计算机科学1994 Vol. 21№. 3

MCP多处理机系统研究与设计

王能忠　王宽全　张为群
（西南师范大学计算机科学系　重庆630715）

摘要　This paper develops a multiprocessor system for modern Chinese processing (MCP). It designed as asymmetric master/slave configuration, which consist of 33 microprocessor. Also the parallel MCP algorithm was designed on the system to realizing Chinese parallel parsing.

关键词　Multiprocessor system, Modern Chinese processing, Parallel parsing.

一、引言

人工智能(AI)中自然语言处理(理解)的研究在八十年代进入了并行处理阶段，国外已有众多研究成果[1]。进入九十年代，其研究水平已相当高，其中比较成功和有名的有美国卡内基·梅隆大学与南加利福尼亚大学联合研制的SNAP(语义网络阵列处理机)，它由160个TMS320C32DSP芯片组成[2]，用于机器翻译系统；美国卡内基·梅隆大学与日本NEC公司联合研制的大规模并行联想处理机IXM2，它由64个处理机组成[3]，用于大规模并行句法分析。笔者1991年在澳大利亚墨尔本大学成功地用Encore多处理机系统进行了汉语并行句法分析[4]，最近我们承担了"汉藏英机器翻译系统"课题中的部分研究任务，进行了适用于MCP(现代汉语处理)的一种多处理机系统设计。

二、多处理机系统中的几个概念

2.1　多计算机系统

多计算机系统是指把多个各自独立的处理器或者计算机组织在一个系统中，在集中或分布控制下协同操作的并行处理系统。系统中各个处理器或计算机可以同步地工作，也可以异步地工作，可执行相同的操作，也可以执行不同的操作，甚至完成不同的任务。

多计算机系统的基本互连结构，按照多计算机间实现通讯联系和交换数据的方式不同，大体上可以区分为两大类：共享存储；消息传递，如图1所示。

2.2　并行处理系统的性能

在理想情况下，如果某作业可均匀地分成p个可并行执行的任务，由p个处理器并行处理，则所需处理时间为t_p，而该作业由单个处理器系统完成时，共需时间T_s，于是$t_p=T_s/p$；并行系统的加速因子为：$S_p=T_s/t_p=p$。

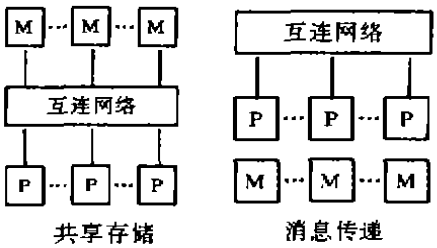

图1　多计算机系统的两种基本互连结构

在实际问题求解过程中，总是存在着不可并行化成份，于是有下式成立：

$$T_o=t_s+t_p'/p+NO_s$$

式中：t_s：串行部分处理时间，t_p'：可并行处理部分处理时间，p：处理器数，N：进入并行处理状态次数，O_s：每次进入/退出并行处理状态所必须的最小开销。

此时有：$S_p=T_s/T_o$，令$t_p'=T_s-t_s$，可得：

$$S_p=p/(1+(p-1)t_s/T_s+pNO_s/T_s)$$

显然　$S_p<p/(1+(p-1)t_s/T_s)$

设$p=32$，$t_s/T_s=0.02$，即串行成分所需时间只占2%，此时加速比$S_p<19.6$[5]。

三、用于MCP的多处理机系统设计

3.1　CPU选择

本系统设计为非对称的主从结构，共由33台微处理机构成，即主处理机由一台功能完备、外设齐全

的486系统组成(主要性能为80486Dx-66,8M 内存,双软,外存340M/680M,高速缓存256K),从处理机选择为32台386系统组成(主要性能为80386DX—40,4M 内存,单软,高速缓存64KB),如图2所示。

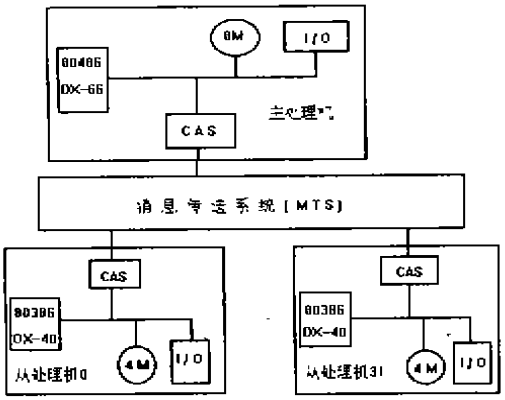

图2 主从结构松耦合非对称多处理机系统

主处理机负责作主要处理任务,管理程序只驻留在主处理机中,它负责管理多机系统中所有其它32台从处理机的状态;给所有从处理机分配工作;负责系统的调度,并执行系统的输入/输出,主从结构具有以下特点[2]:·管理程序一直驻留在主处理机中,·只有主处理机执行访问表,不存在存取冲突和阻塞。·硬件和软件结构比较简单。·主处理机采用486,能高速执行管理功能。·主从结构采用非对称系统,可以降低系统造价。·缺点是当主处理机故障时,将造成系统瘫痪。

3.2 耦合系统

本系统设计为一个松耦合系统,每个处理机都有一个大容量的局部存储器(4M),用于存储经常访问的指令和数据,以减少访问冲突,不同处理机间的进程通过一个信息传送系统(MTS)进行通信(图2)。这种系统的耦合度较之紧耦合系统松,而且给算法设计和实现带来方便。

3.3 互连方式

多处理机系统的互连网络的形式有总线、交叉开关和多端口存储器、以及多级互连网络等四种。在本系统中我们采用相对简单的总线结构,这种结构价格便宜,易于建造,并且系统扩充和修改容易。当然它也有可靠性差,系统能力受总线传输率限制、随着处理机数的增加,访问总线的冲突会降低系统效率等缺点。为在一定程度上克服上述缺点,本系统最终采用双总线结构,以提高系统的传输效率。对总线的使用,系统采用先来先服务(FCFS)算法。

在并行处理计算机中,互连网络是一个重要组成部分,网络的拓扑结构决定处理机间的通信结构,处理机和存储器以及 I/O 部分之间的数据通路,直接影响着系统的信息传输速度、容错能力、路径算法的复杂性和整个并行处理系统的效率。为配合并行分析算法的设计,本系统最终选择了准三维的"线性阵列+环形"拓扑结构(图3)。

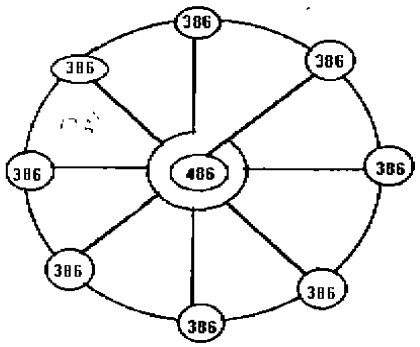

图3 准三维的拓扑结构

线性阵列沟通了主处理机486和32个从处理机386之间的通路,并采用 FCFS 算法实现。通过这一路径实现主从处理机的系统调度、控制、任务分配及信息传递。各从处理机之间则构成环形网络,实质上它是一个允许信息双向流动的最邻近网络,如图4所示,这种网络结构简单,造价低廉[6]。

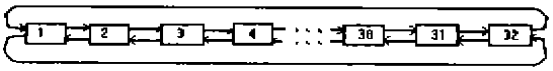

图4 最邻近网络

四、MCP 的并行算法设计

提高并行处理效率的关键之一是设计高效实用的并行算法,并行算法是一些可同时执行的进程的集合,这些进程相互作用和协调,以完成对一个给定问题的求解。

并行算法与并行处理机的关系远比串行算法与串行处理机的关系密切。并行算法直接依赖于并行处理计算机的结构,不同类型的并行处理机,在其上所使用的并行算法也不尽相同,并且所要解决的具

体问题也不完全一样。

在这里我们仅以汉语并行句法分析为例，本系统的硬件结构支持着与它相适应的并行句法分析算法，有关这一算法的研究已经完成并实现[4-7]。

我们采用了一种既有高并行性又有高可靠性的方法。一般来说，这种方法如下：假如输入的句子由 n 个词($W_1, W_2, \cdots, W_n$)组成；语法分析产生由 j 个部件构成的输出结构($C_1, C_2, \cdots, C_j$)；它通过将输入句子送给并行操作的一系列处理机而达到目的。这种方法如图5所示。

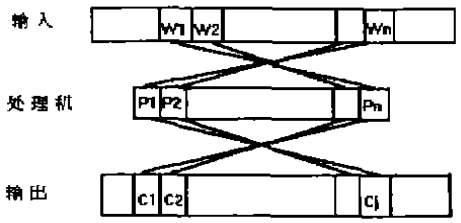

图5　多处理机结构分析模型

这种方法假设输入的句子已经分成了带有词类标记的词。输入串中第 i 个词以串的形式指派给第 i 个处理器。所有的处理器是完全一样的，但又是独立地异步地进行共同操作，且运行相同的软件。第 i 个处理器的目标是发现第 i 个词开始的所有语法结构。当每个处理器只有一个词"指派"给它时，它可以自由地考察句子中所有的其它词。因此，系统假设全句的词被复制到每一个处理器的存贮器中。这种结构的目的是对给定的句子使更多的处理机并行执行。

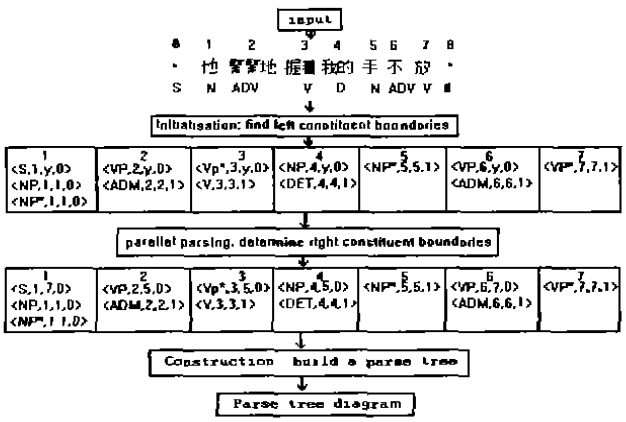

图6　并行句法分析

下面，我们通过一个例句的语法分析过程来阐述语法分析的方法(图6)。第一阶段(输入)得到一个句子，该句子已经被分成了带词类标记(一类终止符号)的词。下一阶段(初始化)产生一个串，在串中句子成份(如NP，VP)的左边界等已确定。再下一阶段(并行分析)，句子成分的右边界被确定。在最后阶段(构造)，产生一个分析树(图7)。

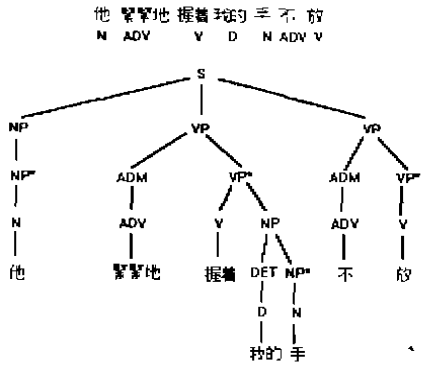

图7　语法分析树

下面图8给出了这些阶段如何联结的详细描述。

```
begin
  INPUT: Split the input string into words; determine morphological type of each word;
         assign positional number to each word
  INIT: Examine left constituent boundary (LCB) table; produce (C, x, y, l) quadruples for
        each processor
  AMP: Analyze and process ambiguous LCB's
  CHECK: Determine state of each processor
  repeat
    VALUE: Find right constituent boundaries for each processor
    CHECK: (Re) Determine state of each processor until First processor is in final state
  OUTPUT: Build and output parse tree
end
```

图8　并行语法分析算法

根据并行模型及与单处理机模型的比较，我们采用的这种方法的处理速度增加的比率为：

$$S_p(n) = O(2n)/O((\log(n))^4)$$

其中 n 是处理器个数。