第 36 卷 第 20 期 计 算 机 工 程 2010 年 10 月
Vol.36 No.20 Computer Engineering October 2010
·开发研究与设计技术· 文章编号：1000—3428(2010)20—0250—03 文献标识码：A 中图分类号：TP301.6

交互式并行化系统 ZIPS

孙煦雪，李 莹，袁新宇，徐印成
(浙江大学计算机科学与技术学院，杭州 310027)

摘 要：目前全自动并行化方法在并行化能力和应用范围上存在较大限制，而交互式并行化方法能弥补全自动并行化系统的不足。基于此，提出一种交互式并行化方法及其系统 ZIPS，描述系统的并行化处理机制，即采用一种计算类型驱动的并行化算法，并以其作为理论基础，针对 2 类不同计算，利用强大的交互功能获取相关程序信息，并结合自动并行化技术进行源到源的变换。实验表明，该交互式并行化方法能够获得较好的性能。

关键词：交互式；并行化；程序变换

ZIPS: Interactive Parallelization System

SUN Xu-xue, LI Ying, YUAN Xin-yu, XU Yin-cheng
(College of Computer Science and Technology, Zhejiang University, Hangzhou 310027, China)

【Abstract】 Automatic parallelization has limitations on the parallelizing capabilities and the scope of applications, while the interactive parallelization method makes up the deficiencies. This paper presents an interactive parallelization system named ZJU Interactive Parallelizing System(ZIPS), and describes the mechanism in detail. The system uses a computation-driven algorithm as the basis, obtains the program information through the powerful interaction interface, combines the technology of automatic parallelization, and finishes the source to source transformation for two different computations. Experiments show that the method can achieve good performance.

【Key words】 interaction; parallelization; program transformation

1 概述

随着片上多核处理器时代的到来，如何使应用程序在多核体系结构下获得好的性能，已成为一个研究热点。加快应用程序的执行速度的方法分为显式和隐式，而其中依赖于并行编译器进行串行程序并行化的隐式方法，是并行处理领域研究的热点问题，也是高性能计算领域需要解决的问题之一。

串行程序并行化分为全自动并行化和交互式并行化 2 种模式。全自动并行化系统，如 Polaris[1]、SUIF[2]、AFT[3]等，可以自动发现程序中可并行的部分来生成并行代码，具有一定的并行化能力。但由于其使用的并行化算法还不能有效地处理复杂应用程序，导致应用范围只能局限于科学计算领域，不能适应于如桌面应用、多媒体及服务器等更广泛的领域。交互式并行化工具则通过给用户提供程序中的有效信息，如 GPE[4]、ParaScope 系统[5]、NARAVIEW[6]和 PathScale-ICI[7]等均是交互式的并行化系统。这些交互式并行化系统，在尽可能采取自动并行化技术的同时，利用人工的能力来提高并行化效果，在部分程度上弥补了全自动并行化系统的不足。但上述交互式并行工具普遍都没有采用最新的自动并行化技术，而且其交互方法以及与用户间协作的紧密性有待研究提高，因此，它们的并行化效果在现行多核体系结构下仍然不够理想。

本文描述了一种交互式并行化编译系统 ZIPS(ZJU Interactive Parallelizing System)的结构及相关机制。

2 系统结构与特点

2.1 系统结构

ZIPS 系统的目的在于提供友好的用户与编译器协作机制，结合最新自动并行化技术来实现串行应用程序源到源的转换，使转换后的程序在多核体系结构下获得好的性能。

ZIPS 系统采用一种计算类型驱动的并行化算法，并以此算法作为理论基础，针对由规则数据流组成的计算和诸如 SPECCPU2000 Compress 等的特定计算，提供了一种基于 Eclipse 编译器框架的交互式并行化方法。

ZIPS 系统包括交互并行化插件和交互并行化引擎两大模块，如图 1 所示。

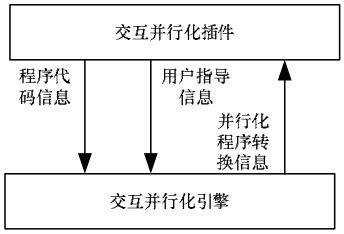


图 1 ZIPS 系统体系架构

交互并行化插件模块是基于 Eclipse 的交互视图插件，提供了友好的可视化环境及用户交互功能。用户可以通过编辑器、性能参数配置页以及用户经验窗口对并行化工作进行指导。另外，交互并行化插件还提供了可视化浏览环境，包括可以查看并行化效果的可视化性能分析器，以及可以查看程序信息图的多图浏览器。

基金项目：国家自然科学基金资助项目(60873045)
作者简介：孙煦雪(1984－)，女，硕士研究生，主研方向：并行计算，编译优化；李 莹，副教授；袁新宇，博士研究生；徐印成，硕士研究生
收稿日期：2010-04-02 **E-mail：**cnliying@zju.edu.cn

交互并行化引擎包含自动并行化模块、交互式模块、程序变换模块以及并行代码生成模块，实现了源到源的并行化操作。引擎的工作机制如图 2 所示，自动并行化模块在接收程序信息后，进行过程内数据流分析、过程间分析以及数据依赖分析。程序变换模块接收自动并行化模块分析的结果，以及用户提供的知识信息，对程序块进行理解并针对不同计算进行并行化变换。并行代码生成模块对程序变换模块中产生的相关并行化变换进行源代码更改，生成源程序代码与 OpenMP 指导语句相结合的并行化代码。

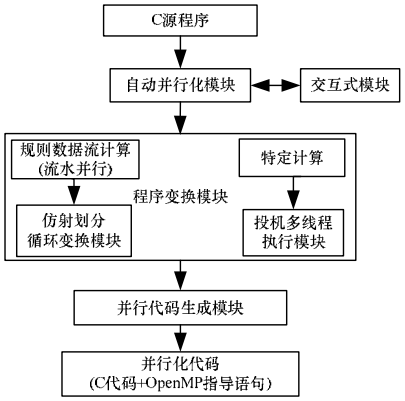


图 2　交互并行化引擎工作机制

2.2　系统特点

系统特点为了更清晰地阐明 ZIPS 系统的特性，将 ZIPS 系统与清华的交互式 Fortran77 并行化系统 TIPS[8]进行比较，结果如表 1 所示。

表 1　ZIPS 系统与 TIPS 系统的比较结果

对比项	ZIPS	TIPS
基底框架	基于 Eclipse 框架，提供了友好的可视化环境和强大的用户交互功能，具有较好的可移植性和可扩展性	基于 Polaris
语言	目前支持 C 代码	支持 Fortran
代码生成	采用源到源的变换，生成 OpenMP 指导语句的并行代码，具有通用性，可针对不同目标机器体系结构不同平台获得较好的并行化效果	并行 Fortran
自动并行化	在依赖分析基础上，使用改进的基于广义 GCD 测试的仿射划分算法进行自动并行化	过程间分析技术，扩展 Range 测试，用户信息确定循环是否并行化
投机执行	支持投机多线程并行化；支持不同类型计算，提供相应的并行化方案；支持多种粒度的并行性	N/A
交互功能	显示程序图模型可视化信息；用户可自定义配置性能参数；具备可视化性能分析器；代码编辑器可让用户选择并行化代码，并直观显示并行化结果；用户经验窗口用于指导特定计算的投机多线程并行化过程	显示并行化结果，向用户询问程序语义信息，解释并行化结果的原因

3　关键技术

3.1　依赖分析与仿射划分

SUIF[2]中提出的仿射划分框架归纳了很多有用的程序变换，其仿射划分算法在挖掘循环级并行性方面被证明是很成功的。经研究和分析，仿射划分算法需要的一个矩阵可以根据广义GCD测试得到的2个矩阵直接构造出来。基于此结果，ZIPS系统提出并实现了基于广义GCD测试的仿射划分算法，其伪代码表示如下：

```
Begin
对于程序中所有数据存取对{
if(广义 GCD 测试==存在依赖) {
保存广义 GCD 测试的中间结果矩阵;
if(通用 ILP 测试==存在依赖) {
根据中间结果矩阵计算出仿射划分算法中需要的矩阵;
计算处理器的仿射划分;}
}
}
综合所有可行的处理器仿射划分得到最终的处理器仿射划分
End
```

3.2　基于交互信息的投机并行化

针对静态时刻依赖无法解析或需运行时刻信息的特点计算，如何确定合适的执行模型以及建立对应的开销模型和性能评估，使用投机多线程来获取理想的性能，这是 ZIPS 系统要研究解决的重要问题之一。

ZIPS 系统中使用的方法，是结合交互式和投机并行化、利用与用户交互获取的信息来确定投机并行化的执行模型，建立评价模型，“ 按需 ”并行化来达到性能要求。使用软硬件相结合来实现投机并行化系统，其框架结构如图 3 所示。ZIPS 系统中投机并行化的研究执行模型使用硬件来检测内存依赖冲突并对投机状态和数据进行缓冲，维护内存的一致性，默认采用循环展开和迭代聚合的程序变换技术以及值预测机制，将线程抽取创建和线程执行作为可变量。编程人员通过 ZIPS 系统的可视化交互工具输入相关机制设计和策略选择信息。ZIPS 系统将根据与用户交互获取的信息来选择线程抽取创建和执行的策略，形成“ 按需 ”执行模型，进行投机并行化。

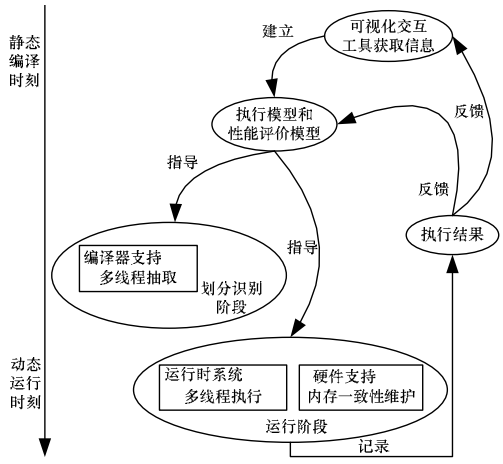


图 3　基于用户交互信息的投机多线程并行化系统

3.3　交互功能

ZIPS 系统的交互技术通过程序信息图浏览器、用户自定义性能参数配置页、可视化性能分析器、用户经验窗口功能以及基于 Eclipse 插件技术的编辑重写功能，形象地向用户展示了经自动化分析获得的程序块信息，并直观地显示了并行化变换后的代码效果，同时系统友好的交互功能使得用户提供的信息可以被并行化引擎有效利用。鉴于篇幅限制，本文仅对程序信息图浏览器作简要描述。

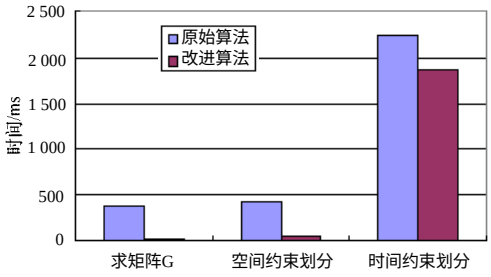
ZIPS 系统中对程序依赖图、程序流图以及程序任务图进行了建模，提供了这 3 种程序信息图的可视化浏览。文献[9]对 3 种图模型进行了形式化定义和描述。依赖图反映了程序的依赖结构，对于循环的依赖分析及整个循环转换，是一个有价值的工具；流图能够有效表示迭代计算，基于流图表示发展了许多针对迭代程序的并行化技术，如软件流水等；任

务图可以反映迭代和非迭代计算，通常用于粗粒度非迭代计算。

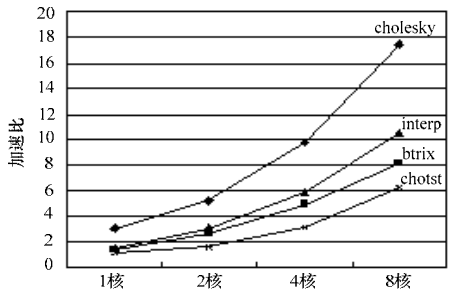
ZIPS 系统中的程序信息图浏览器直观地向用户提供了程序结构和内在信息，便于用户理解程序，从而更合理地指导程序并行化操作。

4 例子与实验结果

ZIPS 系统中改进的基于广义 GCD 测试的仿射划分算法进行划分的效率如图 4(a)所示，实验采用的机器是一台 Intel 2.0 GHz 双核 CPU 机器，内存为 2 GB。对于并行化变换的效果，选取了 interp 等 4 种程序，在配置 Linux 系统的 8 核处理器上使用 CMP-SIM 工具[10]模拟了 4 种核数情况，得到的性能结果如图 4(b)所示。



(a)原始划分算法与改进算法耗费时间对比



(b)程序变换处理后的代码性能

图 4 ZIPS 系统的性能实验结果

ZIPS 系统采用基于交互信息的投机并行化方法。实验中，对投机执行模型中的不同策略通过交互功能进行选择和定制，得到的投机并行化性能结果如图 5~图 7 所示。图 5 为 CMP-SIM 模拟的 4 核和 8 核平台下，正常提交策略和部分并行提交策略对加速比的影响。图 6 为顺序创建策略和乱序创建策略对加速比的影响，实验环境为在配置 Linux 系统的 8 核处理器上使用 CMP-SIM 模拟的 4 核 CMP 平台。图 7 显示了线程抽取对象的不同选择对加速比的影响，实验环境为在配置 Linux 系统的 8 核处理器上使用 CMP-SIM 模拟的 6 核平台。

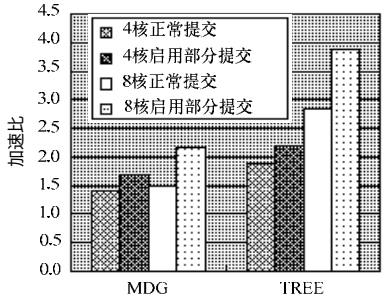


图 5 投机线程不同提交策略的性能结果

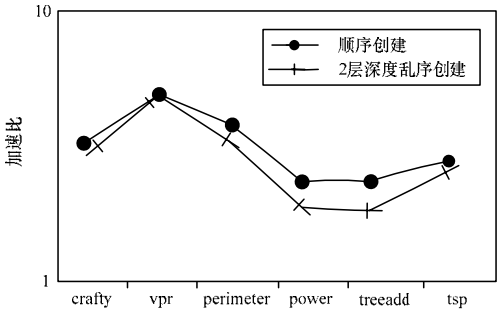


图 6 投机线程不同创建顺序的性能结果

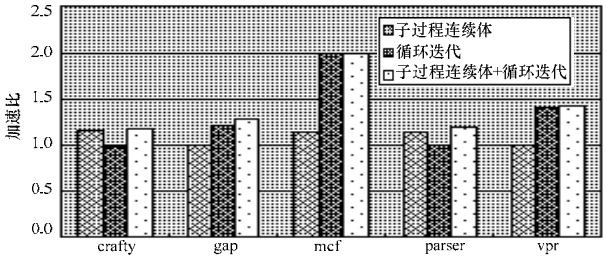


图 7 投机线程不同抽取策略的性能结果

5 结束语

本文介绍了交互式并行化编译系统 ZIPS(ZJU Interactive Parallelizing System)的系统特点及体系架构，并详细阐述了 ZIPS 系统的关键技术和关键算法。该交互式并行化系统提供了友好的交互机制，可以有效地进行串行应用程序并行化处理，能使转换后的程序在多核体系结构下获得好的性能。在下一步的研究工作中，将对基于多体模型的程序变换进行研究来改进自动并行化模块的相关机制，同时将进一步设计交互功能，提供更为强大的用户交互支持来充分获取并行化处理所需的相关信息。

参考文献

[1] Blume W, Eigenmann R, Faigin K, et al. Effective Automatic Parallelization with Polaris[EB/OL]. (1995-05-06). http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.34.3341.

[2] Hall M, Anderson J, Amarasinghe S, et al. Maximizing Multiprocessor Performance with the SUIF Compiler[J]. IEEE Computer, 1996, 29(12): 84-89.

[3] 朱传琪, 臧斌宇, 陈 彤. 程序自动并行化系统[J]. 软件学报, 1996, 7(3): 180-186.

[4] Calidonna C R, Giordano M, Mango F M. A Graphic Parallelizing Environment for User-compiler Interaction[C]//Proc. of the 13th International Conference on Supercomputing. Rhodes, Greece: [s. n.], 1999: 238-245.

[5] Hall M W, Harvey T J, Kennedy K, et al. Experiences Using the ParaScope Editor: An Interactive Parallel Programming Tool[C]//Proc. of the 4th ACM SIGPLAN Symposium on Principles and Practice of Parallel Programming. San Diego, USA: [s. n.], 1993: 33-43.

[6] Sasakura M, Joe K, Araki K. NaraView: An Interactive 3D Visualization System for Parallelization of Programs[J]. International Journal of Parallel Programming, 1999, 27(2): 111-129.

(下转第 255 页)