分离逻辑

1 基本概念

考虑 WhileDeref 语言中有关内存地址的读写，需要新的程序逻辑推理规则。

- 断言 $P * Q$ 表示：可以将程序状态中的内存拆分成为互不相交的两部分，其一满足 P 另一个满足 Q；
- 即，程序状态 s 满足性质 $P * Q$ 当且仅当存在 s_1 与 s_2 使得：
  - s_1 满足 P，
  - s_2 满足 Q，
  - $s.\text{vars} = s_1.\text{vars} = s_2.\text{vars}$ 并且 $s.\text{mem} = s_1.\text{vars} \uplus s_2.\text{vars}$
    简写为 $s_1 \oplus s_2 \Downarrow s$，
- $P * Q$ 中的星号称为分离合取。

例如：

- `* x == m && * y == n && x != y` 可以写作 `store(x, m) * store(y, n)`，
- `* * x == 0 && * x != x` 可以写作 `exists u. store(x, u) * store(u, 0)`。

这里，我们使用 store(a, b) 表示地址 a 上存储了 b 这个值，并且仅仅拥有此内存权限。

下面是一些霍尔三元组的例子：

```
{ store(0x40, 0) }
* 0x40 = 0x80
{ store(0x40, 0x80) }
```

```
{ store(0x40, 0) * store(0x80, 0) }
* 0x40 = 0x80
{ store(0x40, 0x80) * store(0x80, 0) }
```

假设 $0 \leqslant m \leqslant 100$，

```
{ store(x, m) * store(y, n) }
* x = * x + 1
{ store(x, m + 1) * store(y, n) }
```

```
{ store(x, m) * store(y, n) }
* x = * y
{ store(x, n) * store(y, n) }
```

直观上，断言 $P * Q * R$ 表示：可以将程序状态中的内存拆分成为互不相交的三部分，分别满足 P、Q 与 R。

```
* x == n && * y == m && * z == 0 &&
x != y && y != z && z != x
```

可以写作：`store(x, n) * store(y, m) * store(z, 0)`。

```
* x == 10 && * (x + 8) == u &&
* u == 100 && * (u + 8) == 0 &&
x != u && x + 8 != u && x - 8 != u
```

可以写作：`store(x, 10) * store(x + 8, u) * store(u, 0) * store(u + 8, 0)`。
以下霍尔三元组成立：

```
{ store(x, 10) * store(x + 8, u) * store(u, n) }
* x = * * (x + 8)
{ store(x, n) * store(x + 8, u) * store(u, n) }
```

2 霍尔逻辑规则

内存赋值规则（正向）：

- 如果 P 能推出
  - e_1 能够安全求值并且求值结果为 a
  - e_2 能够安全求值并且求值结果为 b
  - $\exists u.\ \mathrm{store}(a, u) * Q$,
- 那么 $\{P\}\ * e_1 = e_2\ \{\mathrm{store}(a, b) * Q\}$
- 其中 a 与 b 都是与内存无关的数学式子。

变量赋值规则（正向）：

- 如果 P 能推出 e 能够安全求值并且求值结果为 a,
- 那么 $\{P\}\ x = e\ \{\exists x'.\ a[x \mapsto x'] = x\ \&\&\ P[x \mapsto x']\}$
- 其中 a 是与内存无关的数学式子。

框架规则：

- 如果 F 中不出现被 c 赋值的变量，并且 $\{P\}\ c\ \{Q\}$,
- 那么 $\{P * F\}\ c\ \{Q * F\}$

存在量词规则：

- 如果对于任意 a 都有，并且 $\{P(a)\}\ c\ \{Q\}$,
- 那么 $\{\exists a, P(a)\}\ c\ \{Q\}$

3 例子