

13-16

并发模型分析

刘海燕 陈火旺

(国防科技大学计算机系 长沙 410073)

TP301.2

摘要 并发一直是计算机科学中矛盾的源泉和争论的焦点之一。本文分析和比较了三种并发模型: Petri 网、CSP 和 CCS 的发展历史、描述方法、处理并发与不确定性的方式等。

关键词 并发, 并发模型。形式语言 计算机语言 petri 网

一、引言

计算机语言的形式语义是计算机科学的重要基础, 对它的研究始于六十年代, 并由此蓬勃发展起来, 先后出现了四大流派, 即操作语义、指称语义、公理语义及代数语义, 它们从不同的侧面、用不同的理论为程序设计语言含义的描述提供了很好的方法, 到目前为止, 对顺序程序设计语言的语义的研究已趋于成熟。六十年代末七十年代初, 计算机界开始遇到并发现象, 但由于并发机的体系结构日趋多样化以及并发程序设计思想和并发语言的不断变化, 人们对并发缺乏理解, 对并发所造成的后果心存迷惑和恐惧心理。几十年来, 并发一直是计算机科学的重要课题, 吸引着大批有创新能力和献身精神的科学家。从六十年代的 Petri 网论到七、八十年代的 CSP 和 CCS, 再到它们的各种精化、导出版本以及其它并发模型, 对并发的研究可谓丰富多彩。本文主要介绍这三种模型, 并对它们进行比较和分析。

二、发展历史和应用^[4,5]

1962 年, C. A. Petri 以其论文“Kommunikation mit Automaten”在达姆施塔特大学获博士学位, 标志着 Petri 网的诞生, 1970 年 Petri 又将其发展成通用网论, Petri 网是为了帮助设计和分析并发系统而开发的理论, 被认为是第一个通用的并发理论, 它是自动机理论的扩充。1972 年在前西德汉堡组织召开“petri”网高级会议, 1986 年也是在前西德的巴特霍涅夫再次召开了 Petri 网高级会议, 自此, 每年都有 Petri 网的国际会议召开。这些会议的目的是使现有的知识系统化, 并向人们提供有关 Petri 网的最新成果, 从而指导未来的研究方向。Petri 网已经开发的许多理论反映了并发领域的各种问题, 包括系统组织的许多原理和分析技术, 被用于计算机科学的

许多方面。CSP 和 CCS 的研究都是在 Petri 网的基础上展开的。

通信顺序进程(CSP)是 Hoare, C. A. R 于 1978 年提出的一个并发程序设计语言, 它把同步输入/输出语句作为基本原语, 通过组合和握手通信来扩展 Dijkstra 的卫士命令。CSP 一经出现就被广泛地传播和使用, 在 CSP 的早期版本中没给出其精确的数学语义, 所以对于象“并发嵌套”、“递归调用”等重要问题无法解决, 于是出现了更新更抽象的版本 TCSP(Theory of Communicating Sequential Process), TCSP 在很大程度上受 Milner 的 CCS 的影响, 它的并发和通讯方法及语义的研究与 CCS 很类似。TCSP 的“失败语义”给出其精确的数学描述, 使 CSP 理论更加可靠完善。

并发的数学模型的主要突破是 Robin Milner 完成的, 在他于 1980 年发表的“通信系统演算(CCS)”中, 给出了一个建立于极小原语集上的函数式程序设计语言 CCS, 为通信的、非确定的并发系统提供了一个通用数学模型。在 Milner 84 年的关于 CCS 的讲话中, 他又给出了更基本的演算, 并证明原来的演算是后者的一个导出演算。此后 CCS 理论的发展主要是围绕更基础的演算进行的。1989 年 Milner 出版了“通信与并发”这本书, 提出“进程演算”, 进一步延拓了 CCS 理论, CCS 框架的开发具有深远的影响, 在 Milner 框架内又出现了许多类似的语言及等价性的研究。

三、PETRI 网、CSP、CCS 模型比较^[1,2,3]

1. 理论出发点

CSP 和 CCS 都是基于“交互的不可分割性”而开发的, 即交互动作是个原语。它们的出发点是: 小系统可以构造大系统, 故其研究重点应该是原子系统的描述, 系统的构造规则及抽象方式。

在 CSP 中,一个系统可以分解成多个子系统,这些子系统并行地运行着,它们之间以及它们与环境之间相互作用,子系统还可以继续分解成更小的子系统。系统与环境均用进程描述,一个进程就象一个服务装置一样,通过通信与环境(其它进程)交互。

CCS 把并发与通信看作一个互补的概念,各种系统都是由几个部件组成的,每个部件在自己的生命周期中保持其个性,它们独立地并发地工作着,并通过各部件间的通信来维持系统的统一性。这些部件称为 agent,agent 是个应用广泛的概念,任何有离散动作的系统都称为 agent,复杂的组织可以看作 agent 组成的网络。CCS 不区分系统及成分,也不区分具有子结构的系统与无子结构的系统,它们统一用 agent 概念表示。

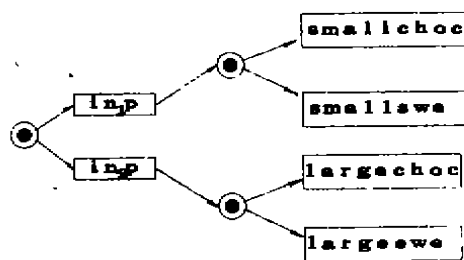
Petri 网论强调的是系统所发生的事件之间的因果关系,用因果关系建立事件发生间的相互依赖关系。一个系统是用它所能完成的事件(动作)以及事件发生的条件描述的,事件与条件间关系用网络描述,给定系统的初始条件,它的一次运行称为一个进程。Petri 网就是在这个框架下研究系统并发性的。

2. 三种理论对系统的描述

下面我们通过一个例子说明三种理论的基本描述方法。

假设存在一个自动售货机,其整个生命周期中的全部可能行为是:接受投入的硬币,放出相应的巧克力或糖,与自动售货机发生交互作用的是人,人的可能行为是投入一元的硬币或投入二元的硬币,则:

1)用 Petri 表示该系统为:



说明:○表示条件,⊙表示条件满足,□表示事件 e,→表示因果关系。

2)CSP 中定义进程 V 表示自动售货机:

$V::in_1p \rightarrow (smallchoc \rightarrow STOP \mid smallswe \rightarrow STOP) \mid$
 $in_2p \rightarrow (largechoc \rightarrow STOP \mid largeeswe \rightarrow STOP)$

进程 M 表示人: $M::(in_1p \rightarrow STOP \mid in_2p \rightarrow$

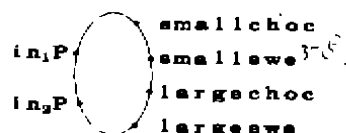
STOP)

人机系统表示为: $V \parallel M$

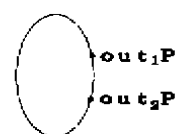
说明:进程可执行的动作集合称为它的字母表,“→”表示动作的先后,“|”表示选择,“STOP”表停机进程, $P \parallel Q$ 表示进程 P 与 Q 并发。

3)CCS 中用 agent V 表示自动售货机,那么对外界观察者来说,它总是吃进一枚硬币,输出相应的食物,所以对外界来说它的所有动作为 $\{in_1p, in_2p, smallchoc, smallswe, largechoc, largeeswe\}$,至于其内部结构,我们并不关心。

agent V 表示为:



进程 M 表示为:



通过重新标记函数把 P 与 M 联结起来:

$$f(x) = \begin{cases} a & \text{若 } x = in_1p \\ b & \text{若 } x = in_2p \\ x & \text{其它} \end{cases}$$

$$g(x) = \begin{cases} a & \text{若 } x = out_1p \\ b & \text{若 } x = out_2p \\ x & \text{其它} \end{cases}$$

系统表示为 $(V[f] \mid M[g])$,具有互补名字(a 与 $\bar{a}$, b 与 $\bar{b}$)的端口被联接起来,若要屏蔽它们之间的通信动作,则用 $(V[f] \mid M[g]) / \{a, b\}$ 表示,那么发生在它们之间的通信动作作为系统的内部动作,在系统外不可见。

3. 通信方式

CSP 和 CCS 中进程间的通信采用的是“握手(handshake)”方式,通过显式的通信命令建立而不是利用共享变元进行隐式通信。

CSP 中由一对输入/输出命令实现通信和同步,当进程 P_i 要把值 e 传递给进程 P_j 时, P_i 执行命令: $P_i!e, P_j$ 执行命令: $P_j?x$,则通过推迟输入(输出)动作直至另一个进程的相应的输出(输入)动作被执行,这样可以实现进程间的同步通信。在 CSP 中,通信的双方必须明确地指出对方的名字。

通信是 CCS 的核心内容,一个 agent 的动作可以分为两类:一类是它与邻近 agent 的交互动作,另一类是自己独立的动作,这类动作可以与其它 agent 的动作并发,但这类动作往往是该 agent 的内部构件间的通信,所以研究一个 agent,可以忽略其内部

动作,只观察其与外界的交互能力。一个系统的行为无非是它的全部通信能力。两个进程要实现通信,只需使一个进程的输入端口与另一个进程的某个输出端口具有互补名字(用重新标记函数),并将它们并发组合,例如:P 进程有输入端口 in,Q 进程有输出端口 out,取 $f_1(out)=a, f_2(in)=a$, 则 $(p[f_2] \mid q[f_1]) \setminus (a)$ 在 P,Q 间建立隐式通信通道。Q 放在 out 端口的信息被 P 从它的 in 端口取走,而且外界看不到这些端口和通信行为。

Petri 网中没有特别的通信手段,它的同步、共享都可用公共的前提和后提实现,有的 Petri 网中引入“流量、容量”等概念,如位置/变迁(P/T)系统,可以描述数值的变化,通过事件的“点火”使信息在网上传播。

4. 并发

Petri 网中所描述的并发是“真并发”,即并行。如果两个事件的前提和后提两两不相交,则这两个事件间无因果关系,当它们都可以点火时,它们点火的先后时间在时间上无法区分。在 Petri 网中系统不存在统一的时钟,除因果关系外没有其它信息可以用来判断两个事件的依赖关系。

CSP 和 CCS 都是“交叠式”并发模型,用顺序化和非确定性来模拟并发,系统内并发的进程,其先后次序不确定,如果有多个进程在活动,那么它们的动作集合按执行时间的先后一定可排成一个全序。

CSP 中两个进程的并发组合表示为 $P \parallel Q$, 它的意义是: $P \parallel Q$ 进程的动作,如果属于 P 和 Q 的公共字母表,那么该动作由 P 和 Q 共同完成;否则,是 P 的动作和 Q 的动作的任意穿插。CSP 的并发组合也同时完成了进程间的同步。

CCS 中的并发组合“ $\mid$ ”与“隐藏”有关,并发系统表现为受限的组合形式:

$$(P_1[f_1] \mid \dots \mid P_n[f_n]) \setminus N$$

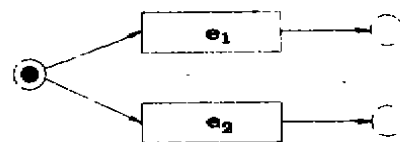
当系统内两个 agent 有互补端口时,它们之间形成通道。它们间的同步通信动作产生一个内部动作,统一用 τ 表示, τ 不能再参与其它的同步动作了,所以 CCS 只能完成二元同步,而在 CSP 中,进程 $(P_1 \mid P_2 \mid \dots \mid P_n)$ 对公共字母表中的动作, $P_1, P_2, \dots, P_n$ 必须同时参与,故 CSP 可实现多个进程的同步。

在 CCS 和 CSP 中都采用了交叠式并发,这大大简化了数学处理,从而可集中精力去研究构造、抽象等问题。

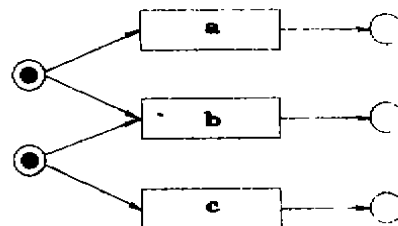
5. 不确定性

Petri 网中的不确定性是由“矛盾和并发”引起

的,如下图所示:



其中 e_1 和 e_2 都具备了点火条件,但又不能都点火,这是一种矛盾现象。若 e_1 先点火则 e_2 就不可能再点火了,反之亦然,从而产生了不确定性。当网中同时存在矛盾和并发现象时,由于并发事件点火顺序完全不确定,矛盾的产生和消失往往取决于并发事件的结局,例如:



事件 a 与 b, b 与 c 之间存在矛盾, a 与 c 可并发点火,一旦 a 或 c 点火,则上面的矛盾便消失了。

CSP 中不确定性进程是指“那些具有一定范围的可能行为但其环境不能影响甚至观察不到它是如何选择”的进程,有非确定“或”算子“ $\sqcup$ ”,和一般选择算子“ $\square$ ”,前者的含义是: $P \sqcup Q$ 进程或按 P 动作或按 Q 动作,但进程在二者间所进行的选择是在外界环境不知道或不能控制的情况下进行的。 $P \square Q$ 的含义是:如果环境开始就试图实行对选择控制,那么环境就能控制 $P \square Q$,选出 P 或 Q,但如果第一个动作既可以是 P 也可为 Q,则它们之间的选择将退化或非确定选择。

CCS 的不确定选择符“ $+$ ”, $P+Q$ 的第一个动作便决定是按 P 动作还是按 Q 动作,而无论该动作是通信还是内部动作(τ),组合操作“ $\mid$ ”也会引起非确定性, $(P \mid Q) \setminus L$, 对于非 L 中的动作,或 P 前进一步或 Q 前进一步,且另一个进程不变化,也可以 P 与 Q 进行内部通信(τ),这三种可能性间是不确定选择关系。

6. CCS 与 CSP 模型复杂度比较

CSP 用单一的简单模型达到简明性,用该模型定义许多算子,算子所对应的概念清晰,从而算子所遵循的代数法则简明清晰。

CCS 给出了在不同抽象程度上构造和比较不同

模型的框架,采用最基本的原子算子和最简单的语法集,达到最强的表达能力,便于进行数学的研究。

三、结论

上面我们分别介绍和比较了并发领域的三个重要模型 Petri 网、CSP、CCS,这三种模型各有千秋,在一些方面,某个模型可能优于其它模型,而在另一些方面,其它模型可能更方便。目前,既有局限于某一模型内的研究又有对三者关系的研究。值得注意的是,某个领域的成果对其它领域的研究具有启发和指导作用。

三种理论虽然比较成熟地解决了并发问题,但是由于它们都是抽象掉某些因素后简化了的模型,现实中还有许多与并发有关的问题尚未解决,而且并发理论本身复杂性太大,使得对它的进一步的理论研究和实践探讨相当困难,因此需要更多有献身

精神的科研人员投入到并发的研究开发中来。

参考文献

- [1] 陆汝铃,形式语义,科学出版社 1992
- [2] 周巢尘,通信顺序进程,北京大学出版社 1990
- [3] Robin Milner, Communication and Concurrency, 1989 published by Prentice Hall International (UK) Ltd 66 Wood Lane End, Hemel Hempstead, Hertfordshire, HP2. 4RG
- [4] Mogen Nielsen, CCS—And its relation to Net-Theory, Advances in Petri Nets 1986, Part I, Lecture Note in Artificial Intelligence 254 1986
- [5] Ernst-Rüdiger Olderog, TCSP—Theory of Communicating Sequential Process, Advances in Petri Nets, Part I, Lecture Note in Artificial Intelligence 255, 1986

(上接第 3 页)

进行编辑。

三、小 结

自 1988 年提出 Dexter 模型后,它对超媒体系统的设计和实现起着重要的指导作用。在设计新一代超媒体系统时 Dexter 模型是一种必不可少的参考模型。但是 Dexter 模型还处于一个早期阶段,它所定义的功能远比现有的系统强。特别是在以下两个方面:

- Dexter 模型支持多头链,而目前在超媒体系统中实现多头链还非常困难。

- 目前绝大部分超媒体系统还不支持复合元素。

因此在设计未来的超媒体系统时,要重点解决以下几个方面的问题:

- 应当设计支持复合结构的超媒体系统。
- 支持协作与共享的超媒体结构。
- 支持与时间有关的媒体。
- 能够支持其它类型应用程序的开放式超媒体系统。

参考文献

- [1] Akscyn, R., et al., KMS: A distributed hyper-

text for managing knowledge in organizations, CACM, Vol. 31, No. 7, 1988

- [2] Kaj Gronbæk, et al., Design issues for a Dexter-based hypermedia system, CACM, Vol. 37, No. 2, 1994
- [3] Kaj Gronbæk, et al., Cooperative hypermedia systems; a Dexter-based architecture, same to [2]
- [4] Halasz, F. G., Reflections on NoteCards, Seven issues for the next generation of hypermedia systems, same to [1]
- [5] Frank Halasz et al., The Dexter hypertext reference model, NIST Hypertext standardization Workshop, Gaithersburg, MD, January 16-18, 1990
- [6] Halasz, Frank et al., The Dexter Hypertext reference model, same to [2]
- [7] Lynda Hardman, et al., The Amsterdam hypermedia model; adding time and context to the Dexter model, same to [2]
- [8] John leggett et al., Viewing Dexter with open eyes, same to [2]