嵌入式系统中同步式语言的智能化

On the Intelligence of Synchronous Programming Languages of Embedded Systems

苏运霖

Su Yunlin

(**广西大学梧州分校,广西梧州** 543002)

(Guangxi University Wuzhou Branch, Wuzhou, Guangxi, 543002, China)

摘要:分析嵌入式系统中同步式语言的特征和智能特性方面的不足,提出嵌入式系统同步式语言智能化扩充时应增加的设施,并给出智能化同步式语言的具体模型。

关键词:同步式语言 嵌入式系统 智能化 模型

中图法分类号:TP312 **文献标识码**:A **文章编号**:1002-7378(2005)04-0236-03

Abstract: The characteristics and intelligentizing of synchronous programming languages in the embedded systems are analyzed. The extension facilities for the intelligentizing of the synchronous programming languages in the embedded systems are listed, and the relative model is presented.

Key words: synchronous, embedded system, intelligence, model

迄今为止,嵌入式软件还没有引入人工智能的概念,而智能化的嵌入式软件是嵌入式软件发展的必然趋势。今天的嵌入式软件是在14年前提出的同步式语言的概念基础上实现的[1]。现在,同步式语言已经被建立为进行模拟、描述、验证和实现实时嵌入式应用所选择的一种技术。同步性的范例已经作为基于数学上的健全性的工具,成为对于工程师友好的一种设计方法。

以嵌入式控制系统领域为中心目标为例,同步式语言已经证明自己是最好的工具。然而,如果我们希望嵌入式系统具有更强的人工智能特征,则同步式语言不具备这一能力。因此,本文试图在同步式语言的基础上,增加人工智能的特征,这样,同步式语言就成为一个宿主语言,而所增加的智能特征使它满足实时智能嵌入式系统的要求。

1 同步式语言

要满足实时嵌入式系统需要,同步式语言必须有坚实的数学基础,以确保同步的正确实现,并且与作为分布式系统特有的、确定的并发性相结合。因此,以安全为关键的嵌入式系统的设计者,其主要目标是使他自己、顾客,以及评估权威都能对于这个设计和实现的正确性表示确信无疑。同时,他还必须使得开发和维护的费用保持在控制之下,且满足系统的非功能方面的限制,例如费用、电源、重量,或者系统结构本身。满足这些目标要求的设计方法和工具还要与现有的设计流程实现无缝地集成,就需要在坚实的数学基础上进行设计。

同步式语言在集成方面的需要是明显的,且对于一个设计的信心至高无上。在设计者的经验之外的任何东西几乎毫无疑问地会降低这个自信,从而也影响这个系统的质量。使用坚实数学基础的好处在于,有能力对系统的操作进行形式推理。这就使得评估变得容易,因为它减少了二义性,并能对系统操作构造关于系统操作的证明,这也就改进了实现的过程,因为它使得为满足系统非功能方面的限制,能够使程序的操作自动地构造出有用的不同实现。

上述的这些分析,导致了对于同步式语言的以下要求:

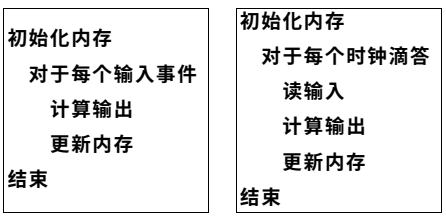
(1)并发性。同步式语言必须支持功能的并发性,而且它们必须依赖于以对用户友好的方式表达并发性的记号。因此,依赖于目标应用领域,这些语言应当提供目标领域工程师界所熟悉记号的方框图

收稿日期:2005-08-11

作者简介:苏运霖(1940-),男,广西博白人,教授,博士生导师,主要从事算法设计与分析、分布式系统与算法、人工智能等领域的研究工作。

(或叫数据流程图),或者层次式自动机,或者某种强制性的语法。随后的发展,还满足把这些不同风格的记号混合起来的需要。这就明确地要求,它们全都有相同的数学语义。

(2)同步性。同步式语言必须支持如图1所示的简单和频繁使用的实现模型。图1中所示的所有动作都被假定仅使用有限的存储和时间。



(a) (b)

图1 两类普通的同步执行模型

(a)事件驱动;(b)抽样驱动。

(3)简单性。同步式语言必须具有最简单形式的模型,以便能使形式推理成为可处理。特别地,两个进程的并行合成必须能干净利落地实现。

基于上述三点,同步式语言需要在维护简单的数学模型的同时,把同步和并发组合在一起。法国人创建的三种同步式语言 Estreal[2], Lustre[3] 和 Signal[4] 把时间分成为瞬间,这种模型在数学和工程中很流行。因此,决定了一个同步式程序是依照逐次的自动反应而往下运行。因此,为方便起见,可以令 R 表示所有可能的反应动作的集合,并以上标 w 表示不终止的迭代。而令 P 表示我们的程序,自然可以假设 P 不终止地迭代的反应动作的集合,于是就有了虚拟数学的语句 $P = R^w$。

控制工程的框图中,整个系统的第 n 个反应是对于每个基本模块的个别第 n 个反应的组合。例如,对于模块 i 来说,

$$X_n^i = f(X_{n-1}^i, V_n^i), \quad (1)$$

$$Y_n^i = g(X_{n-1}^i, V_n^i),$$

其中,V,X,Y 是(向量)输入、状态和输出,而它们的"组合"意味着模块 i 的某个输入或输入出被连接到模块 j 的某个输入,比如说

$$Y_n^j(k) = V_n^i(l), \text{或者 } Y_n^i(e), \quad (2)$$

其中,$Y_n^j(k)$ 表示在瞬间 n 时第 j 个模块的向量输出的第 k 个坐标。因此,整个的反应动作只不过是对于每个模块反应动作(1)的合取。(1)式中的第一个式子表示第 i 个模块在瞬间 n 的状态是由第 i 个模块在瞬间 $n-1$ 时的状态和它在瞬间 n 时的输入确定的。同样,它在瞬间 n 的输出是由在瞬间 $n-1$ 的状态和它在瞬间 n 时的输入确定的。

硬件上以同步数字逻辑方便地实现2个有限状态机器(Finite State Machine,简称FSM)的连接。一个无循环的(因此是功能的)逻辑模块计算输出和下一状态作为输入和当前状态的一个函数。并发运行2个有限状态机器,并使它们保持通讯的最自然的方式,需把一个有限状态机器的某些输出连到另一个有限状态机器的输入上,以及反过来。

因此,也就选定了表达同步式语言中并发组成的自然定义,即 $P_1 || P_2 = (R_1 \wedge R_2)^w$,其中 R_1 是对应于 P_1 的所有可能反应动作的集合;而 R_2 是对应于 P_2 的所有可能反应动作的集合;$\wedge$ 表示合取(即"与"运算),这个表达式的正确性应当是显然的。左边的并行意味着其中的反应动作的同时成立和运行。当然,这就要求对于每个组成都要形成反应动作的合取。而众所周知,这样的合取一般地将不是一个函数而是一个关系,或者等价地是一个限制。

对于在同步并发的合成下功能系统不封闭的问题,我们可以用4种以上的方式解决:

(1)微步骤。人们坚持认为,一个反应动作可通过定义它为一个基本的微步骤序列而保持动作。

(2)无循环。控制工程师们坚持认为,他们的框图不含零延迟的循环。这就确保系统的功能行为,该方法很适合于抽样驱动方法。

(3)惟一不动点。这一方法承认,每个反应是一个不动点方程的解,但同时系统总是按功能来运作,即每个反应是下列形式的一个确定函数:

{状态,输入}→{下一个状态,输出}。

(4)关系或限制。这个方法承认关系是个限制并且允许反应无解(即程序被封锁而无反应)。

到目前为止,同步式语言就是以上述原理创建起来,它们在处理嵌入式系统的同步、并发、分布式这些方面很有效,但在调度,验证,乃至更具智能特性的性能方面显得不足。因此,我们提出对它的智能化扩充。

2 智能化的同步式语言

从上述可以看出,同分布式语言虽然具有描述或实现分布式、并发以及同步的功能,但在描述和实现智能方面,似乎没有提供必要机制。因此,在上述同步式语言中,应该增加的设施包括:

(1)自适配性。系统必须顺应环境的变化,而使状态和交互模式连续地发生变化,我们以刨须器为例,它应能作如下判断:

如果{胡须浓密而且坚硬}则刀旋转加快;

如果{胡须稀疏}则刀速减慢；

如果{脸部不平}则刀速适中；

……

再以控制汽车在公路行驶的嵌入式软件为例，它应有如下的适配能力：

如果{在高速公路行驶且限速标志限定为110km/h且车辆较少}则置车速为$100km/h \leqslant V \leqslant 110km/h$。

如果{车辆密集，交通堵塞}限定$\leqslant 60km/h$。

如果{道路弯曲或空气能见度$\leqslant 500m$}限定车速为$\leqslant 40km/h$。

(2)自诊断性。自诊断性指系统能对自身的各种参数进行诊断，检验它们是否仍在正常工作范围内。假如出现某个参数异常，还应能进一步判断造成异常原因，即诊断出异常的性质，从而给出处理的决策。这表现为如果(传感器1送来信息异常∪传感器2送来信息异常∪...)则转入诊断检查。

诊断检查：

故障1：{现象：

结论：

处理方法：

}

故障2：{现象：

结论：

处理方法：

}

……

(3)预测性。预测性是一个学习过程，它允许系统通过对一系列状态的分析或经验积累，而对自身性能作出预测，判断系统在今后一段时间里性能的走向。通过使用传感器，智能体可以从它所处的每个状态感知到周围环境，并获知当前状态。理想的智能体希望知道对其决策有益的一切信息。因而语言应该提供实现这个特性的设施，从而产生最优决策。

(4)从故障或异常状态中的可恢复性。嵌入式系统不同于普通的系统，对于它而言，从故障或异常状态中恢复需要的代价更高。因此语言的设计也应为状态的恢复提供方便的条件。大体上说，智能化的同步式语言应该具有用于实现上述这些特性的设施。

3 智能化同步式语言的具体模型

同步式语言的智能化可以有以下3种实现模型：

(1)创建新的语言，使之兼具同步式和智能两方面的特征。

(2)在现有的同步式语言的基础上，增加智能化的设施。

(3)在现有的智能语言中，增加分布式，同步和并发等设施。

理想地说，这三者当中第一个方案是最好的。因为它从一开始就兼顾了智能和同步等特征。然而，也正因为一切都要从头做起，其工作的难度和强度很大，特别是如果要做成通用的智能化嵌入式软件的实现语言，其困难程度会更大。

现有的智能语言，公认的是Lisp和PROLOG。两者在风格上迥异，但在实现人工智能系统上都获得相当广泛的应用。但是，由于它们本身的限制，如果要在它们的基础上加上同步语言的特性。如果不是不可能，那也是极其困难的。

因此，第二条路是可行的，即在现有的同步式语言中加上智能化的设施。鉴于本文所提出的那些智能化设施基本上是基于一阶逻辑，因此，只要在现有的同步式语言扩充逻辑推理的设施就可以解决问题。但由于还包括如预测性或学习功能，也还应能描述马尔可夫过程这样的随机过程，因此，以PROLOG作为基础构建同步式语言的智能化设施是一条可行的路。

4 结束语

嵌入式系统是上一世纪最后十年才迅猛发展的新生事物。从一出现它就得到广泛的应用。惟其如此，它必然会向更高水平发展。智能化必将是嵌入式更高阶段，而智能化嵌入式软件的编程语言必定也是人们在把嵌入式系统推向前进时必须解决的问题。因此，本文的讨论将对解决这个问题提供借鉴。

参考文献：

[1] Benveniste A. The synchronous languages 12 years later[J]. Proc IEEE, 2003, 91(1): 64-83.

[2] Boussinot F, R de Simone. The esterel language[J]. Proc IEEE, 1991, 79(9): 1293-1304.

[3] Halbuachs N, Caspi P, Raymona P. The synchronous data flow programming language LUSTRE[J]. Proc IEEE, 1991, 79(9): 1305-1320.

[4] Le Guernic P, Gaupidrl P. Programming realtime applications with SIGNAL[J]. Proc IEEE, 1991, 79(9): 1321-1336.

(责任编辑：黎贞崇)