计算机网络 协议 形式描述 验证 (3)

计算机科学 1994Vol. 21, №. 6

11-15

基于Z的协议形式描述与验证

李腊元
(武汉交通科技大学 武昌 430063)

TP393

摘 要 This paper has made the study on protocol and verification in Z. It first describes main features of Z notation. Taking a simplified transport protocol as an example, it then details a specification of the protocol and discusses the verification of safety and liveness properties, which is based on Z. Finally, some conclusions are given.

关键词 Protocol, Specification and verification, Z.

一、引 言

在过去的十年，计算机网络与分布式系统已取得了很大的进展[6]。步入九十年代，各种新型通信技术和分布式应用已相继出现，并已对计算机通信软件发生了重要影响。这些新技术及应用主要包括高速光纤网、多媒体通信、宽带综合业务数字网(B-ISDN)、智能网络技术，如智能服务器、智能路由选择、智能协议开发环境等，以及综合语音、数据、图文和图象服务等等。为了适应这种形势的发展，一门新兴的学科—协议工程已应运而生[5]。协议工程，实际上是计算机硬件工程和软件工程的理论方法在协议设计和实现中的运用，但由于协议具有实时、互操作和同步性等要求，因而其复杂度要此一般传统软件大得多。它所包含的基本内容，亦叫作协议生命周期的主要阶段是：协议及服务的形式描述、协议验证和证实(validation)、协议形式描述的自动生成、协议变换、性能分析、自动实现和一致性测试。其中，形式描述是整个协议开发活动的基础和核心研究课题，它对生命周期的各个阶段均有直接影响。迄今为止，适用于协议及服务的形式描述技术(FDT)已开发出多种，如转移模型、程序设计语言模型、混合模型、代数说明、模态逻辑、公理方法、跟踪(traces)等。其中较有影响的还要算三种标准FDT：Estelle, LOTOS和SDL[2,4]。它们已对OSI协议的开发作出了很大贡献。然而，现已开发的FDT均在不同程度上还存在某些限制，其中最突出的问题是目前绝大多数的FDT，包括上述三种标准FDT，都不能直接处理活性性质。本文将研讨一种基于Z的形式描述和验证技术，旨在为协议提供一种新的FDT途径。

二、Z及其形式描述

现在，我们先描述Z的基本特征，然后介绍一个简化的传输协议，继而给出该协议的Z描述。

1. Z

这是一种基于集合论和一阶谓词逻辑的表示法[1]。为了实现分组命名(grouping naming)和建立各描述部分的关系，Z使用了图解演算(schema calculus)技术。它的数据类型主要以集合为基础，可提供大量的标准类型，如自然数等。其中说明部分的 $n:N$ 表示在自然数集合中(包括零)，n将有一个具体值。Z还允许引入或根据原有的类型来定义新的数据类型。例如：

$$[Person]$$
$$Color ::= red \mid blue \mid green$$
$$N_1 == N - \{0\}$$
$$LottoNums == \mathbb{P} N_1$$
$$Coord == N \times N$$

其中Person是不再进一步定义的基本类型；Color是红、兰、绿三值的集合；N_1是非零自然数的集合；LottoNums是以上所有可能数集之集合；Coord是所有有序对的自然数之集合。为了便于处理像关系、函数和顺序之类的类型，Z还提供了一些特殊的表示方法。其中关系可建模成有序对的集合；函数可视为一种特殊的关系；顺序又可作为一种特殊的函数。研究表明，Z由于其抽象性好，数据类型丰富灵活，描述能力

李腊元 教授，IEEE, INFOCOM等国际学术委员和国际论文评委，从事计算机网络和协议工程的教学和科研工作。

强，因而在协议系统的形式化技术中具有很好的应用前景。

2. 协议实例

本协议如图 1 所示，它描述了客户(Client)与服务器(Server)实体之间的交互，可视为一种简化的传输协议，具体描述如下：1)客户用户先发送一连接请求(conrq)信息，当客户实体接收到此 conrq 后，若响应此请求则发送一连接报文(con)给服务器实体。在此阶段，协议将从空闲状态变为连接准备(conp)状态。2)当服务器实体接收到 con 报文后将给它的用户发送一连接指示(conin)信息。在此阶段，协议将从 conp 状态变为准备响应连接(conpp)状态。3)服务器用户将发送一 conrp 原语(附加一特征参数)给其实体，以响应 conin。若此参数有效，则表示服务器接受此连接，否则该连接失败。当连接有效，服务器实体将发一 conr 报文给客户实体。在此阶段，协议将从 conpp 变为空闲或连接(conected)状态。4)客户实体接收到 conr 报文后，将发送一 concf 原语给其用户。此时，协议(客户)状态将为 conected 或空闲状态。

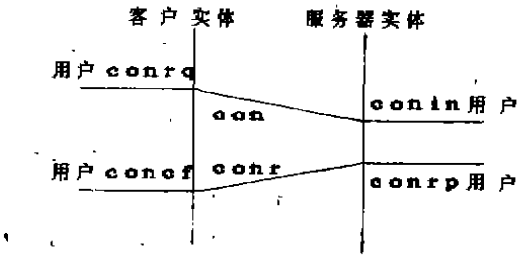

图 1 协议例子

3. 协议的 Z 描述

下面将以此传输协议为例，给出它的 Z 描述。根据全局状态和报文通道状态的变化，上述协议每个事件的作用，可通过下面的 Z 图解进行描述。

STATE
cs, sc, sclisv, servcli, recs, rscli : seq message
s : state

该图解表达了本协议系统的状态空间。其中 cs 和 sc 分别表示客户至服务器，以及服务器至客户之间的信道。sclisv 和 servcli 分别表示客户发给服务器，以及服务器发给客户的报文。recs 和 rscli 分别表示服务器和客户接收到的报文。s 表示该协议的全局状态。为了易于证明活性性质，这里考虑的是系统的全局状态，而不是客户和服务器的局部状态。

Init
Δ STATE
s' = idle
cs' = sc' = sclisv' = servcli' = recs' = rscli' = empty

该图解表达了本协议系统的初始状态。它断言协议状态是闲的(idle)，所有的信道是空的(empty)，因此所有的历史也是空的。

Conrq
Δ STATE
(s = idle
sc' = sc = empty
cs = empty
cs' = cs ⌢ ⟨con⟩
sclisv' = sclisv ⌢ ⟨headcs'⟩
servcli' = servcli
recs' = recs
rscli' = rscli
s' = comp)
∨
(s ≠ idle
sc' = sc
sclisv' = sclisv
servcli' = servcli
recs' = recs
rscli' = rscli
s' = s)

该图解表达了 conrq 事件发生后的某些影响或作用。它断言如果该状态是闲的，且 cs 信道是空的，那么将发生一状态转移，从而将导致出现一个 conpp 新状态，cs 将包含报文 con 和 sclisv 历史。此时全局状态其它的部分维持不变(即可由 lns' = lns 之类的公式表示)。例外情况由连接词"∨"后的第二个谓词提供，它主要用于处理发生 conrq 事件的前置条件不满足的情形。

Con
ΔSTATE
(s = conp
head(cs) = con
sc' = sc
cs' ⌢ ⟨headcs⟩ = cs
sclisv' = sclisv
recs' = recs ⌢ ⟨headcs⟩
servcli' = servcli
rscli' = rscli
s' = conpp)
∨
(s ≠ conp
sc' = sc
sclisv' = sclisv
recs' = recs
rscli' = rscli
s' = s)

该图解表达了 con 事件发生后的某些影响和作用。它断言如果该状态为 Conp，且报文通道 cs 头为 con，那么将发生一状态转移，从而导致出现一个 conpp 新状态。该转移还会引起报文通道 cs 删除其报文，以及 vecs 历史包括报文 con 等事件。图解中的其它谓词表达了事件 con 发生和前置条件不满足的情形。

Conrp
ΔSTATE
b?, b! : bool

```
(s=conpp
cs' =cs
sc' =sc⌢⟨conr⟩
sclisv' =sclisv
servcli' =servcli⌢⟨headsc'⟩
recs' =recs
rscli' =rscli
((b?=T
s' =conected
b! =b?)
∨
(b?=F
s' =idle
b! =b?)))
∨
(s≠conpp
sc' =sc
sclisv' =sclisv
servcli' =servcli
recs' =recs
rscli' =rscli
s' =s)
```

该图解表达了 conrp 事件发生后的某些影响和作用。它断言如果该状态为 conpp，且布尔参数为真(T)，那么将发生一状态转移，从而使协议变成连接(conected)状态；否则，如果布尔参数为假(F)，那么该状态将变闲。在这两种情况中，报文通道 sc 都包含报文 conr。例外情况(当前置条件不满足时)由上述第二个"∨"后的最后一个谓词提供。

```
Conr
ΔSTATE
b?:bool
((s=idle ∨ s=conected)
head(sc)=conr
cs' =cs
sc' ⌢⟨headsc⟩=sc
sclisv' =sclisv
servcli' =servcli
recs' =recs
rscli' =rscli⌢⟨headsc⟩
((b?=T
s' =conected)
∨
(b?=F
s' =idle)))
∨
((s≠idle ∧ s≠conected)
head(sc)≠conr
sc' =sc
sclisv' =sclisv
servcli' =servcli
recs' =recs
rscli' =rscli
s' =s))
```

该图解表达了 conr 事件发生后的某些影响或作用。它断言如果该状态是闲的，或处于连接状态，以及 sc 头为 conr 和布尔参数为 T 和 F，那么将发生一状态转移。对于前一种情况，新状态将是连接的，否则是空闲的。在两种情况下，通道 sc 都将使其报文移去，并将它添加到历史 rscli 中去。例外情况由最下面一个析取连接词的最后一个谓词提供。

在上述各图解中，各括号的含义主要用于说明一个特定谓词公式的辖域。

三、安全性与活性验证

为了运用 Z 来证明协议的安全性与活性，我们先引入某些概念和记号，约定初始状态用 s_0 表示；最终状态用 s_ω 表示。在 Z 中，模态逻辑最终(eventual)算子的作用可由下式表示：

$$\exists i:\mathrm{nat}; s':\mathrm{Schema}; s:\mathrm{seq}\ \mathrm{Schema} \mid s(i)=s'$$

即存在一系列图解 s，在点 i 某性质适合于图解 s'。

在 Z 中，模态逻辑今后(henceforth)算子的作用可由下式表示：

$$\forall s:\mathrm{seq}\ \mathrm{Schema} \bullet \mathrm{Property}$$

其中 Property 表示某谓词，即适合于所有状态的某性质。

已知一系列图解 s，其终止序号为 i，那么由·(由图解 N 表征)给定的下一状态则可通过将图解 s(i)与 N 顺序组合来获得，此时，该顺序的长度为 i+1。

已知图解 s_1 和 s_2 的顺序，那么 $(s_1;s_2)$ 也是一个图解组合，且 $(s_1;s_2)(i)$ 是一图解。

为了证明图解 A 与图解 B 能组合，只需证明 B 的前置条件与 A 的后置条件相同，且 B 的输入与 A 的输出匹配。

下面，以前述传输协议为例讨论基于 Z 的安全性与活性验证及分析。

1. 安全性

安全性涉及协议某些公理的正确性。一个安全状态是指决不发生有害的事情。某些安全性可从协议系统的状态得到，其它的安全性则要通过考察历史状况才能获得。安全性的含义可运用 Z 进行如下形式描述：

```
State Safety Invariants.
Behav Safety Invariants:System→P Property
∀ s:System. State Safety Invariants(S)=
⋂{st:Reachable(s)•Sprops(st)}
∀ s:System; P:Property•
P∈ Behav Safety Invariants(s)⇔
∀ b:s; t:Time•p∈b preps (prefix(b.t))
```

让我们考虑用谓词演算表达的安全性质：

$$\forall eh:\mathrm{eventhist}.\ \mathrm{sent}(eh)=\mathrm{received}(eh)\frown\mathrm{transit}(eh)$$

它断言所发送的报文与所接收的报文相等，直到转移发生为止(对于所有的事件历史—eventhist)。该性质也可用上述概念和记号来表达：

$$\forall s:\mathrm{seq}\ \mathrm{Schema}, \exists i:\mathrm{nat}(s(i)\bullet\mathrm{sclisv}=s(i)\bullet\mathrm{recs}\frown\mathrm{transit})$$

其中，转移表达了对于各种状态(即各种 i 值)的报文通道 cs 的状态，到用符号·，可为访问一个图解的有

关属性提供手段。上述性质的正确性可通过将不同的 i 值代入上式来验证：

$i=1 \Rightarrow empty = empty \frown empty$
$i=2 \Rightarrow \langle con \rangle = empty \frown \langle con \rangle$
$i=3 \Rightarrow \langle con \rangle = \langle con \rangle \frown empty$
$i=4 \Rightarrow \langle con \rangle = \langle con \rangle \frown empty \cdots$

无死锁是安全性所包含的一个主要性质。一个死锁状态是指系统不会有进一步的转移发生。该状态并非终止状态，且报文通道均空，亦即协议双方将无止境地停留在该状态[5]。在本例中，如果对于某个 i 有：

$s(i) \cdot s' = state \wedge cs(i) = sc(i) = empty$

那么当 s(i+1)不存在时，则发生死锁，亦即没有下一个状态。为了证明该协议无死锁，必须引入下式：

$\forall i' (s(i) \cdot s' = state \wedge cs(i) = sc(i) = empty) \Rightarrow (s(i+1) \cdot s' = state \mid \wedge state \mid \neq state)$ 。

上式的性质容易通过 case 分析来验证(即将 i 的具体值代入)。实际上所有的安全性质都可通过 case 分析来证明其正确性(即通过按顺序将值代入某个索引字来进行)。

2. 活性

活性涉及协议的进展情况。一个活性状态是指某些"好"事情能有效地发生。形式地讲，活性性质是一种行为性质，对于每个完整的行为它必须为真，否则为假。活性性质实际上对于无限和有限系统均适用。关于这一点目前尚无统一的结论，有的认为它只适用于无限系统，并且实际上包含无限推理过程，因而对于实际系统并不适用。还有的认为活性性质可为协议的某些特性提供高级的规范说明，并能在一定程度上减少协议实现错误。总的来看，协议的活性验证要比安全性验证更为困难。我们的研究表明，迄今为止，绝大多数形式描述技术，包括 ISO 和 CCITT 提出的三种标准形式描述技术：Estelle (IS9074)、LOTOS (*IS8807*)和 SDL(*Z. 101-104*)均不能直接刻画协议的活性性质。模态逻辑虽然可用于分析和证明协议的活性性质，但其证明过程较复杂烦琐，难以自动实现。本研究表明，Z 不必引入模态算子即可用于分析和证明协议的活性性质，且较之模态逻辑的方法直接简单。

活性的含义可运用 Z 形式描述如下：

$$\begin{array}{l} Liveness\ Invariants : System \rightarrow \mathbb{P}\ Property \\ \forall s : System; p : Property \cdot p \in Liveness\ Invariants(s) \\ \Leftrightarrow \forall b : s \cdot p \in bprops(b) \\ \exists b : s; t : Time \cdot p \notin bprops(prefix(b, t)) \end{array}$$

若已知当前的某一条件集(状态)，则可推断在将来的某一时刻，必将使其中一个条件为真。这可用于证明协议的活性性质。例如，在前述传输协议例子中，若在某状态 s_n 报文通道 cs 为空，那么将存在一状态 s_n 使 cs 为空，但对于某个中间状态 s_i，cs 非空。这个性质的证明主要包括证明存在一系列图解，它们相继改变协议的状态，从 s_0 开始到 s_i，最后到 s_n。进而证明报文通道 cs 满足上述条件。以上活性命题证明包含以下步骤：

步骤1　求出满足状态 s_0 的一图解，在 s_0 中的 cs 为空，此时 $s_0 = idle$。

步骤2　求出一图解，该图解具有如下前置条件：$s = idle \wedge cs = empty$。

步骤3　确定该图解对于变量 s 和 cs 的影响，它提供 s_i 和 cs'，以此作为下一图解的前置条件。

步骤4　分别对变量 s_i 和 cs' 重复步2和步3，直到 $s_n = conpp$ 和 $cs' = empty$ 为止。

若将某些基本的算子进行组合，则可获得复合的模态逻辑描述形式。通过推断该协议总能进入空闲状态，则可将安全性性质 $= \langle\ \rangle s = idle$ 描述如下：

$\forall s_1 : seq\ Schema, \exists s_2 : seq\ Schema, s_3 : Schema, i : nat \cdot (s_1 ; s_2)(i) = s_3 \wedge S_3 \cdot state = idle$

其中 i 取1与 $(s_1 ; s_2)$ 长度之间的值。

这样，活性性质也可进行类似地描述：

$s = idle \Rightarrow \langle\ \rangle \cdot s = idle$

即已知过去在某点曾有这个状态，那么协议最终将进入此空闲状态：

$s(i) \cdot state = idle \Rightarrow \exists s : seqSchema, i : nat, s_1, s_2 : Schema \mid s(i) ; s_1 = s_2 \wedge s_2 \cdot state = idle$

很显然按这种方式继续下去，其它的模态算子可简单地由 Z 的谓词表达式所取代。

在上述安全性与活性的分析中，推导图解的顺序是其中的关键。为此我们先考虑下面的性质：

$(s(i) \cdot s' = idle \Rightarrow (s(i+1) \cdot s' = conpp \wedge s(i+1) \cdot cs' = empty)$

这需要为假，因为没有图解能与 s(i)组合，以给出 $s(i+1) \cdot s' = conpp$。其推理如下：

$$\begin{aligned} & T \Rightarrow (F \wedge F) \\ & \Rightarrow F \end{aligned}$$

在上面的推论中，前项为真。这是因为存在一个状态且满足 s = idle；其后项中的第一个合取运算必为假，这是因为现在已知道了 i 的值(即作为 $s(i) \cdot s' = idle$ 的结果)。类似地，第二个合取运算也必定为假。这种推理方法也适用于证明所有其它类似的公式。

考虑下面的谓词：

$(s(i) \cdot s' = conp \Rightarrow (s(i+1) \cdot s' = conpp \wedge s(i+1) \cdot cs' = empty)$

它为真是容易得到证明的：

$$\begin{aligned} & T \Rightarrow T \wedge T \\ & \Rightarrow T \end{aligned}$$

下面，我们进一步给出适用于两个图解之间的推理规则。如果 B 是 A 的顺序组合，则有 A⇒B。在下面的规则中，符号⇒与一般的蕴涵关系略有不同：

规则1 如果 A 为假，那么 A⇒B 也为假。这可能由两个方面的因素所引起：1)在逻辑推理中，A 中的一谓词为假；2)在两个图解之间的等式中，A 中一项为假。在这两种情形中，由于 A⇒B 都意味着组合中的蕴涵关系，因此整个公式必然要受到 A 的影响。

规则2 如果 A 为真且 B 为真，则 A⇒B 为真。

规则3 如果 A 为真且 B 为假，则 A⇒B 为假。

需要说明的是上面所涉及到的谓词，均假设为一阶谓词演算。

四、结 论

适用于协议工程的形式化技术和方法虽已取得了长足的进步，但正象协议工程本身那样，仍处在不断发展和完善之中，有许多问题尚待进一步探讨。本文研究了基于 Z 表示法的形式描述与验证技术，给出了基于 Z 的协议形式描述实例，对协议的两类重要性质，即安全性与活性进行了形式定义，讨论了这两类性质的验证问题，从而得到了一种适用于通信进程的时态推理途径。不同图解之间的推导，首先应确保这些图解能形成一个顺序组合的图解链，其次应注意运用标准概念及记法表示所处理的图解，从而使输出必须与输入匹配，前置条件必须与后置条件匹配。研究表明，这种形式化途径对于协议的形式描述和正确性(安全性与活性)，尤其是活性验证是适用有效的。

参考文献

[1] Woodcock, J. Mathematics as a management tool: proof sules for prmotion. Technical report PRG, Oxford Univ., UK(1989)

[2]Li Layuan(李腊元). A formal technique for communication protocol specification, Proc. IEEE INFOCOM. 1989. USA

[3]Li Layuan(李腊元). Formal description and verification of a transport protocol for local networks, J. of Comput. Sci. & Technol., Vol. 5, No. 4, 1990, Sci. Press, Allerton Press, USA

[4]Li Layuan(李腊元). A new formal method for communication protocol specification, J. of Comput. Sci. & Technol., Vol. 4, No. 1, 1989, Sci. Press, Allerton Press, USA

[5]李腊元，通信协议工程学进展，计算机研究与发展，1993.7

[6]李腊元等，计算机局部网络，湖北科技出版社，1987

(上接第81页)

第5期

第6期(略)