Введение в ИИ на примере языка Prolog

Списки

https://github.com/Inscriptor/IntroductionToAI/tree/master/pdf

Федеральное государственное автономное образовательное учреждение высшего образования
«Новосибирский национальный исследовательский государственный университет»

2 октября 2019 г.

Понятие и вид списка в Prolog

Списки

Список в Prolog

- Списком называют конечную последовательность элементов.
- В синтаксисе языка Prolog границы списка определяются квадратными скобками [и].
- Элементы списка отделяются друг от друга запятой. Элементами списка могут быть любые термы.
- Любой непустой список можно разделить на голову и хвост (head и tail). Для записи используется специальный символ |. `[Head | Tail] = [one, two, three] => Head = one, Tail = [two, three]`. Голова списка — это элемент, хвост списка — это список.
- Пустой список не может быть разделен на голову и хвост.

Списки

Простой пример

```
[]
[first, second, third, fourth, fifth]
[1, 2, 3, 4, 5, 6, 7, 8, 9, 0]
[first, 2, color(cornie, black), F, fifth, F]
[first, second, [third, fourth], [fifth, color(cornie, black)]]
[[], [], car(volkswagen), F, 1, 2, [1, F, car(bmw), [1, 2, 4]], X]
```

Списки

Получение элементов списка

```
[abyssian, bobtail, [bengal, birman]]
```

Списки

Получение элементов списка

```
[abyssian, bobtail, [bengal, birman]]
```

- `[H|T] = [abyssian, bobtail, [bengal, birman]]`.

Списки

Получение элементов списка

```
[abyssian, bobtail, [bengal, birman]]
```

- `[H|T] = [abyssian, bobtail, [bengal, birman]]`.
- `[F,S|T] = [abyssian, bobtail, [bengal, birman]]`.

Списки

Получение элементов списка

```
[abyssian, bobtail, [bengal, birman]]
```

- `[H|T] = [abyssian, bobtail, [bengal, birman]].`
- `[F,S|T] = [abyssian, bobtail, [bengal, birman]].`
- `[_,S|T] = [abyssian, bobtail, [bengal, birman]].`

Списки

Получение элементов списка

```
[abyssian, bobtail, [bengal, birman]]
```

- `[H|T] = [abyssian, bobtail, [bengal, birman]].`
- `[F,S|T] = [abyssian, bobtail, [bengal, birman]].`
- `[_,S|T] = [abyssian, bobtail, [bengal, birman]].`
- `[First,_,_,Fourth|_] = [abyssian, bobtail, [bengal, birman]].`

Списки

Получение элементов списка

```
[abyssian, bobtail, [bengal, birman]]
```

- `[H|T] = [abyssian, bobtail, [bengal, birman]].`
- `[F,S|T] = [abyssian, bobtail, [bengal, birman]].`
- `[_,S|T] = [abyssian, bobtail, [bengal, birman]].`
- `[First,_,_,Fourth|_] = [abyssian, bobtail, [bengal, birman]].`
- `[_,_,[_|T]|_] = [abyssian, bobtail, [bengal, birman]].`

Операции со списками

Проверка вхождения элемента

Предикат `member/2 = member(?Elem, ?List)` имеет два аргумента — некоторый терм и список, и принимает значение `true` в случае, когда `?Elem` содержится в списке `?List`. В противном случае предикат принимает значение `false`.

Операции со списками

Проверка вхождения элемента

Предикат `member/2 = member(?Elem, ?List)` имеет два аргумента — некоторый терм и список, и принимает значение `true` в случае, когда `?Elem` содержится в списке `?List`. В противном случае предикат принимает значение `false`.

Как можно реализовать предикат `member/2` самостоятельно?

Операции со списками

Проверка вхождения элемента

Предикат `member/2 = member(?Elem, ?List)` имеет два аргумента — некоторый терм и список, и принимает значение `true` в случае, когда `?Elem` содержится в списке `?List`. В противном случае предикат принимает значение `false`.

Как можно реализовать предикат `member/2` самостоятельно?

```
member(X, [X|T]).
```

Операции со списками

Проверка вхождения элемента

Предикат `member/2 = member(?Elem, ?List)` имеет два аргумента — некоторый терм и список, и принимает значение `true` в случае, когда `?Elem` содержится в списке `?List`. В противном случае предикат принимает значение `false`.

Как можно реализовать предикат `member/2` самостоятельно?

```
member(X, [X|T]).
member(X, [H|T]) :- member(X,T).
```

Операции со списками

Проверка вхождения элемента

Предикат `member/2 = member(?Elem, ?List)` имеет два аргумента — некоторый терм и список, и принимает значение `true` в случае, когда `?Elem` содержится в списке `?List`. В противном случае предикат принимает значение `false`.

Как можно реализовать предикат `member/2` самостоятельно?

```
member(X, [X|T]).
member(X, [H|T]) :- member(X,T).

member(X, [X|_]).
member(X, [_|T]) :- member(X,T).
```

Операции со списками

Другие операции

- Подсчет длины списка.
- Конкатенация двух списков.
- Поиск префикса, суффикса и подсписка заданного списка.
- Поиск последнего элемента списка.
- Обращение списка
- Сортировка.

Домашнее задание

Домашнее задание

1. В программе `lists.pl` реализовать предикат `revAcc`, обращающий список более эффективно, чем предикат `rev`. Использовать дополнительный список для аккумуляции результата.
2. В программу `lists.pl` добавить реализацию предиката `listGen`, генерирующего по заданному числу N список длины N, заполненный случайными целыми числами (см. предикаты `randon` и `random_between`).
3. Реализовать в программе `lists.pl` алгоритм быстрой сортировки quicksort двумя способами и сравните их производительность на списках большой длины. Для разделения списка можно использовать встроенный предикат `partition/4`.
   - 3.1 Реализовать предикат `qsortLast`, который в качестве опорного берет последний элемент списка.
   - 3.2 Реализовать предикат `qsortMiddle`, где в качестве опорного берется центральный элемент списка.
4. Изменить программу `monkey.pl` так, чтобы избежать генерации бесконечного множества абстрактных решений.