

Seaman.h.zhang

博客园 :: 首页 :: 新随笔 :: 联系 :: 订阅 [XML](#) :: 管理 34 Posts :: 0 Stories :: 2 Comments :: 0 Trackbacks

公告

昵称: seaman.kingfall

园龄: 4年3个月

粉丝: 4

关注: 1

+加关注

搜索

找找看

谷歌搜索

常用链接

我的随笔

我的评论

我的参与

最新评论

我的标签

我的标签

练习题(6)

合一(3)

递归(3)

中断(2)

类型变量(2)

数字(2)

列表(2)

Haskell(2)

recursive(2)

比较(2)

更多

随笔分类

Haskell(2)

Prolog(32)

随笔档案

2015年8月 (7)

2015年7月 (22)

2015年6月 (5)

最新评论

1. Re:Learn Prolog Now
翻译 - 第一章 - 事实, 规则
和查询 - 第一节, 一些简单
的例子
学习!

--深蓝医生

2. Re:Learn Prolog Now
翻译 - 第一章 - 事实, 规则
和查询 - 第一节, 一些简单
的例子
翻译了这么多了, 而且每天一
篇, 不能望其项背啊。

Learn Prolog Now 翻译 - 第四章 - 列表 - 第三节, 递归遍历列表

内容提要

通过递归对列表进行遍历, 从而完成各种操作。

`member/2`这个谓词逻辑通过递归遍历了列表, 对列表头部有一些操作, 然后递归地对列表尾部做另外一些相同的操作。通过递归遍历列表在Prolog是十分普遍的做法,

事实上, 我们必须掌握这项技能。所以我们学习如下的例子。

当我们使用列表的时候, 我们经常会将一个列表和另一个列表进行对比, 或者拷贝一个列表的内容到另一个列表去, 或者翻译一个列表到内容到另一个列表去, 或者

类似到一些操作。这里有一个例子, 假设我们有一个谓词`a2b/2`, 有两个参数, 第一个参数是`a`的列表, 第二个参数是`b`的列表, 而且两个列表的长度相同; 比如, 如果

我们进行查询:

```
?- a2b([a, a, a, a], [b, b, b, b]).
```

我们希望Prolog能够回答true。另外一方面, 如果我们进行查询:

```
?- a2b([a, a, a, a], [b, b, b]).
```

或者进行查询:

```
?- a2b([a, c, a, a], [b, b, 5, 4]).
```

我们希望Prolog能够回答false。

当我们面对此类任务时, 通常最好的解决问题的方式是首先从最简单的情况入手。现在, 当使用列表时, 思考最简单的情况通常意味着从空列表开始, 而且在这个例

子中确实也是有意义的。毕竟, 什么样的列表是关于元素`a`最简单的列表? 是空列表。为什么? 因为它没有包含一个`a`元素。那么关于元素`b`最简单的列表呢? 也是空列

表。所以我们能够定义出最基础的信息如下:

```
a2b([], []).
```

这个明确的事实记录了关于`a`的空列表和关于`b`的空列表时相等的。虽然这个事实很明确, 但是它将会在程序中发挥至关重要的作用, 我们稍后就会看到。

直到现在一切还好, 那么如何进行下一步呢? 这里有一个思路: 对于更长的列表, 通过递归去思考。所以, 当谓词`a2b/2`去检查两个非空的列表, 一个是关于`a`

--Benjamin Yan

阅读排行榜

1. Learn Prolog Now 翻译 - 第三章 - 递归 - 第一节, 递归的定义(1168)
2. Learn Prolog Now 翻译 - 第一章 - 事实, 规则和查询 - 第一节, 一些简单的例子(1087)
3. Learn Prolog Now 翻译 - 第一章 - 事实, 规则和查询 - 第二节, Prolog语法介绍(781)
4. Haskell学习笔记二: 自定义类型(767)
5. Learn Prolog Now 翻译 - 第六章 - 列表补遗 - 第一节, 列表合并(753)

评论排行榜

1. Learn Prolog Now 翻译 - 第一章 - 事实, 规则和查询 - 第一节, 一些简单的例子(2)

推荐排行榜

1. Haskell学习笔记二: 自定义类型(1)
2. Learn Prolog Now 翻译 - 第三章 - 递归 - 第四节, 更多的实践和练习(1)

的, 另

一个是关于b的, 如何确认两个列表有相同的长度呢? 很简单: 当第一个列表的头部是a, 同时第二个列表的头部是b, 并且a2b/2能够证明两个列表的尾部有相同的长度,

我们就能够立刻写出如下的规则:

```
a2b([a | Ta], [b | Tb]) :- a2b(Ta, Tb).
```

解释一下: a2b/2能够成功的条件是, 第一个参数是头部为a的列表, 第二个参数是头部为b的列表, 同时a2b/2能够在两个列表的尾部操作成功。

这个定义有了很好的声明性。这是一个简单而又自然的递归谓词, 基础子句处理空列表, 递归子句处理非空列表。但是它实际是如何工作的? 即, 它的程序性含义是怎么样的?

比如, 如果我们查询:

```
? - a2b([a, a, a], [b, b, b]).
```

Prolog将会回答true, 也是我们期望的, 但是这一切是如何发生的?

让我们通过这个例子来学习。在例子中, 两个列表都是非空的, 所以事实子句不能提供帮助。所以Prolog就尝试使用递归规则, 现在, 查询能够满足这个规则 (因为第一个列表

的头部是a, 并且第二个列表的头部是b), 所以Prolog有了新的目标, 即:

```
a2b([a, a], [b, b]).
```

再一次地, 事实子句不能提供帮助, 但是递归规则能够再次被使用, 导致接下来的目标是:

```
a2b([a], [b]).
```

事实子句还是不能提供帮助, 但是递归规则可以, 所以我们又有了如下的新目标:

```
a2b([], []).
```

最终我们可以使用事实了: 它告诉我们true, 我们确实有两个关于a和b的、长度相同的列表 (空列表, 什么都没有), 这意味着如下的目标:

```
a2b([a], [b]).
```

也是成立的, 这会导致目标:

```
a2b([a, a], [b, b]).
```

也是被满足的, 所以原始的目标:

```
a2b([a, a, a], [b, b, b]).
```

是满足的。

我们总结这个过程如下: Prolog从两个列表开始, 通过检查第一个列表的头部是否为a, 第二个列表的头部是否为b, 然后去掉两个列表的头部; 而后, 使用相同的处理方式对两个

列表的尾部进行操作。为什么这个过程会终止? 因为每一次的递归后, 列表都会变短, 最终因为是空列表, 所以会终止。从这个角度来说, 程序中的事实会发挥

决定性的作用：

它给出true的回答，并且终止了递归，从而确保原始的查询是成功的。

了解查询失败也是同样重要的。比如，如果我们查询：

```
? - a2b([a, a, a, a], [b, b, b]).
```

Prolog会正确地回答false，为什么？因为经过了去掉头部-循环尾部的处理过程三次后，会剩下如下的目标：

```
a2b([a], []).
```

但是这个目标无法满足。如果我们查询：

```
a2b([a, c, a, a], [b, b, 5, 4]).
```

经过去掉头部-循环尾部的处理过程仅仅一次，Prolog会有如下的新目标：

```
a2b([c, a, a], [b, 5, 4]).
```

同样地，这个目标也无法满足。

以上是a2b/2简单的使用情况，但是我们还没有完全覆盖完所有的使用场景。Prolog的使用过程中，查询输入变量始终是一个尝试的好方式。这时a2b/2会有一些有趣的事情发生，

它会表现得像一个转换器，将元素a的列表，转换为元素b的列表。比如查询：

```
? - a2b([a, a, a, a], X).
```

```
X = [b, b, b, b]
```

即，元素a的列表，已经被转换为元素b的列表。类似地，通过在第一个参数位置使用变量，我们可以将元素b的列表，转换为元素a的列表：

```
?- a2b(X, [b, b, b, b]).
```

```
X = [a, a, a, a]
```

你能够根据这个结果知道它是如何发生的吗？总结一下：a2b/2是一个很简单的、通过递归遍历列表的例子。但是不要被它的简单性迷惑：此类程序展示了Prolog的基础功能。

无论是其声明形式（一个处理空列表的基础子句，一个处理非空列表的递归子句），还是具体执行的程序性（在列表头部做一些操作，然后对其尾部进行同样的递归处理），

都会在Prolog编程中反复使用。事实上，在你的Prolog生涯中，你会发现你在写各式各样的a2b/2谓词，或者是其更复杂的变体，许多时候加入了很多的装饰，但是本质上是一样的。

分类：Prolog

标签：递归, 遍历列表

好文要顶

关注我

收藏该文





seaman.kingfall

关注 - 1

粉丝 - 4

+加关注

0

0

« 上一篇: Learn Prolog Now 翻译 - 第四章 - 列表 - 第二节, 列表成员

» 下一篇: Learn Prolog Now 翻译 - 第四章 - 列表 - 第四节, 练习题和答案

posted on 2015-07-13 11:29 seaman.kingfall 阅读(636) 评论(0) 编辑 收藏

[刷新评论](#) [刷新页面](#) [返回顶部](#)

努力加载评论框中...

【推荐】超50万C++/C#源码: 大型实时仿真组态图形源码

【活动】看雪2019安全开发者峰会, 共话安全领域焦点

【培训】Java程序员年薪40W, 他1年走了别人5年的路

最新新闻:

- 微信公开课聚焦“增长”: 墨迹天气小程序DAU环比增100%
 - 知否 | 太空垃圾如何清理? 卫星测试用鱼叉击中太空垃圾碎片
 - 一线 | “美团配送”品牌发布: 对外开放配送平台 共享配送能力
 - 苍蝇落在食物上会发生什么? 让我们说的仔细一点
 - 科学家研究板块构造变化对海洋含氧量影响
- » 更多新闻...

Copyright © seaman.kingfall
Powered by: .Text and ASP.NET
Theme by: .NET Monster