Севастопольский государственный университет
Кафедра «Информационные системы»

Курс лекций по дисциплине "МЕТОДЫ и СИСТЕМЫ ИСКУССТВЕННОГО ИНТЕЛЛЕКТА" (СИИ)

Лектор: Бондарев Владимир Николаевич

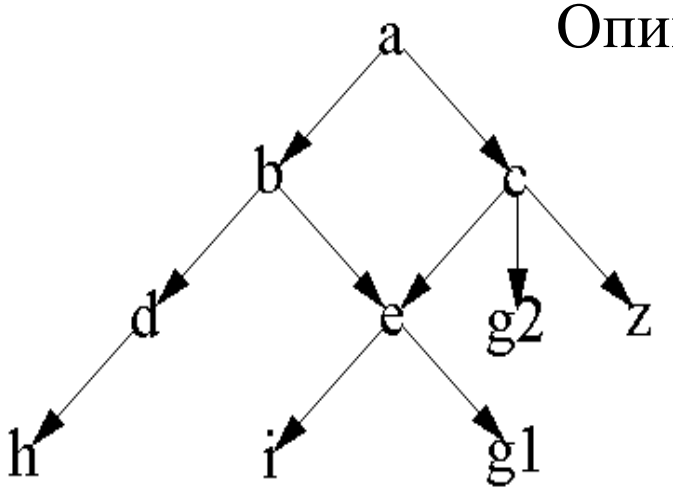
Лекция 16

Поиск решений в пространстве состояний на языке Пролог.

В.Бондарев

Поиск в глубину

Рассмотрим граф состояний, изображенный на рисунке.



Опишем его с помощью фактов: **после(X, Y).**

после(a, b).
после(a, c).
после(b, d).
после(b, e).
…
после(e, g1).

Чтобы выяснить, имеется ли путь из стартовой вершины **X** в целевую вершину **У**, определим предикат **путь(X,У):**

путь(X, X). **%выход из рекурсии, путь(g1,g1)**
путь(X, Y): – после(X, Z),путь(Z, Y).
? – путь(a, g1).

Данный вариант программы, возвращает ответ “да” или “нет”.
Чтобы запомнить решающий путь введем в рассмотрение список закрытых вершин **ЗКР.**

Поиск в глубину

Определим предикат, выполняющий поиск в глубину:

в_глубину(Старт, Цель, Реш_путь): –
путь(Старт, Цель, [Старт], ОбрПуть),
реверс(ОбрПуть, Реш_путь).

Здесь **предикат в_глубину** переопределяется через предикат **путь**. Третьим аргументом предиката **путь** является список **ЗКР**.

путь(X, X, ЗКР, ОбрПуть): – ОбрПуть=ЗКР.
путь(X, Y, ЗКР, ОбрПуть): – после(X, Z),
путь(Z, Y, [Z|ЗКР], ОбрПуть).

При каждом рекурсивном вызове очередная вершина **Z**, следующая после **X**, вносится в начало списка **ЗКР**. Если вершина **Z** совпадает с целевой вершиной **Y**, то рекурсивные вызовы прерываются и переменная **ОбрПуть** получает значение списка **ЗКР**.

Поиск в глубину

Если вершина **Z** является **тупиковой**, то цель **после(Z, _)** терпит неудачу, и пролог-система выполняет возврат к предыдущей вершине, в которой имеется альтернативный путь для продолжения поиска. В ходе возврата **все подстановки**, которые были сделаны в список **ЗКР** на тупиковом направлении, **стираются.**

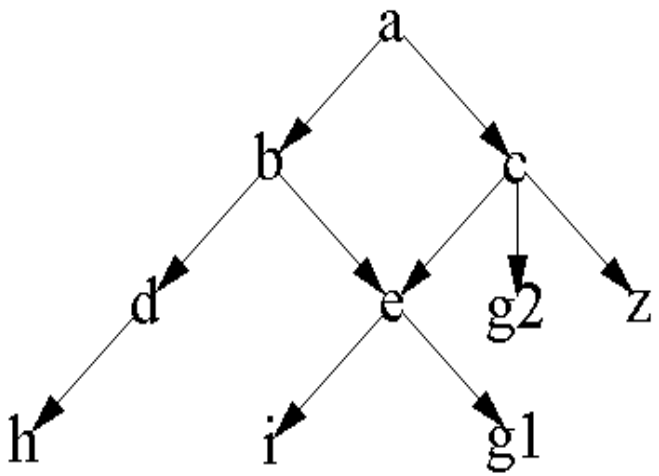
Проиллюстрируем это, внеся изменения в предикат **путь**:

путь(X, Y, ЗКР, Путь): – write(ЗКР), nl, после(X, Z), …

Тогда при запросе

? – в_глубину(a, g1, P), write(P).

[a]
[b, a]
[d, b, a]
[h, d, b, a]
[e, b, a]
[i, e, b, a]
P=[a, b, e, g1]



Поиск в глубину

Если на графе состояний **имеются циклы**, то программа может зациклиться. Чтобы устранить этот недостаток, необходимо добавлять очередную вершину **Z** в список **ЗКР** только в том случае, если ее там нет:

```
путь(X, Y, ЗКР, ОбрПуть): – после(X, Z), not элемент(Z, ЗКР),
                              путь(Z, Y, [Z|ЗКР], ОбрПуть).
```

Поиск с ограничением глубины:

```
путь2(X, X, ЗКР, ОбрПуть, _ ):-ОбрПуть=ЗКР.
путь2(X, Y , ЗКР, ОбрПуть, МаксГлуб):-
    МаксГлуб > 0,
    после(X,Z),
    Макс1 is МаксГлуб-1,
    путь2(Z, Y, [Z|ЗКР], ОбрПуть, Макс1).
```

Поиск в ширину

Реализуем **стандартный алгоритм поиска в ширину**. Введем в рассмотрение предикат **в_ширину(ОТК,ЗКР,Решение)**, где **Решение** – список обследованных вершин, соответствующий списку **ЗКР** в момент достижения целевой вершины.

Указанные списки состоят из **подсписков,** в каждом из которых два элемента. Первый элемент – это имя некоторой вершины графа состояний, второй – имя ее родительской вершины.

При определении предиката **в_ширину(ОТК,ЗКР,Решение)** для раскрытия очередной вершины и генерации дочерних вершин будем использовать встроенный предикат **bagof**. Цель

bagof(X, C, L)

формирует список **L** всех объектов **X**, удовлетворяющих условию **C**.

Поиск в ширину

Например,

после(a, b).
после(a, c).
после(b, d).
после(d, h).
после(a, e).
после(e, k).
? – bagof([Z | a], после(a, Z), L).
L = [[b | a], [c | a], [e | a]]

В этом случае в списке **L** накапливаются подсписки, содержащие дочерние вершины для вершины ‘**a**’.

Используя рекурсию, определим предикат **в_ширину( )**:

Поиск в ширину

в_ширину([[О1|О2]|Ост_ОТК],ЗКР,Решение):-
цель(О1),!,Решение=[[О1|О2]|ЗКР].

В начале поиска список **ЗКР** пустой, а список **ОТК** содержит подсписок, включающий начальную вершину. На каждом шаге рекурсии выделяется первый подсписок **[О1|О2]** списка **ОТК**. Выясняется, является ли вершина **О1** целевой (**цель(О1)**). Если **О1** целевая вершина, то подсписок **[О1|О2]** добавляется в начало списка **ЗКР**, и рекурсия прерывается.

Поиск в ширину

```
в_ширину([[O1|O2]|Ост_ОТК],ЗКР,Решение):-
    bagof([Z|O1],(после(O1,Z), not(элемент1(Z,ЗКР)),
                                  not(элемент1(Z,Ост_ОТК))),L),
    слияние(Ост_ОТК,L,Нов_ОТК),!,
    в_ширину(Нов_ОТК,[[O1|O2]|ЗКР],Решение);
    в_ширину(Ост_ОТК,ЗКР,Решение).
```

Если **O1** не целевая вершина, то находятся все продолжения пути после вершины **O1**. Для этого применяется предикат **bagof**, который обеспечивает поиск всех отрезков графа состояний **[Z|O1]** таких, что вершина **Z** не входит в список **ЗКР** и в остаток списка **ОТК (Ост_ОТК)**. Все найденные дочерние вершины накапливаются в списке **L** в виде подсписков и помещаются в конец списка **Ост_ОТК**, образуя список **Нов_ОТК**. Затем обработанный первый подсписок **[O1|O2]** списка **ОТК** перемещается в начало списка **ЗКР**, и предикат **в_ширину** вызывается при новых значениях аргументов.

Поиск в ширину

Если вершина **O1** является тупиковой, то предикат **после(O1,Z)** завершается неудачей. В этом случае никакие новые вершины не добавляются в конец списка **ОТК**, и обеспечивается альтернативный вызов предиката **в_ширину**:

в_ширину(Ост_ОТК, ЗКР, Решение).

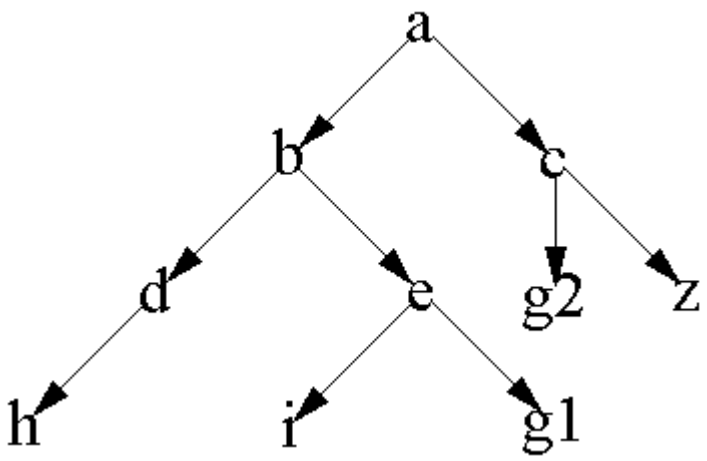
Пример вызова:

?-в_ширину([[a|a]], [ ],Решение), *nl*,
** *write*('Решение='), *write*(Решение), *nl*.**
** Решение=[[g1|e], [e|b], [d|b], [c|a], [b|a], [a|a]]**

Список **Решение** формируется таким образом, что его первый подсписок в качестве первого элемента содержит целевую вершину. Просмотрев список **Решение** от целевой вершины до стартовой вершины, можно восстановить решающий путь.

Реш_путь=[[g1|e],[e|b],[b|a],[a|a]]

Поиск в ширину



Будем хранить в списке **ОТК** множество путей-кандидатов. Поиск начинается с одноэлементного множества путей-кандидатов: **[[Старт]]**

В случае графа, изображенного на рисунке список **ОТК** будет содержать следующие пути-кандидаты:

1. [[a]]
2. [[b,a], [c,a]]
3. [[c,a], [d,b,a], [e,b,a]]
4. [[d,b,a], [e,b,a], [g2,c,a],[z,c,a]]
…
7. [[g2,c,a],[z,c,a], [h,d,b,a], [i,e,b,a], [g1,e,b,a]]
Решение : [g2,c,a]

Поиск в ширину

Правила поиска в ширину:

1) если голова первого пути – это целевая вершина, то взять этот путь в качестве решения, иначе

2) удалить первый путь-кандидат из списка **ОТК** и **породить множество всех возможных продолжений этого пути** на один шаг; добавить это множество продолжений в конец **ОТК** и выполнить поиск в ширину с полученным новым множеством