ISSN 1000-9825, CODEN RUXUEW
Journal of Software, Vol.20, Supplement, December 2009, pp.34−39
© by *Institute of Software, the Chinese Academy of Sciences*. All rights reserved.
E-mail: jos@iscas.ac.cn
http://www.jos.org.cn
Tel/Fax: +86-10-62562563

反馈指导的链式数据结构预取优化*

漆锋滨[1+], 王 飞[2], 李中升[3]

(江南计算技术研究所,江苏 无锡 214083)

Feedback Directed Prefetching Optimization for Linked Data Structure

QI Feng-Bin[1+], WANG Fei[2], LI Zhong-Sheng[3]

(Jiangnan Institute of Computing Technology, Wuxi 214083, China)

+ Corresponding author: E-mail: qifb116@sina.com

Qi FB, Wang F, Li ZS. Feedback directed prefetching optimization for linked data structure. *Journal of Software*, 2009,20(Suppl.):34−39. http://www.jos.org.cn/1000-9825/09005.htm

Abstract: Traditional data prefetching based on static compiler analysis mainly aims at array access. Now, linked data structures due to pointers abound in applications, which is difficult to prefetch by traditional technique. Feedback directed data prefetching optimization has become one of the advanced compiler optimization techniques. it can do well in linked data prefetching. This paper researches the profile guided optimization in ORC compiler, improves the prefetching algorithm according to features of Alpha architecture. SPEC2000 performance test shows 4.1% speedup.

Key words: feedback directed optimization; data prefetching; stride profiling; pointer chasing; linked data structure

摘 要: 传统的基于静态编译的数据预取大多针对数组访问,现在的应用程序中大量出现由指针构成的链式数据结构,依赖传统的编译方法难以进行预取优化.反馈式数据预取优化技术是当前高性能计算技术中前沿的一种编译优化手段,可以很好地解决链式结构的预取问题.在研究 ORC 编译器反馈式编译优化技术的基础上,针对 Alpha 结构的特点,对针对链式结构的反馈式数据预取进行了优化.SPEC2000 测试表明,平均性能提高了 4.1%.

关键词: 反馈式编译优化;数据预取;跨度反馈;指针追逐;链式数据结构

随着技术和工艺的发展,现代高性能微处理器的运算速度越来越快,而存储系统的增长速度却相对较慢,“内存墙”越来越成为系统的一个性能瓶颈.作为一种编译优化方法,数据预取技术为解决该问题提供了一种有效的途径[1].

基于静态编译分析的数据预取技术在国际上已经做了大量研究,它主要针对规则的数组访问.现在的应用程序中出现大量的基于指针构成的链式数据类型,如链表、树、图等数据结构;指针的访问方式与数组元素不同,对于指针 int *p,必须先知道指针 p 的地址,到该地址中读出存放的内容才能得到需要访问的地址,这样对于一个由 p=p->next 构成的链表,要想预取 p 后面的第 n(n>1)个节点,必须访问第 n−1 到第 1 个节点间的所有节点,这就是所谓的“指针追逐”问题.

* Supported by the National Basic Research Program of China under Grant No.2007CB310907 (国家重点基础研究发展计划(973))
Received 2008-07-01; Accepted 2009-04-02

© 中国科学院软件研究所 http://www.c-s-a.org.cn

基于静态的编译分析难于处理链式结构的“指针追逐”问题.在最近的研究中,提出了基于反馈的软件预取方法,如Chilimbi[2],Wu[3],通过程序执行时的信息来获取不规则访存中隐藏的规律,从而伺机插入预取,提高应用程序的性能.

开源编译器 ORC2.1 中实现了基于反馈的数据预取,将其移植到 Alpha 上后,对 SPEC2000 进行反馈预取的测试,发现性能反而下降.本文针对 Alpha 的结构特点,对算法进行改进,使性能得到较大的提升.

1 反馈指导的链式结构数据预取及其在 ORC 中的实现

预取的关键是对可能发生cache不命中的访存地址的预测,并对预取指令进行恰当的调度以求隐藏全部访存延迟.对于规则的数组访问,如Mowry[1]的预取算法,通过局部性分析,循环展开和软件流水可以达到很好的优化效果.

基于指针的链式数据结构,通常依靠 malloc 等内存分配例程为其分配存储空间.研究表明,大量的节点地址之间存在 stride(跨度:也即同一条访存指令在连续两次循环迭代内访存地址的差值)是常数的模式,采用反馈方法可以很有效的发现这些跨度规律,进而进行预取优化.

反馈式预取在 ORC 编译器后端的代码生成阶段(code gen)实施,主要分为以下 4 个过程:

(1) 插桩函数(Do_instrument);

(2) 第 1 遍运行,同时产生访存信息;

(3) 读取反馈信息(Do_annotation);

(4) 插入预取(prefetch insertion).

如图 1 所示,在第 1 遍编译时,寻找属性为循环的基本块,对其中的访存指令插入桩函数,它们负责搜集这些访存的 stride 信息;程序第 1 遍运行将产生这些反馈信息,并写入到中间文件中;第 2 遍编译时,读取反馈信息得到某条访存指令的 stride 信息,通过编译分析得到循环所在基本块的执行时间 tb,访存延迟 Load_Latency/tb 表示需要提前进行预取的迭代次数 d,则 stride*d 即为预取距离,最后在适当的位置插入预取指令.

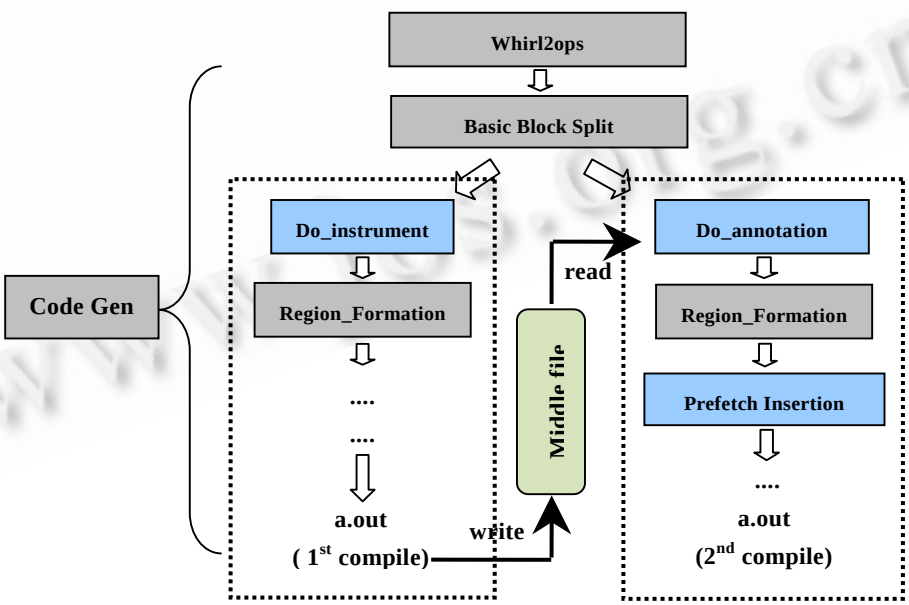

Fig.1 Feedback directed prefetching in ORC

图 1 ORC 反馈式数据预取示意图

© 中国科学院软件研究所 http://www.c-s-a.org.cn

2 反馈指导的链式结构数据预取针对 Alpha 的优化

在本文的工作开始之初,我们在Alpha上已经具有一个可用的编译器[4],在将ORC中的反馈式预取移植到Alpha以后,测试发现经过反馈式预取优化后,SPEC2000 整体性能不升反降,预取没有了效果,反而起负作用.经过反复实验,并仔细分析对比了优化前后产生的汇编代码后,发现主要存在以下两个方面的问题:(1) 原预取算法会产生冗余的预取,带来的开销反而影响了cache性能;(2) 循环展开后增加的访存没有实施预取,且对浮点的lds和ldd指令没有进行处理.

本文针对 Alpha 体系结构的特点,对上述问题提出了相应的解决方案.

2.1 预取最小化

ORC 编译器的目标机器是 EPIC 架构的 Itanium,EPIC 架构固有的顺序执行特性阻碍了它们容忍访存延迟的能力;而乱序执行能力很强的 Alpha,容忍访存延迟的能力相对较强,所以预取作为一种投机优化,在 Alpha 上更容易产生负面影响.

实验发现,Alpha 对预取是很敏感的,预取的访存流量会占用总线带宽;每个预取的高速缓存块会驱逐一个现存的块;此外,预取指令本身需要额外的执行时间,预取地址保存在寄存器中,其他活动变量的寄存器空间将会减少,这样会增加寄存器分配的压力.总之,过多预取带来的额外开销反而会影响 cache 的性能.

预取会发生冗余,即两条预取的目标地址是同一个 cache 行,出现这种情况的原因大多是因为对于同一个结构体内的不同成员或者同一个数组的不同元素分别进行了预取.在这种情况下,本文对预取做最小化处理,也就是减少冗余的预取并修改与之相关的其它预取指令的预取距离.

图 2 给出预取最小化的示意图:对于同一个基地址 R1 的四条预取指令,本文考虑它们的偏移最多跨越的 cache 行数,这里是 3 个 cache 行,偏移为 16 的预取指令是 3 个 cache 行的开始,将偏移 64 变成 80,预取第 2 个 cache 行,偏移为 72 的预取被合并,最后保留偏移为 118 的预取.

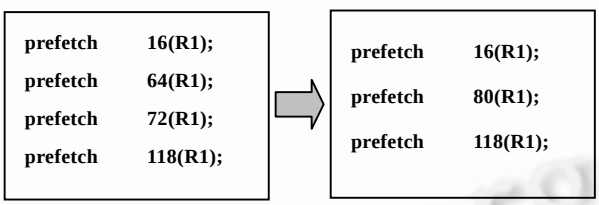

Fig.2 Framework of Prefetch minimization

图 2 预取最小化原理示意图

为了说明预取最小化的应用,图 3 给出一段 300.twolf 的代码实例进行分析:

```
for( net = 1 ; net <= numnets ; net++ ) {
        dimptr = netarray[net] ;
        avg_VW += dimptr->Vweight ;
            cost += ((int)( dimptr->Hweight *
    (double)( dimptr->xmax - dimptr->xmin))) +
            ((int)( dimptr->Vweight *
            (double)( dimptr->ymax - dimptr->ymin)));
```

Fig.3 Example of prefetch minimization

图 3 预取最小化代码示例

图 3 的代码段中,for 循环内出现了对同一个结构体 dimptr 内的不同成员进行访问的模式,对 Vweight,

© 中国科学院软件研究所 http://www.c-s-a.org.cn

Hweight,xmax,xmin,ymax,ymin 这些访存的汇编代码如图 4(a)所示,如果单独考虑每一个访存,不做综合分析,那么对这些访存中的每一条都会产生一条对应的预取指令,这样做显然是不必要的.

ORC 采用了将预取距离不断推后的做法,直到同一次迭代内的预取不会访问同一个 cache 行,如图 4(b)所示.经过分析发现,这种将预取距离不断推后做法尽管避免了部分冗余预取,但它的做法并不准确:因为预取距离不断推后意味着从发射预取到实际数据需要之间间隔的迭代次数不断增加,而实际上这应该是一个常量.实验结果也表明,这种对每条访存都产生预取的方法在 Alpha 上达不到好的性能.

本文不采用 ORC 的调整预取距离的方法,而是采用预取最小化方法.通过分析发现,以$8 作为基址进行的预取距离最多跨 2 个 cache 行,调整预取距离,并消除掉冗余的预取,调整后插入的预取如图 4(c)所示.

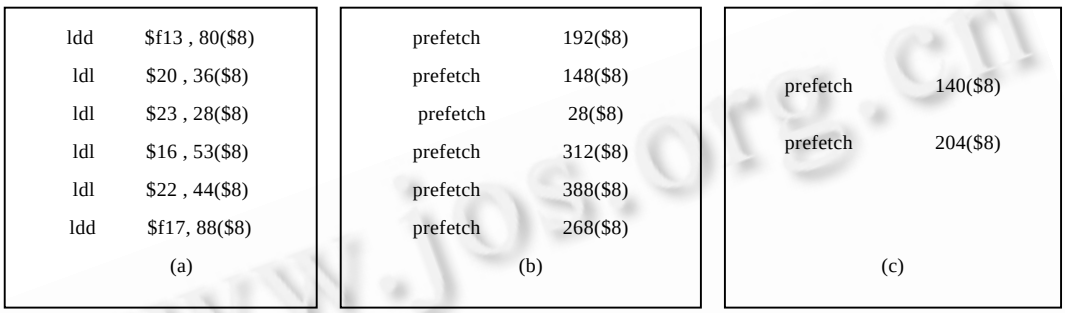

Fig.4 Effect of prefetch minimization

图 4 预取最小化效果示意图

2.2 预取扩大化

第 2.1 节的工作主要是减少不必要的预取,本节论述的工作则增加了预取的数量,本文称之为“预取扩大化”,主要在以下两个地方对算法做出改进.

在循环展开阶段,复制循环体时,预取指令一般只保留一份.这对基于静态编译时产生的预取通常都是适用的,因为静态编译时产生的预取大多针对循环内连续的数组访问,一次成功的预取可以满足多次迭代中计算所需的数据.反馈式预取大多处理链表结构或不规则访存模式,此时,迭代之间的访存一般都不连续.实验中发现大量的跨度都是超过 cache 行大小的,虽然一次迭代会装入一个 cache 行的数据,但下次迭代所需的数据并不会被装入,因此,每次迭代都必须有相对应的预取.本文改进了循环展开部分的算法,对基于反馈式预取产生的预取指令,会判断其对应的访存在迭代之间的跨度,若跨度大于一个 cache 行,那么展开后的预取也会进行复制.

此外,原有的算法主要对整数和指针的访存指令进行反馈处理,为了使反馈式预取更好的作用于浮点应用程序,本文增加了反馈对浮点指令的处理,对于浮点访存,如果第 1 遍插桩后的运行显示它们的访存具有固定的跨度模式,第 2 遍编译就会产生相应的预取指令.试验表明,增加对浮点的处理后,对 SPEC2000 浮点程序也带来一定的性能提升.

经上述改进后,反馈式数据预取显示了令人满意的优化效果,下一章给出本文的实验数据.

3 实验结果

在一台 Alpha21264(linux)工作站上进行了反馈式数据预取优化的实验,采用 SPEC2000 课题进行测试.

相比于 O3 优化,实现反馈式预取后,整数的 181.mcf 和 255.vortex,浮点的 183.equake 都有比较明显的提升,超过了 10%;对整数程序整体提高了 5.8%,对浮点程序整体提高 2.5%,平均提高了 4.1%.下图是 SPEC2000 经反馈预取优化后相比于 O3 的加速比.

© 中国科学院软件研究所 http://www.c-s-a.org.cn

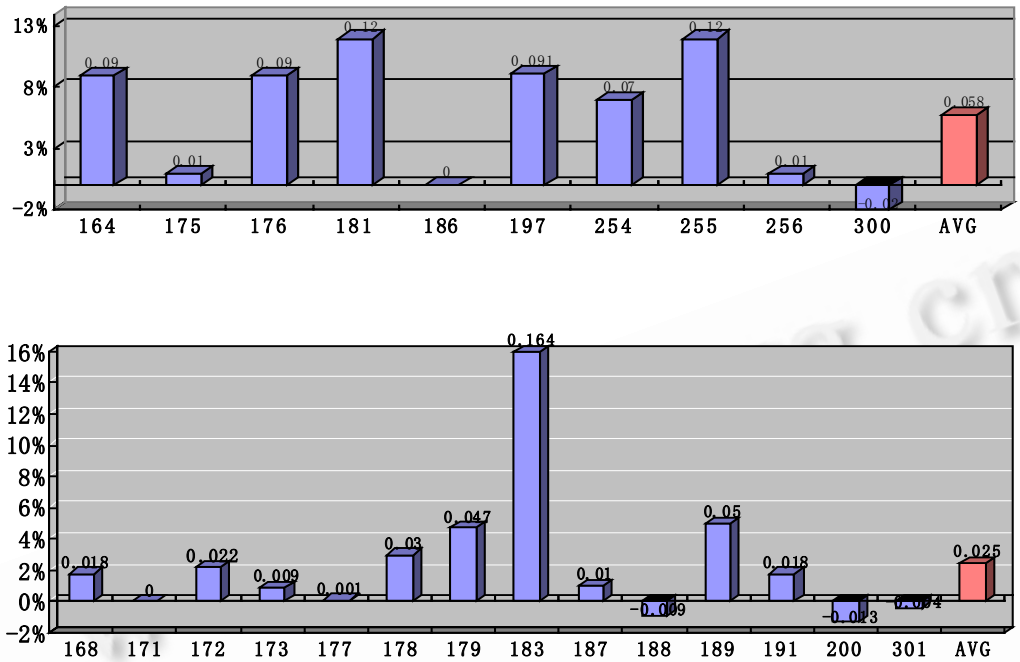

Fig.5 Performance of SPEC2000 Speedup

图 5 SPEC2000 性能优化加速比

4 结束语

在常规的编译优化方法之外,反馈式编译优化技术越来越成为当前比较流行的一种编译优化技术,它对发挥微处理器的性能有着重要的作用,如用于分支预测[5].

本文将反馈式编译优化运用于链式数据结构的预取,创新点在于,将反馈式预取移植到 Alpha 上,并针对 Alpha 结构的特点,提出并实现预取最小化和预取扩大化等改进方案,取得了比较好的优化效果.

本文所采用的方法对预取指令的插入并没有仔细考虑 cache 不命中的情况,在一些情况下,对于已经命中的访存仍然会发出预取,这样带来的额外开销反而会影响 cache 的性能,这也是今后需要进一步研究的方向.

致谢 在此,我们向对本文的工作给予支持和建议的同事表示感谢.

References:

[1] Mowry TC. Tolerating latency through software-controlled data prefetching [Ph.D. Thesis]. Stanford University, 1994.

[2] Chilimbi TM, Hirzel M. Dynamic hot data stream prefetching for general-purpoes programs. In: Proc. of the 2002 ACM SIGPLAN Conf. on Programming Language Design and Implementation. 2002.

[3] Wu YF, Serrano M, Krishnaiyer R, Li W, Fang J. Value profile guided stride prefetching for irregular code. In: Proc. of the 11th Int'l Conf. on Compiler Construction. 2002.

[4] Li ZS, Zhang HJ. Portability analysis of Open64 compiler. High Performance Computing Technology, 2003,160.

[5] Bai SJ, Li ZS, Qi FB. Branch prediction based on feedback directed optimization. High Performance Computing Technology, 2006, 178.

© 中国科学院软件研究所 http://www.c-s-a.org.cn

附中文参考文献:

[4] 李中升,张海军.Open64 编译器的可移植性分析.高性能计算技术,2003,160.

[5] 白书敬,李中升,漆锋滨.基于反馈式编译的转移预测优化.高性能计算技术,2006,178.

漆锋滨(1966－),男,江西人,博士生,高级工程师,主要研究领域为计算机并行体系结构.

王飞(1981－),男,硕士,助理工程师,主要研究领域为高性能编译优化技术.

李中升(1975－),男,高级工程师,主要研究领域为并行编译优化.

www.jos.org.cn

www.jos.org.cn

© 中国科学院软件研究所 http://www.c-s-a.org.cn