第 7 章 分析动态

在第 5 章中，我们用动态结构定义了表达式 E 的分析。动态结构对于证明安全性非常有用，但是出于某些目的，比如编写用户手册，另一个被称为分析动态的公式更可取。分析动态是一个短语与它的值之间的关系，定义时没有详细说明逐步分析的过程。采用成本度量来指定分析的资源使用情况，用动态成本丰富动态分析。一个主要的例子是时间，它是根据表达式的动态结构来计算表达式所需的转换步骤数。

7.1 分析动态

分析动态包括分析判断 e ↓ v 的归纳定义，其中，封闭表达式 e 的值为 v。用下列规则定义分析动态：

$$\frac{}{num\,[\,n\,]\ \downarrow num\,[\,n\,]}$$

(7.1a)

$$\frac{}{str\,[\,s\,]\ \downarrow str\,[\,s\,]}$$

(7.1b)

$$\frac{e1\ \downarrow num\,[\,n1\,]\ e2\ \downarrow num\,[\,n2\,]\ n1\ +\ n2\ =\ n}{plus\,(\,e1;\ e2\,)\ \downarrow num\,[\,n\,]}$$

(7.1c)

$$\frac{e1\ \downarrow str\,[\,s1\,]\ e2\ \downarrow str\,[\,s2\,]\ s1\,\hat{}\,s2\ =\ s}{cat\,(\,e1;\ e2\,)\ \downarrow str\,[\,s\,]}$$

(7.1d)

$$\frac{e\ \downarrow str\,[\,s\,]\ |s|\ =\ n}{len\,(\,e\,)\ \downarrow num\,[\,n\,]}$$

(7.1e)

$$\frac{[e1/x]e2\ \downarrow v2}{let\,(\,e1;\ x.e2\,)\downarrow v2}$$

(7.1f)

一个 let 表达式的值取决于在绑定在主体中的替代部分。规则不是语法制导的，因为规则(7.1f)的前提不是该规则结论中表达式的子表达式。

规则(7.1f)指定定义的按名解释。对于按值解释，应采用以下规则:

$$\frac{e1\ \downarrow v1\ \ [\,v1/x]\ \ e2\ \downarrow v2}{let\,(\,e1;\ x.e2\,)\ \downarrow v2}$$

(7.2)

由于分析判断是归纳定义的，我们通过规则引入来证明其性质。具体地说，通过展示 P (e↓v)持有的属性，足以看出 P 是按规则（7.1）封闭的

1. P (num [n] ↓num [n]) .
2. P (str [s] ↓str [s]) .

3. P(plus (e1; e2) ↓num [n]) , 如果P(e1↓num [n1]) , P(e2↓num [n2]) , 且 n1 + n2 = n.

4. P(cat (e1; e2)↓str [s]) , 如果P(e1↓str [s1]) , P(e2↓str [s2]) , 且 s1 ^ s2 = s.

5. P(let (e1; x.e2) ↓v2) , 如果P([e1/x] e2↓v2) .

这一归纳原则与 e 本身的结构归纳不一样，因为分析规则不是语法制导的。

引理 7.1 如果 e↓v, 那么 v 是值。

证明：通过归纳规则（7.1）。除规则(7.1f)外所有情况是即时的。对于后一种情况，结果直接遵循以分析规则为前提的归纳假设。

7.2 关联结构与分析动态

对于 E，我们已经给出了两种不同形式的动态，自然会提出它们是否等价这个问题，但我们首先要考虑的是我们所谓等价的具体意义。结构动态描述一个逐步执行的过程，而分析动态则忽略中间状态，只关注初始状态和最终状态。正确的对应关系是结果动态的完整执行序列对应分析动态中的分析判断。

定理 7.2 对所有闭式 e 和值 v，e→*v iff e↓v.

我们应如何证明它？下面将考虑所有方向，首先是最简单的一个

引理 7.3 如果 e↓v, 那么 e→* v.

证明：通过归纳分析判断的定义。比如，假设 plus (e1; e2) ↓num [n]是分析加法的规则。归纳可知 e1→* num [n1]和 e2→* num [n2].理由如下

$$\begin{aligned} \text{plus}(e1;\ e2) &\to^* \text{plus}(\text{num}[n1];\ e2) \\ &\to^* \text{plus}(\text{num}[n1];\ \text{num}[n2]) \\ &\to \text{num}[n1 + n2] \end{aligned}$$

由此有 plus (e1; e2) →*num [n1 + n2]。其他情况处理相同。

对于逆向，回顾第 5 章中多步分析和完整分析的定义。因为 v 是值时 v↓v，结果表明，在逆向分析下分析是封闭的。

引理 7.4 如果 e→e′ 且 e′ ↓v 那么 e↓v

证明：通过归纳转化判断的定义。比如，假设在 e→e′ 时 plus (e1; e2) →plus (e1′ ; e2)。假设进一步有 plus (e1′ ; e2) ↓v，故有 e1′ ↓num [n1] ， e2′ ↓num [n2]且 n1 + n2 = n，v 是 num [n]。通过归纳 e1↓num [n1]，有 plus (e1; e2) ↓num [n]，满足要求。

7.3 类型安全，重查

第 6 章给出了类型安全的定义（定理 6.1），即保留性与进展性。当将这些概念应用到转换系

统给出的分析动态时，这些概念是有意义的，也正是贯穿全书的做法。但是如果我们在动态下分析呢?这种情况下如何保证类型安全？

回答是不能。虽然有分析动态的保存属性的类比，但是没有明确的进展属性的类比。保留性可以这么说，如果 e ↓ v 且 e : τ ，那么 v : τ 。通过对分析规则的归纳，可以很容易地证明这一点。但是进展性的类比是什么呢？我们可能会说，如果 e: τ，那么对于某个 v，e↓v。虽然这个性质对 E 成立，它要求的不仅仅是进展性------它要求每个表达式都有一个值!如果扩展 E 以承认可能导致错误的操作(如第 6.3 节中所讨论的)，或承认不终止的表达式，则此属性无效，即使进展仍然有效。

对于这种情况，一种可能的看法是得出这样的结论:类型安全不能在分析动态的上下文中得到适当的讨论，而只能通过结构动态来讨论。另一种观点是通过对动态类型错误的显式检查来检测动态，并且要显示任何带有动态弱类型的表达式都必须是静态弱类型的。在反例中重新声明，这意味着静态良类型的程序不能导致动态类型错误。这种观点的一个困难是，我们必须明确地解释一种错误的形式，以证明它不可能出现!然而，可以利用分析动态建立一种表面的类型安全。

我们定义一个判断 e ??表示表达式 e 在执行时出错。“出错”的精确定义通过一组规则给出，但那是在涵盖类型错误对应的所有情况的目的下。以下规则代表一般情况

$$\frac{}{plus\ (\ str\ [\ s]\ ;\ e2)\ ??} \tag{7.3a}$$

$$\frac{e1\ val}{plus\ (\ e1;\ str\ [\ s]\ )\ ??} \tag{7.3b}$$

这些规则明确检测附加到字符串中的错误应用；类似的规则控制着语言的每一个原始结构。

定理 7.5 如果 e ??, 那么不存在 τ 使 e : τ .

证明：通过规则（7.3）的归纳规则得到。例如，规则（7.3a），我们记有 str [s] : str, 因此 plus (str [s] ; e2) 是弱类型的。

推论 7.6 如果 e : τ , 那么¬(e ??)

除了不得不定义判断 e??的不便之外，仅为了表明它对于良类型程序是不可避免的，这在方法论上有明显的缺陷。如果我们省略了定义判断 e??的一个或多个规则，定理 7.5 的证明仍然有效；无法确保包含足够多运行时的类型错误检查。我们可以证明我们定义的那些不能在一个良类型程序中出现，但是我们不能证明我们已经涵盖了所有可能的情况。相比之下，结构动态不考虑任何弱类型表达式的行为。因此，任何弱类型表达式都将在没有我们明确干预的情况下“卡住”，而进展定理排除了所有这种情况。此外，转换系统更接近于实现--------编译器不需要为检测运行时类型错误做任何规定。相反，它依赖静态来确保这些不会出现，并且不为任何弱类型的程序赋予任何意义。因此，执行效率更高，语言定义也更简单。

7.4 成本动态

结构动态为程序提供了时间复杂度的自然概念，即达到最终状态所需的步骤数。然而，分析

动态并不能提供时间的直接概念。由于不清楚完成分析所需的单个步骤，我们不能直接读出分析值所需的步骤数。我们必须用成本衡量来扩充分析关系，从而产生成本动态。

分析判断具有形式 e $\downarrow^k$ v，意即 e 通过 k 步分析出 v

$$\frac{}{num\,[n]\ \downarrow^0\ num\,[n]} \tag{7.4a}$$

$$\frac{e1\ \downarrow^{k1}\ num\,[n1]\quad e2\ \downarrow^{k2}\ num\,[n2]}{plus\,(e1;\ e2)\ \downarrow^{k1+k2+1}\ num\,[n1\ +\ n2]} \tag{7.4b}$$

$$\frac{}{str\,[s]\ \downarrow^0\ str\,[s]} \tag{7.4c}$$

$$\frac{e1\ \downarrow^{k1}\ s1\quad e2\ \downarrow^{k2}\ s2}{cat\,(e1;\ e2)\ \downarrow^{k1+k2+1}\ str\,[s1\hat{}s2]} \tag{7.4d}$$

$$\frac{[e1/x]\,e2\ \downarrow^{k2}\ v2}{let\,(e1;\ x.e2)\ \downarrow^{k2+1}\ v2} \tag{7.4e}$$

对于 let 的逐值解释，规则(7.4e)替换为以下规则:

$$\frac{e1\ \downarrow^{k1}\ v1\quad [v1/x]\,e2\ \downarrow^{k2}\ v2}{let\,(e1;\ x.e2)\ \downarrow^{k1+k2+1}\ v2} \tag{7.5}$$

定理 7.7 相同类型的任何闭式 e 和闭值 v，e$\downarrow^k$v iff e $\rightarrow^k$ v

证明：从左到右对成本动态的定义进行归纳。从右到左进行自然归纳，通过对结构动态定义的内部规则归纳。

7.5 小结

分析动态和分类规则之间的结构相似性最早出现在标准 ML 的定义中(Milner et al., 1997)。分析语义的优势在于其直接性;它的缺点是不适合证明类型安全等属性。罗宾·米尔纳(Robin Milner)引入了恰当的短语“出错”，作为类型错误的描述。Blelloch 和 Greiner(1996)在一项并行计算研究中引入了成本动态（见 37 章)。

习题

7.1. 试说明分析是正确的: 如果 e↓v1 且 e ↓v2, 那么 v1 = v2.

7.2. 完成引理 7.3 的证明

7.3. 完成引理 7.4 的证明. 然后说明：如果 e1→* e′ 且 e′ 是值，那么 e↓e。

7.4. 在第 5 章中，使用 e ??表示 e 引起一个检查(或未检查)的错误，用检测错误增强分析动态。关于类型安全的证明还有何缺陷?你能想出更好的选择吗?

7.5 考虑下述形式的一般假设判断

$$x \downarrow v1, \ldots, xn \downarrow vn \vdash e \downarrow v$$

v1 至 vn 为值。此假设记作 Δ，被称为环境分析；它们提供了 e 中的自由变量的值。此判断假设 $\Delta \vdash e \downarrow v$ 叫做环境分析动态。

给出一个环境分析动态的归纳定义，不使用任何替代。特别地，应该包含下条来定义自由变量的取值：

$$\frac{}{\Delta,\ x \downarrow v \vdash x \downarrow v}$$

试证：x1↓v1, . . . , xn↓vn ⊢ e↓v iff [v1, . . . , vn /x1, . . . , xn] e ↓ v（使用逐值分析）