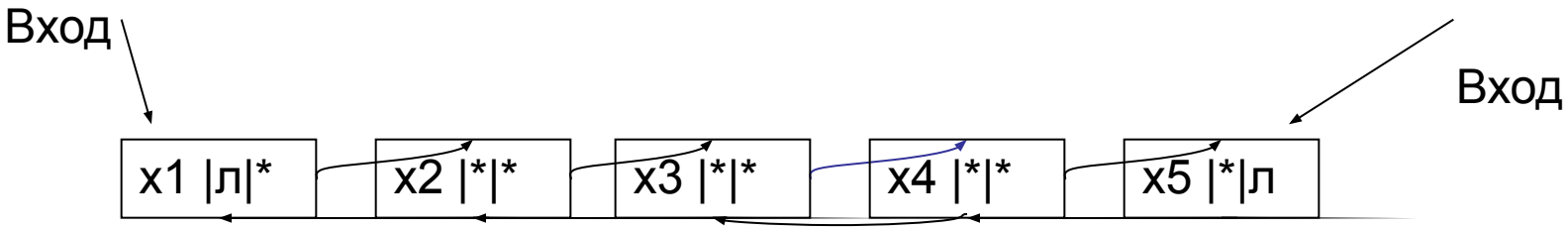


Виды списков

- Список (цепь/цепная структура) с одним указателем



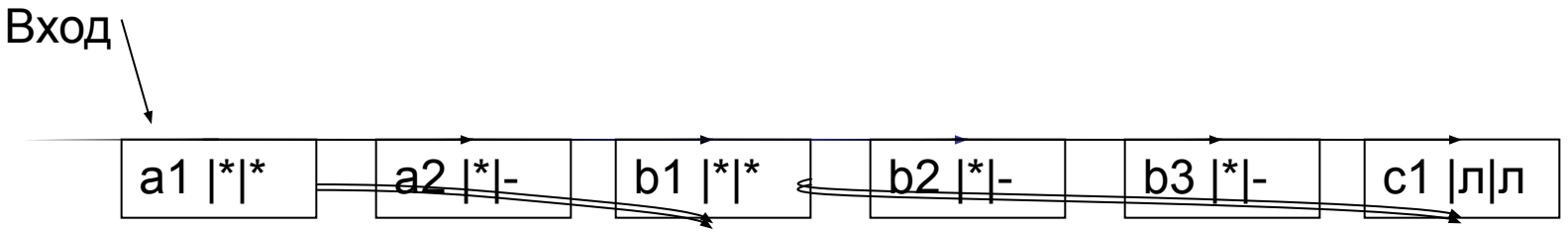
- Цепь с двумя указателями (Список с обратным проходом)
  - Хранят указатель на голову и на хвост
  - Обход в прямом направлении и обход в обратном направлении



Список с пропусками

- Цепь с пропусками (Мультицепь)
  - Указатель на группу в прямом или обратном направлении
  - Оптимальный размер группы r(n) при количестве элементов n

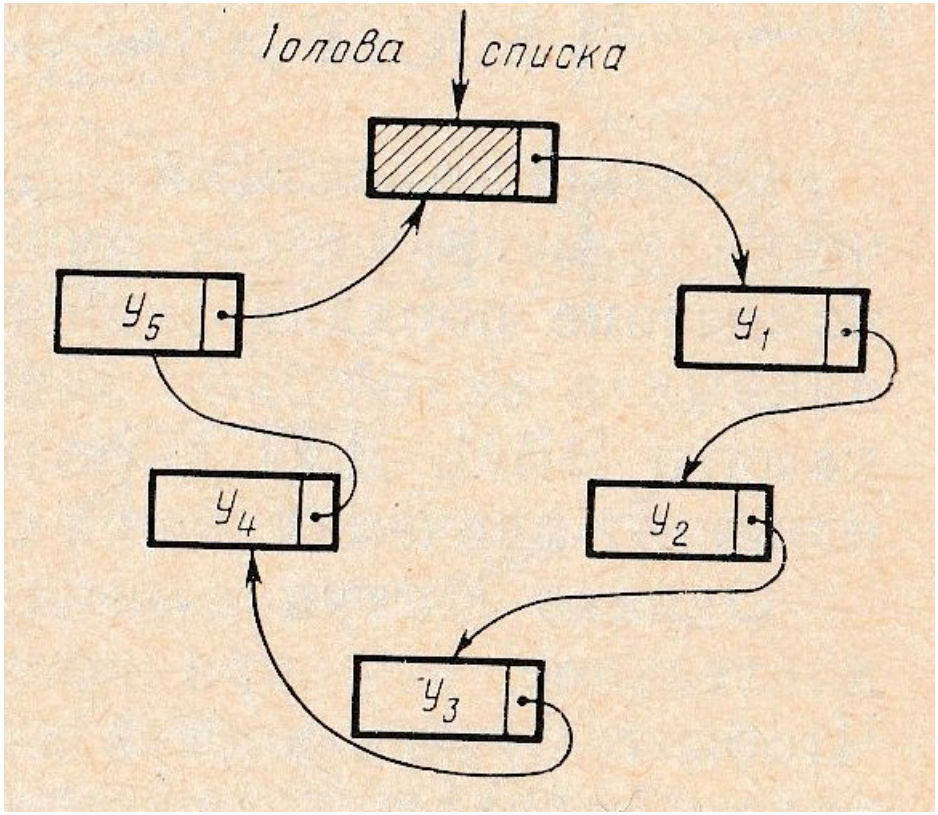
$$r(n) = \sqrt{n}$$



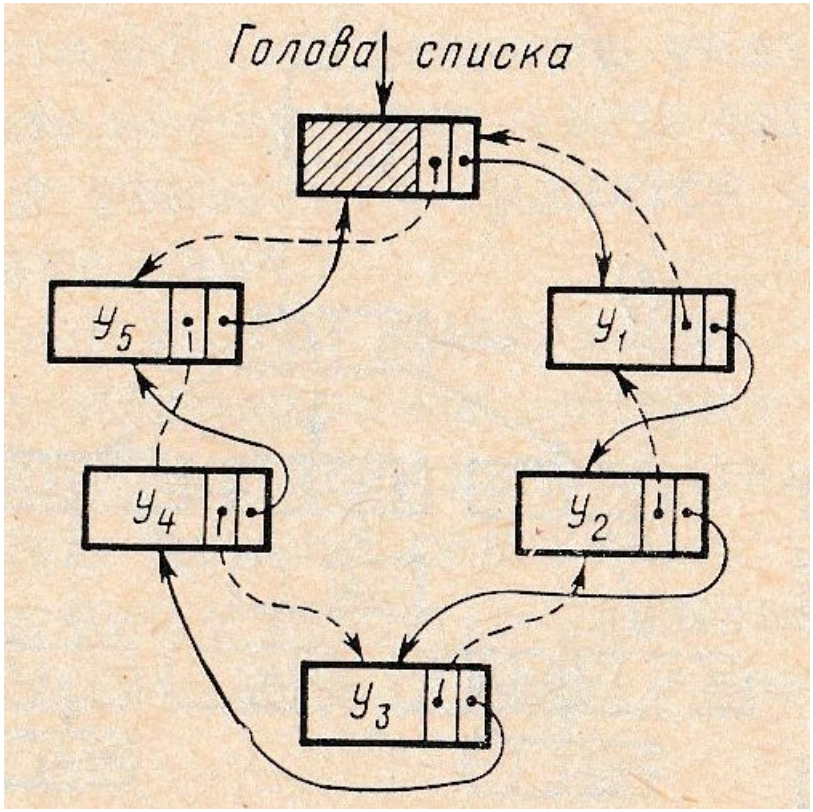
Кольцевые списки (кольца)

Однонаправленный циклический список

(+) Каждый элемент достижим из каждого

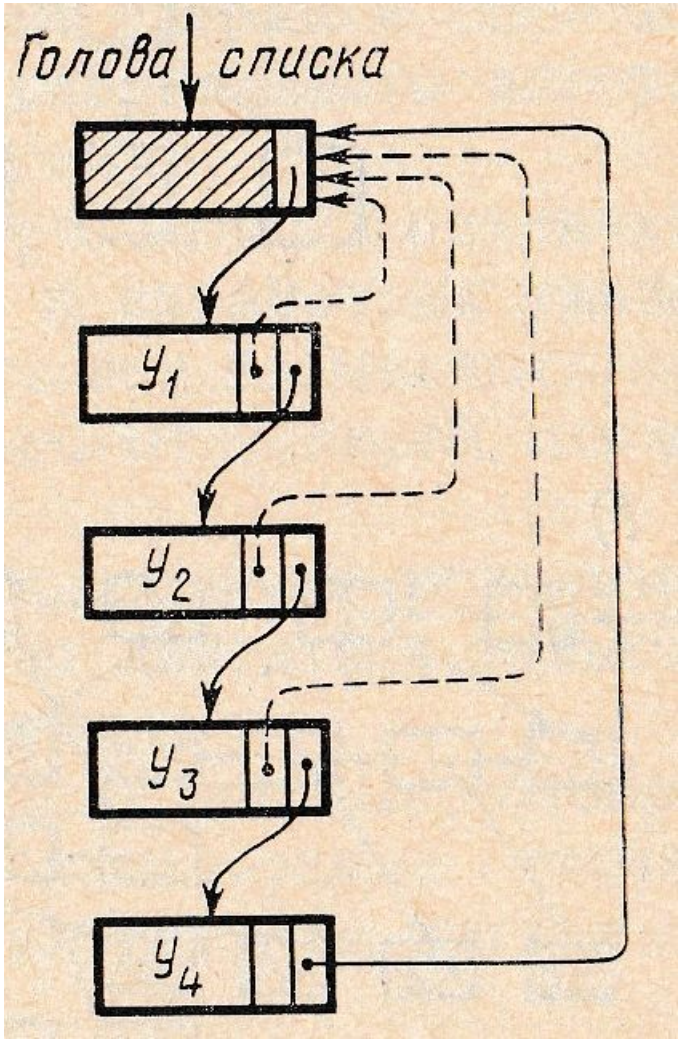


Двунаправленный циклический список

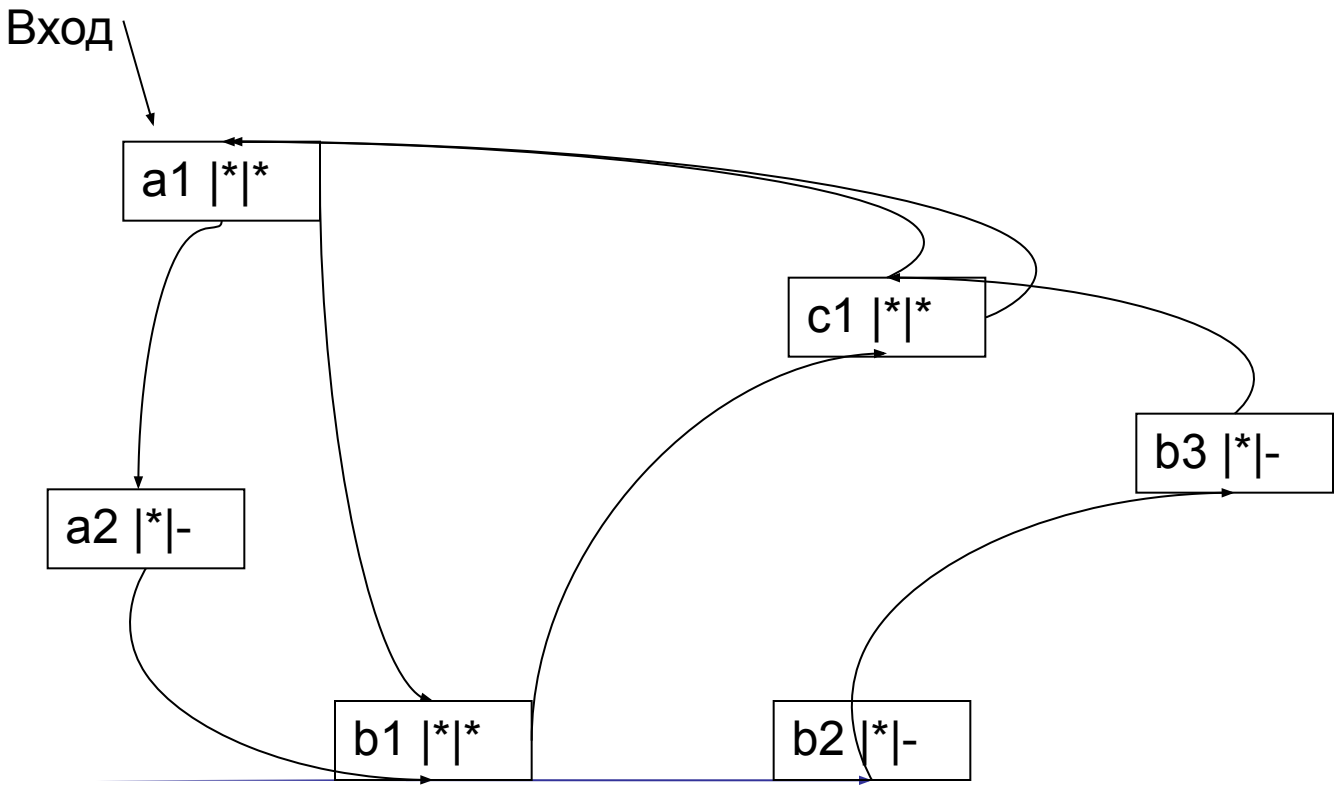


Коралловое кольцо

- Каждый элемент имеет указатель на голову списка



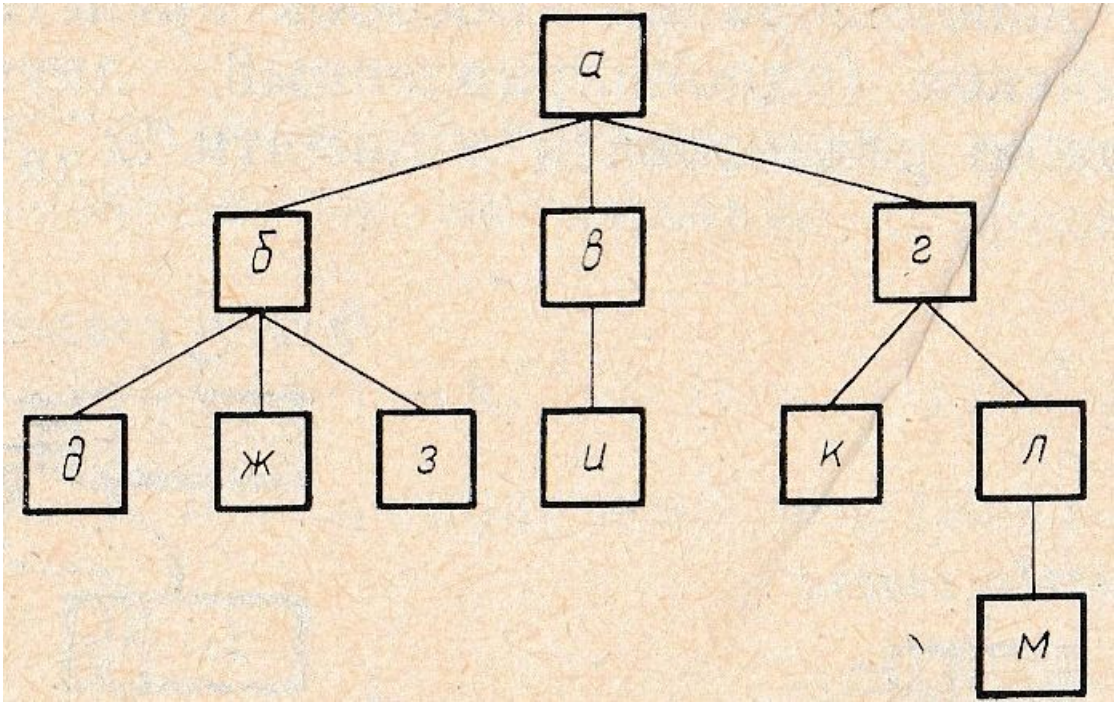
Кольца с пропусками



Многосвязанные списки

- Виды линейных списков
  - Однонаправленные
  - Двунаправленные
  - циклические

Используются для организации древовидных и сетевых структур



Типы указателей

- Абсолютный
  - физический адрес памяти

  (+) скорость

  (-) жесткая привязка
- Относительный
  - базовый адрес
  - расстояние между элементами списка
- Символический
  - адрес вычисляется по значению ключа
  - если совпадает, то цепочка

  (+) независимость изменения элемента

  (-) низкая скорость

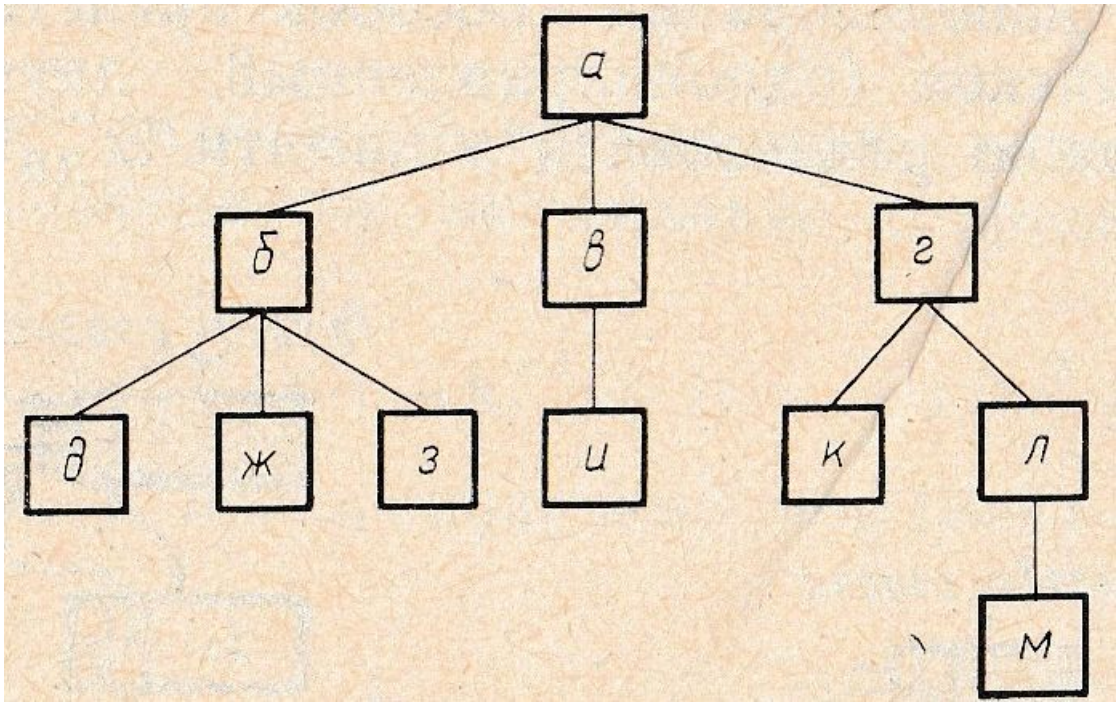
Виды указателей

- По организации структуры данных:
  - Встроенные (часть записи)
  - Справочник (хранятся отдельно)

- Назначение:
  - Направление доступа
  - Цепочки связных данных
  - Семантические связи данных

Методы представления древовидных структур

- Допустим, что данные узла – это записи фиксированной длины



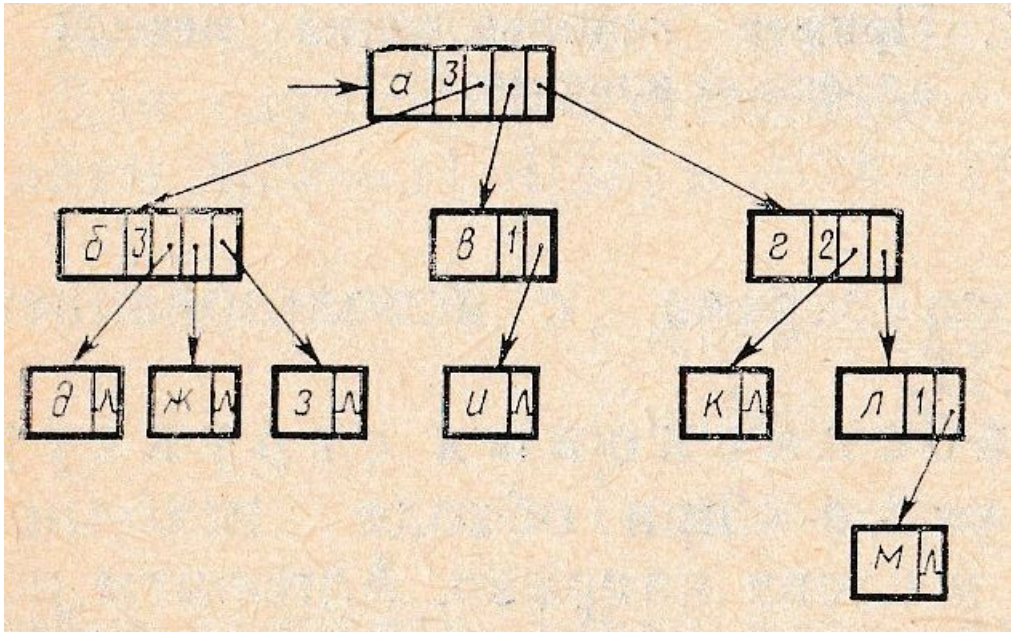
1-й уровень

2-й уровень

3-й уровень

4-й уровень

Метод указателей на порожденные записи



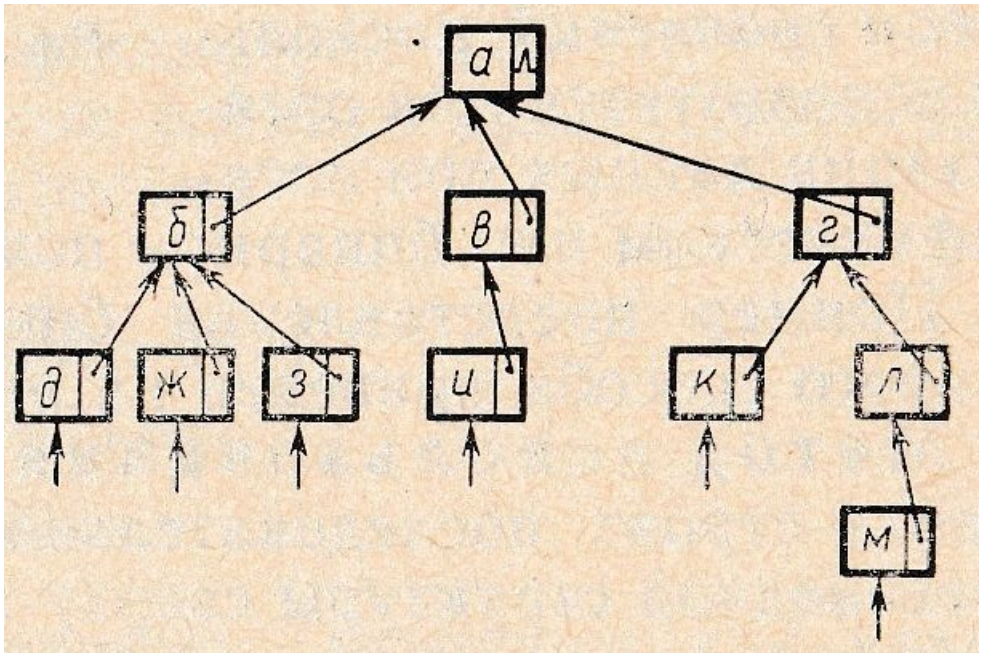
(+) обход БД в прямом направлении

(-) Переменное количество указателей

Метод указателей на исходные записи

(+) минимум указателей

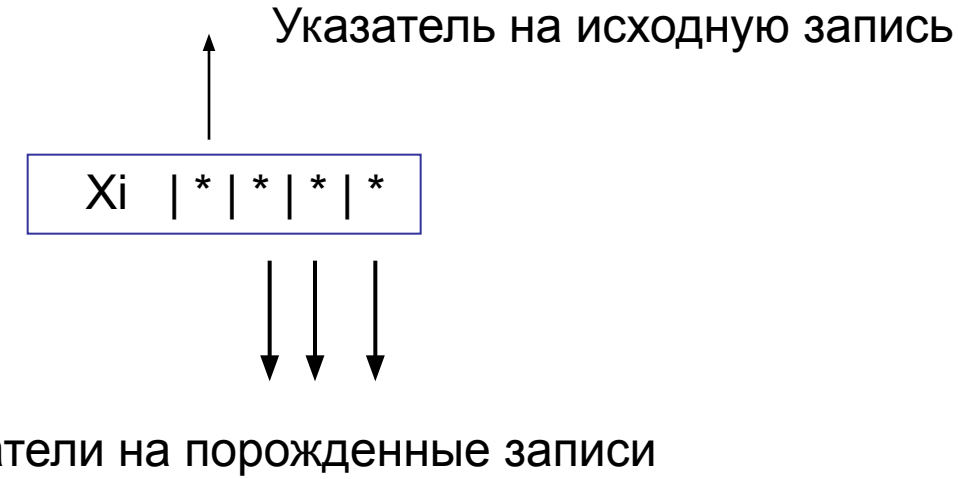
(-) много точек входа



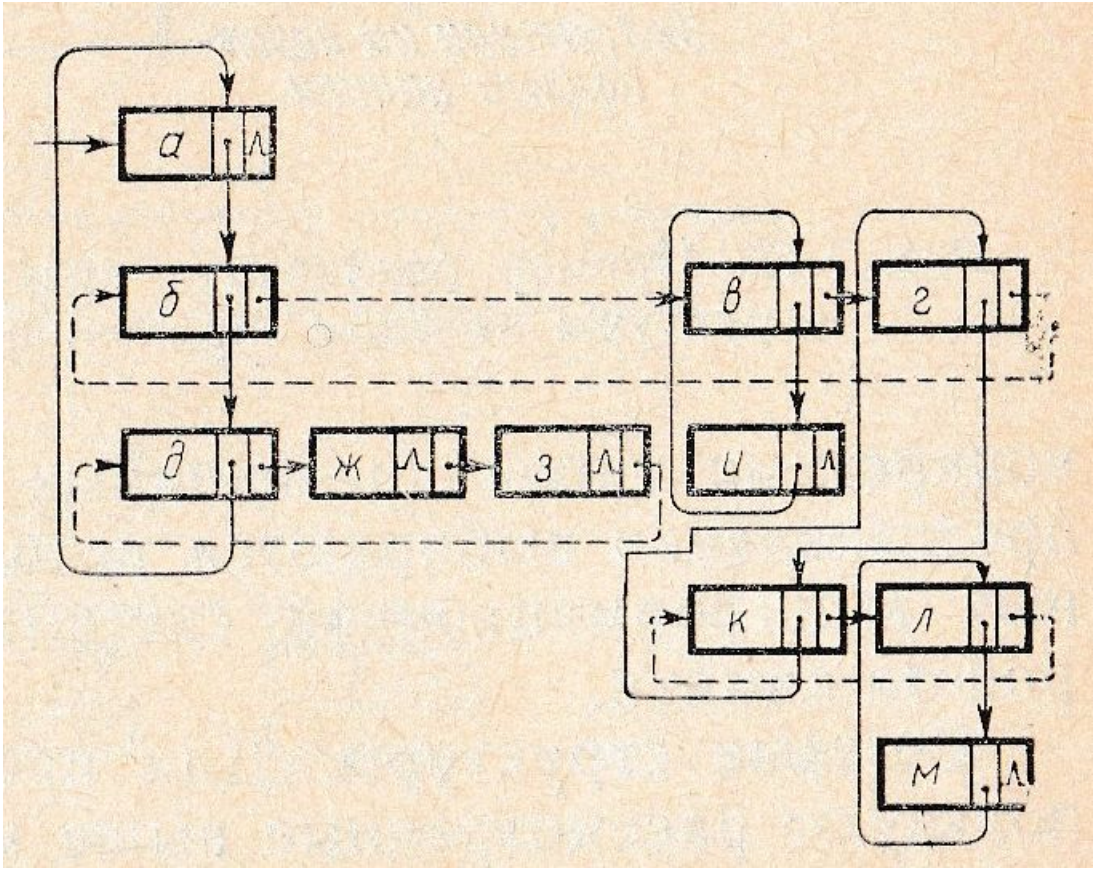
Метод указателей на порожденные и исходные записи

- Сочетает указатели на исходные и указатели на порожденные
- На первом месте ставят указатель на исходную запись
- Храниться только указатель на корень дерева

(+) большая надежность
(+) прямой и обратный обход



Метод указателей на порожденные и подобные записи

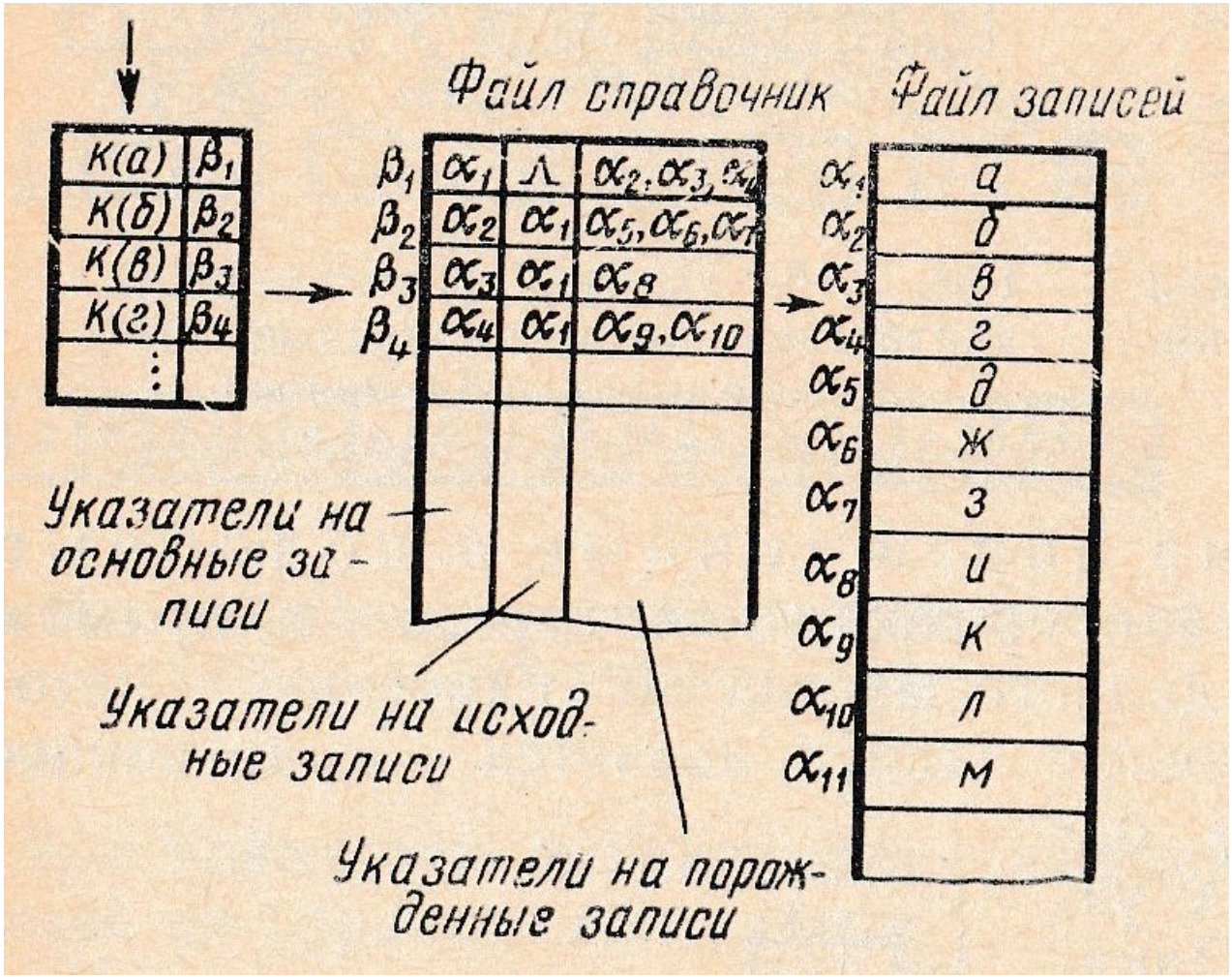


- с кольцевыми структурами

(+) В прямом и обратном направлении

(+) Постоянное количество указателей (2)

Метод справочника

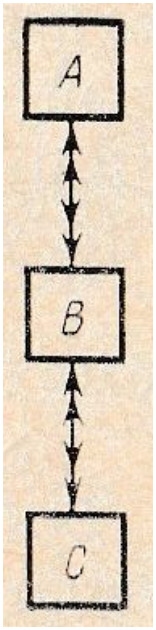


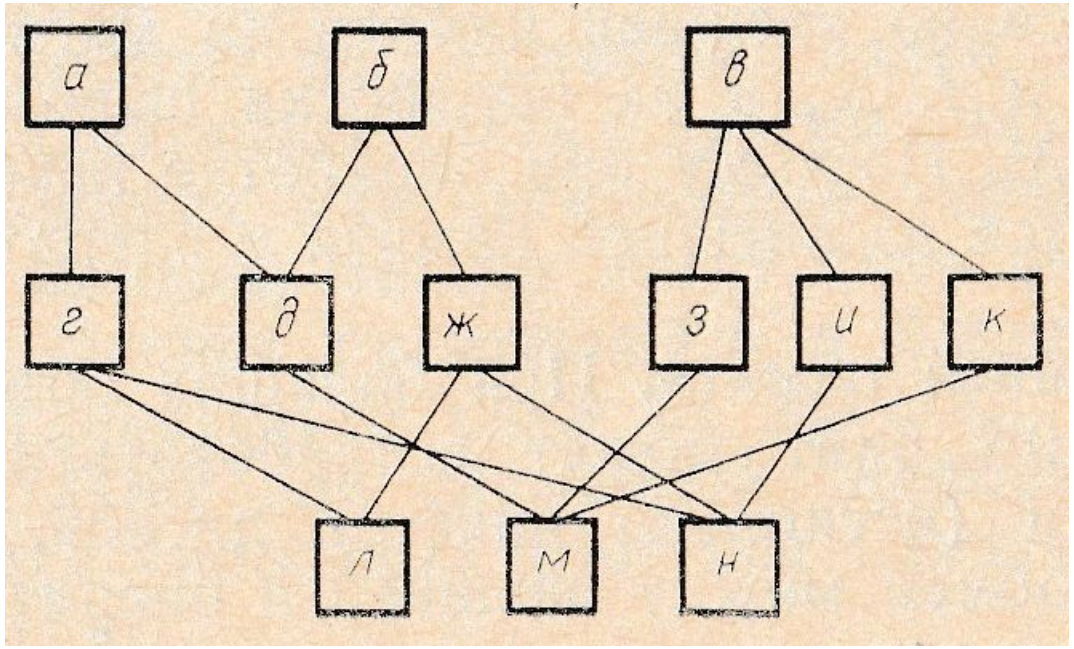
- Файл данных (записей)
- Файл справочника
- Файл индекса

Методы организации сетевых структур

- Метод указателей на порожденные и исходные записи
  - используют разделитель между указателями на порожденные и исходные записи или счетчик
- Метод типа справочник

Пример сетевой структуры





Реализация сетевой структуры