国防科技大学学报
JOURNAL OF NATIONAL UNIVERSITY OF DEFENSE TECHNOLOGY
第10卷第2期 1988年6月 Vol.10.No.2

实时仿真计算机系统的若干问题

张 晨 曦
（计算机科学系）

摘 要 本文比较了实时仿真计算机系统中配备A/D、D/A变换器的两种方案。分析了几种检索算法，并且提出了一种新的全并行检索方法——映象检索。文中论述了仿真计算任务的特点及其并行性的开发。本文在分析了AD-10的缺点之后指出，纯异构型仿真专用多处理机的局限性较大，不能充分开发仿真计算任务的并行性。较好的结构形式应是以同构型多处理机为核心。

关键词 实时仿真，函数生成，映象检索，并发性开发

1. 引 言

电子计算机在仿真中扮演着越来越重要的角色，成了仿真不可缺少的计算工具。许多仿真模型的求解只是在出现了计算机以后才成为可能。计算机的出现及其随后的不断发展，使仿真技术逐渐成为人们设计、分析、研究、测试各种系统的有力手段。

在仿真中应用的计算机有模拟计算机、混合计算机和数字计算机。随着数字技术的飞速发展，仿真越来越趋于数字化。许多模拟计算机已被数字计算机所取代。在仿真中可以采用一般的通用数字计算机。但是，为了提高性能价格比，为了满足大型动态连续系统实时仿真对计算速度的特殊要求，人们研制了仿真专用计算机（简称仿真机）。例如，AD-10、YH-F1等。

作为专用机，仿真机在许多方面与传统机器不同。它的语言、算法、系统结构、硬件等都是面向仿真的。本文根据连续系统实时仿真的特点从算法和系统结构的角度分析了仿真机设计中应考虑的几个问题。着重讨论了函数生成算法和仿真任务并行性开发等问题。

2. 配备A/D、D/A的两种方案

在连续系统实时仿真闭环中，计算机与实物的界面是A/D、D/A变换器。当这个

本文1986年7月16日收到

计算机是一般的通用机时，A/D、D/A是作为其外设配备的。这样构成的闭环系统如图1所示。但是，若这个计算机是一个仿真专用机，则它一般采用主——辅机结构，由一台小型通用机（作主机）和一个仿真专用处理机（作辅机）构成。由于这个系统中有多个处理机，因此在设计该系统时，就存在如何配置A/D、D/A的问题。可有两种方案，分别如图2(a)和图2(b)所示。

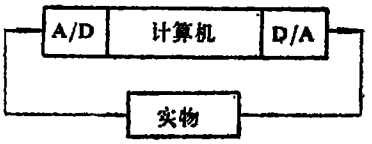
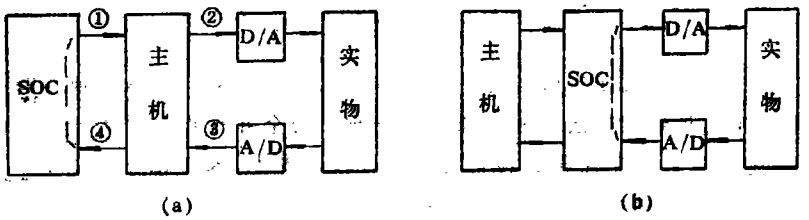


图 1

图 2(a) 所示方案的优点是：主机也是闭环中的一部分，它也参与了实时仿真任务的计算和控制（至少要控制数据传输）。仿真任务可以在SOC和主机之间进行合理的分



SOC——仿真专用处理机

图 2 A/D、D/A的两种配备方案

配。具有较大的灵活性。而且可以利用主机配备的A/D、D/A以及控制它们的软、硬件。SOC因此变得比较简单和易于实现，缩短了研制周期。其缺点是数据传送频繁。每一帧中至少要进行四次数据传送（图2(a)中的①、②、③、④）。这对主机总线的数据传送率要求较高。如果SOC的速度不是非常快，在每一帧中它求解模型一步所需时间比数据传送所花时间大得多（如图3所示），则采用这种结构是合理的。这时帧时间主要花在SOC的计算上。减少数据传送时间并不能显著地减少帧时间。PDP-SP系统采用了这种结构[5]。

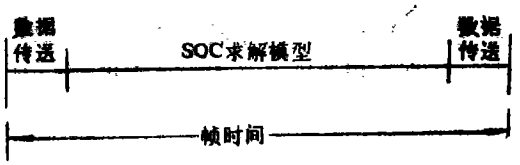


图 3

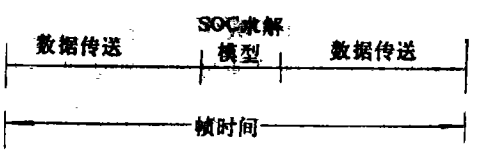


图 4

然而，在帧时间大部分都花在数据传送的情况下(如图4所示)，就应采用图2(b)的方案。这样能大大缩短帧时间。AD-10采用了这种方案[1]。这一方案的特点是：

1. 在实时仿真的过程中，A/D、SOC、D/A和实物等四部分构成闭环。主机并不参与实时仿真任务的计算。

2. A/D、D/A由SOC控制。SOC中有相应的控制硬件和软件，这就增加了SOC的复杂程度。

3. 每一帧中数据传送次数较少。而且，由于传输总线是SOC中的，在设计SOC

时，可把其传输率设计成很快，使数据传送时间很短。

3. 函数生成算法及相应的硬件支持

在仿真中，经常要遇到一些表格函数。当对于给定的自变量值，求这些函数的值时，就要进行检索（查表）和插值计算来求其近似值。这就是所谓的函数生成问题。实际上，在实时仿真应用中，即使是对解析函数，有时也不按其函数式进行计算，而是预先在其自变量的取值范围内进行造表，把它变为表格函数放入内存。在实时仿真中进行函数生成，快速求出其函数值。

多变量函数生成在连续系统仿真，特别是实时仿真中有着重要的作用。在航空与航天仿真，多变量函数生成可能占总计算量的一半。它的速度直接影响到帧时间的大小。因此，如何从算法和硬件上对函数生成提供支持是实时仿真计算机系统设计中的一个重要问题。

函数生成分两步进行：检索和插值计算。下面分别讨论。

3.1 检索

函数表是以函数在其自变量的一些离散点（称断点）上的值给出的。设断点为x_1，x_2，…，x_n，把$\Delta x_i=x_{i+1}-x_i$叫做造表步长。检索的任务就是对于给定的自变量x（设为定点值），要找出满足$x_i\leqslant x<x_{i+1}$的i．对于各Δx_i的不同取值情况，我们可以采用不同的检索算法。

3.1.1 等步长造表($\Delta x_1=\Delta x_2=\cdots=\Delta x_{n-1}$)

设x是一个m位二进制定点数。断点数$n=2^k(k\leqslant m)$，则截取x的前k位便得到了i。而剩下的$m-k$位就是$\beta=(x-x_i)/(x_{i+1}-x_i)$.

这种方法的优点是检索速度快，不必存放断点值，而且不必计算线性插值公式中的因子β.其缺点是各区间的插值精度不均匀。为了保证在各区间内的插值精度满足要求，就得按最小步长造表。这就使得函数表需占用较多的存贮空间。此外，实验数据也难以完全按等间隔给出。所以，这种方法的适应面较窄。

3.1.2 分区间变步长造表[5]

这种方案是我们在设计一个小型实时仿真机系统的过程中提出的。它允许用户把其造表区间分成几个（设为r个）子区间。虽然在每一个子区间内必须采用固定步长，但不同的子区间可采用不同的步长。这种方案既保持了固定步长的优点，又给用户提供了更多的灵活性。它是介于等步长造表与变步长造表之间的一种方法。有较广的适应范围。

下面我们来推导它的检索算法。

假设：(1) $r=8$；

(2) 造表子区间的端点为：

$$x_{(1)}=a,\ x_{(2)},\ \cdots,\ x_{(9)}=b$$

(3) 第j个子区间$(x_{(j)},\ x_{(j+1)}](1\leqslant j\leqslant 8)$的造表步长为$\Delta x_{(j)}$，所包含的断点数为$n_j$。

若 x 落在第 j 个造表子区间，则

$$i=\sum_{k=0}^{j-1}n_k+\left\lfloor\frac{x-x_{(j)}}{\Delta x_{(j)}}\right\rfloor=ADR_j+\left\lfloor(x-x_{(j)})\cdot\frac{1}{\Delta x_{(j)}}\right\rfloor$$

其中 $ADR_j=\sum_{k=0}^{j-1}n_k$，$n_0=1$。

$$\beta=\frac{x-x_i}{x_{i+1}-x_i}=\frac{x-x_i}{\Delta x_{(j)}}=\frac{x-x_{(j)}}{\Delta x_{(j)}}-\frac{x_i-x_{(j)}}{\Delta x_{(j)}}$$

$$=(x-x_{(j)})\cdot\frac{1}{\Delta x_{(j)}}-\left\lfloor(x-x_{(j)})\cdot\frac{1}{\Delta x_{(j)}}\right\rfloor$$

即 β 是 $(x-x_{(j)})\cdot\frac{1}{\Delta x_{(j)}}$ 的小数部分。

该检索算法的流程图如图5所示。对于任意一个变量 x，检索所需的信息仅是 $x_{(1)}$，$x_{(2)}$，…，$x_{(8)}$；$1/\Delta x_{(1)}$，$1/\Delta x_{(2)}$，…，$1/\Delta x_{(8)}$；ADR_1，ADR_2，…，ADR_8。数量不多，可存放在专用的高速存贮器中，以提高检索速度。此外，若设置多个比较器实现 x 与 $x_{(1)}$，$x_{(2)}$，…，$x_{(9)}$ 的并行比较，那么就能快速实现上述检索算法。其计算量（包括 β 的计算）仅是二个加减、一个乘法和一个取整操作。

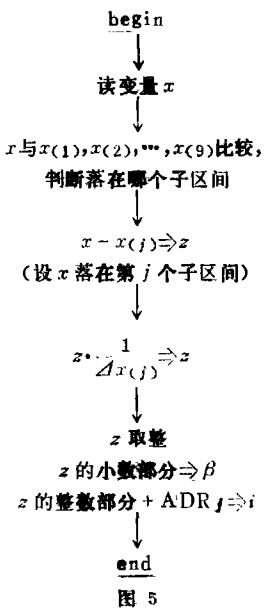


图 5

3.1.3 变步长造表

这是最灵活的造表方法，它允许用户以任意的步长造表。因此，对任何表格都是适用的。然而，这种表格的检索比较费时间。需从硬件上提供支持。

1. t 元检索

通常，人们采用二元检索（$t=2$）[1]。所需的检索步数为：$N_1=\lceil\log_2(n-1)\rceil$。

为了缩短检索时间，可取较大的 t。如取 $t=4$，则所需的检索步数为：$N_2=\lceil\log_4(n-1)\rceil=\left\lceil\frac{1}{2}\log_2(n-1)\right\rceil$。即检索速度提高了约一倍。

实现 t 元检索所需的硬件支持包括：

(1) $(t-1)$ 个并行比较器。

(2) 存放断点值的存贮器应是单体多字存贮器。一个存贮周期能并行读出 $(t-1)$ 个字。

2. 映象检索

最快的检索算法显然是 n 元检索。即把 x 与所有 n 个断点值并行比较，一步确定出 i。这是一种全并行的方法。这种方法可用相联存贮器来实现。相联存贮器是按内容访

问的存贮器。它的每个单元都具有比较功能。但由于目前相联存贮器片子容量较小、价格较高，用它来实现全并行检索还存在一定的困难。所以，必须另寻途径。我们提出了一种新的方法——映象检索。它是利用一般的按地址访问的存贮器来实现全并行检索。

我们知道，检索最基本的任务就是求 i。对于任意给定的 x，检索的结果就是满足 $x_i \leqslant x < x_{i+1}$ 的 i。这可以看成是一种映象关系：

$$f:\ X \to I$$

其中 $X=\{x$ 所有可能的取值$\}$，$I=\{1, 2, \cdots, n\}$。x 是以离散量给出的，故 X 是一个有限集合。若 x 是一个 m 位二进制数，则 X 中元素的个数最多为 2^m 个。我们可以用一个容量为 2^m 字 $\times m$ 位的存贮器来实现上述映射。如图6所示。其中 $i_{x'}$ 是对应于 x 的检索结果（x' 是当把 x 看成地址时的整数值）。各 $i_l (l=0,1,\cdots,2^m-1)$ 是在进行实时仿真前预先计算出并装入该存贮器的。在实时仿真中只要以 x 作为地址读出相应单元的内容，便得到了检索结果。

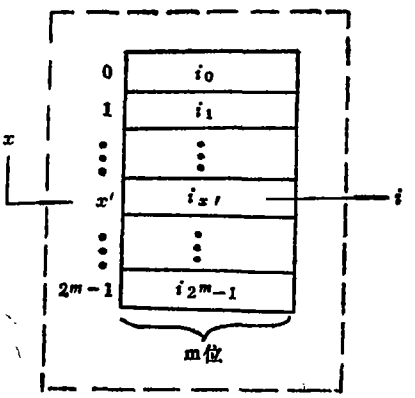


图 6　映象检索的实现

在 x 是一个从A/D变换器取得的实时输入变量的情况下，x 的位数 m 由A/D变换器的精度确定，目前一般为12位。因此，为实现映象，每一变量只需 4K × 12 位的存贮空间。这是不难实现的。目前一个存贮器片子的容量就已达256K位、512K位。即使是对于较大的 m，随着器件技术的发展，也是能够实现映象检索的。

3.2　插值公式的实时形式

在实时仿真中，通常采用的插值算法是线性插值。这是因为线性插值有其固有的简单性，计算速度快。其它更高次的插值方法，如三点插值、四点插值等由于其计算公式的一般形式[4]复杂，较少被考虑。但是，通过把这些公式变形，可推导出其另一种形式：

1．线性插值：

$$\left.\begin{aligned} &\text{实时部分：}\ f_{(2)}(x)=A_2x+B_2 \\ &\text{非实时部分：}\ A_2=(f_{i+1}-f_i)/(x_{i+1}-x_i) \\ &\qquad\qquad B_2=f_i-x_iA_2 \end{aligned}\right\} \tag{1}$$

2．三点插值：

$$\left.\begin{aligned} &\text{实时部分：}\ f_{(3)}(x)=A_3x^2+B_3x+C_3 \\ &\text{非实时部分：}\ \text{（略）} \end{aligned}\right\} \tag{2}$$

3．四点插值：

$$\left.\begin{aligned} &\text{实时部分：}\ f_{(4)}(x)=A_4x^3+B_4x^2+C_4x+D_4 \\ &\text{非实时部分：}\ \text{（略）} \end{aligned}\right\} \tag{3}$$

与此类似，对于其它的插值公式，不管它的形式怎样复杂，都可分离出实时部分和

非实时部分。可以看出，次数相同的插值公式，其实时部分的形式都一样。所不同的仅是非实时部分。

上述各式中的非实时部分可在实时仿真准备工作阶段预先计算好。这样，插值用的函数表中存放的值就不是通常的函数值 $f_1, f_2, \cdots, f_n$，而是相应的中间结果，亦即是实时部分中的多项式系数。在实时仿真中，只要进行一个简单多项式的计算即可。

上述这种把插值公式的计算分离成两部分的方法实际上增加了总的计算量（例如，对于四点插值来说，计算量增加了将近一倍。）换来的好处是减少了实时计算过程中的计算量，大量的计算可在准备工作阶段完成。由此可见，插值公式的一般形式更适合于非实时计算。而（1）、（2）、（3）式则更适于实时计算。故称之为插值公式的实时形式。

4. 仿真计算任务并行性的开发

与其它的计算任务一样，仿真计算任务中也存在着固有的并行性。提高仿真机速度的重要措施之一是提高并行性。本节分析仿真计算任务的特点及其并行性的开发。

4.1 并行处理系统简介

提高计算机系统的并行性所遵循的技术途径有：时间重叠、资源重复和资源共享。在众多的并行处理系统中，主要的有并行处理机、流水线处理机和多处理机。流水线处理机和并行处理机都是单指令流多数据流（SIMD）机器，特别适合于向量处理，但不能实现多个任务的并行执行。多处理机是多指令流多数据流（MIMD）机器，能实现任务级并行。

多处理机有同构型和异构型之分。同构型多处理机由多个同类型、至少担负同等功能的处理机组成。这些处理机同时地处理同一程序中能并发执行的多个任务。异构型多处理机由多个不同类型、至少担负不同功能的处理机构成。这些处理机按时间重叠原理依次完成各自对每一个数据集的处理任务。异构型多处理机也可以看成是多计算机循功能专用化这一途径发展的结果[3]。

近年来，器件技术在飞速发展，片子集成度不断提高，价格不断下降。这使得按资源重复这一途径大幅度提高仿真机的并行性成为可能。但是，目前大多数通用多处理机系统中的处理机个数并不多。其主要原因是在通用的系统中，程序并行性的识别与开发、并行任务派生、进程同步、任务调度、机间通讯等问题比较复杂。但是，就面向某一类型计算问题的专用多处理机系统来说，这些问题容易解决。所以，设计处理机个数很多的专用处理机系统是可能的。

4.2 连续系统仿真计算任务的特点及其并行性的开发

连续系统仿真的主要计算任务是求解常微分方程组，包括右函数计算和积分两大部分。积分计算一般是向量运算，并行处理机或流水线处理机已能很好地发挥其并行性。然而，右函数的形式一般是不规整的。各方程的右函数互不相同，其向量化比较困难。因而在SIMD计算机上的并行度不高。右函数计算往往占解常微分方程组计算量的大部分。为了提高速度，就必须使这部分计算也能高度并行。较好的办法是采用多处理机。

异构型多处理机是一种可能的形式。AD-10就属于这类机器。在AD-10中，按不同的功能设置了多个专用处理机。这在很大程度上提高了计算速度。然而，AD-10所能开发的并行性受限制较大。从系统结构的角度来看，AD-10的主要缺点是只依靠功能专用化的思想来提高并行性。原因有三个：

1．按功能专用化的原则来实现仿真计算任务的任务级并行是困难的。在宏观上，仿真中一帧的计算任务，特别是右函数计算，并不能有效地按计算功能划分成多个独立性较好的模块。所以，AD-10是对于单个的计算，特别是结构化计算，如数值积分、函数生成和坐标变换等，分别独立地开发其并行性。

2．按功能专用化设计的多处理机中，处理机个数不可能很多。这种结构能开发的并行性受限制较多。

3．AD-10中各处理机的指令相互不同，而且很复杂。各处理机之间的同步也比较复杂。所以，用汇编编写能使各个处理机有效地并行执行的仿真程序非常困难，软件研制的工作量很大。

4．AD-10由于采用了总线结构，在硬件上有良好的可扩性。但是，若增设处理机，不重写软件模块库中有关的部分是难以发挥其效用的。从这一点来说，AD-10的扩展性并不好。

我们认为，从发展的角度来看，异构型多处理机（不包括以同构型多处理机为核心，并辅助一些专用处理机的异构型多机综合系统）的结构形式局限性较大，不能充分开发仿真计算任务的并行性。

同构型多处理机是一种很好的结构形式。在仿真过程的每帧计算中，各微分方程的独立性较大，能有效地划分成多个子任务在多个处理机上并行计算。因此，仿真专用同构型多处理机是一个重要的研究方向。在这样的专用系统中，并行任务分配、机间通讯、同步等问题都比较容易解决。其软、硬件的实现都不太困难。[6]中介绍有关这种系统的一个方案设计。

5. 结束语

本文论述了仿真计算机系统中的A/D、D/A变换器的配制方案、检索和插值、并行性开发等几个主要问题。

分区间变步长造表检索方案是一种值得考虑的方案。它具有检索速度快、检索所需的信息量少的优点。但其应用范围受到一定的限制。改进的办法是增加子区间个数，并采用分层检索。最灵活且速度最快的检索算法是映象检索。它的唯一缺点是所需的存贮空间较大，但随着器件技术的发展，这个缺点将逐渐变小。

同构型仿真专用多处理机能较好地开发仿真任务的并行性，具有很大的速度潜力。而且通用性较强，是一个值得研究的方向。

吴连伟老师和周兴铭老师审阅了本文，并给予了指导，在此向他们表示感谢。

参 考 文 献

[1] AD-10参考资料
[2] 全国第五届系统仿真学术会议论文集，1985年
[3] 苏东庄，计算机系统结构，西北电讯工程学院出版社，1984年
[4] 武汉大学，山东大学计算数学教研室，计算方法，人民出版社，1979年
[5] 张晨曦，一个小型实时仿真计算机系统PDP-SP，硕士学位论文，国防科大计算机科学系，1984年
[6] 张晨曦，连续系统仿真专用多处理机的探讨，全国第六届系统仿真学术会议论文集，1987年

On Some Problems in the Design of Real-Time Simulation-Oriented Computer Systems

Zhang Chenxi

Abstract

In the paper, two schemes for attaching A/D's、D/A's to real-time simulation-oriented computer systems are discussed first. Then a number of searching algorithms are analysed, and a new searching method——mapping search is presented. In it are also discussed the characteristics of computing tasks in continuous system simulations and the exploitation of parallelism of these ones. After analysing the defaults of the architecture of AD-10, this paper points out that pure heterogeneous multiprocessor systems have many constraints on the exploitation of parallelism of computing tasks in continuous system simulations. Better architecture should be based on homogeneous multiprocessor.

KEY WORDS Real-time simulation, Function generation, Mapping search, Exploitation of parallelism