实验指导手册

V1.1



华为技术有限公司

|  |
| --- |
| 版权所有 © 华为技术有限公司 2022。 保留一切权利。  非经本公司书面许可，任何单位和个人不得擅自摘抄、复制本文档内容的部分或全部，并不得以任何形式传播。  商标声明  C:\Users\jwx341670\Desktop\华为标志 Huawei Logo 2018\竖版标志Vertical Version\PNG\HW_POS_RBG_Vertical-150ppi.png 和其他华为商标均为华为技术有限公司的商标。  本文档提及的其他所有商标或注册商标，由各自的所有人拥有。  注意  您购买的产品、服务或特性等应受华为公司商业合同和条款的约束，本文档中描述的全部或部分产品、服务或特性可能不在您的购买或使用范围之内。除非合同另有约定，华为公司对本文档内容不做任何明示或暗示的声明或保证。  由于产品版本升级或其他原因，本文档内容会不定期进行更新。除非另有约定，本文档仅作为使用指导，本文档中的所有陈述、信息和建议不构成任何明示或暗示的担保。 |

|  |  |
| --- | --- |
| 华为技术有限公司 | |
| 地址： | 深圳市龙岗区坂田华为总部办公楼 邮编：518129 |
| 网址： | https://[e](http://e.huawei.com/).huawei.com |

目录

[前 言 2](#_Toc117934875)

[简介 2](#_Toc117934876)

[内容描述 2](#_Toc117934877)

[读者知识背景 2](#_Toc117934878)

[实验环境说明 2](#_Toc117934879)

[1 构建实验环境 4](#_Toc117934880)

[1.1 购买ECS 4](#_Toc117934881)

[1.2 通过ssh登录系统 6](#_Toc117934882)

[1.3 安装毕昇编译器 7](#_Toc117934883)

[1.3.1 yum源安装方式 7](#_Toc117934884)

[1.3.2 软件包安装方式 8](#_Toc117934885)

[2 使用毕昇编译器进行自动并行化 10](#_Toc117934886)

[2.1 实验目的 10](#_Toc117934887)

[2.2 实验设备 10](#_Toc117934888)

[2.3 实验原理 10](#_Toc117934889)

[2.3.1 SIMD, 数据级并行与矢量化 10](#_Toc117934890)

[2.3.2 编译器自动矢量化 11](#_Toc117934891)

[2.3.3 线程级并行 11](#_Toc117934892)

[2.4 实验任务操作指导 11](#_Toc117934893)

[2.4.1 开启毕昇编译器自动矢量化 11](#_Toc117934894)

[2.4.2 使用OpenMP使能线程级并行 12](#_Toc117934895)

[2.5 思考题 13](#_Toc117934896)

[3 云资源清理 15](#_Toc117934897)

[3.1 ECS关机 15](#_Toc117934898)

[3.2 删除云资源 16](#_Toc117934899)

[4 术语和缩略语表 18](#_Toc117934900)

前 言

简介

本实验指导手册为《编译原理》课程的实验指导，本课程的实验环境依赖毕昇编译器和鲲鹏920处理器，适用于对编译原理、操作系统、计算机原理等有一定基础，并希望深入熟悉掌握编译优化技术原理和效果的读者。

内容描述

本实验指导手册共包含4个实验，以毕昇编译器为基础，探究从编译器或者编译技术角度来向大家介绍编译优化相关的技术，包括SIMD、PGO、Autotuner等。

* 实验一：使用毕昇编译器进行SIMD矢量优化，获得更好的应用性能。
* 实验二：使用毕昇编译器构建openEuler第三方库，掌握毕昇编译openEuler库的方法。
* 实验三：使用毕昇编译器进行autotuner自动化调优，提升程序性能。
* 实验四：使用毕昇编译器进行PGO反馈优化，深度提升程序性能。

## 读者知识背景

本课程为编译原理基础课程，为了更好地掌握本书内容，阅读本书的读者应首先具备以下基本条件：

具备基本的Linux命令能力；

了解毕昇编译器并会使用毕昇编译器；

具备基本的汇编语言编码能力；

具备基本的C/C++语言编码能力。

## 实验环境说明

华为鲲鹏云主机，其网络拓扑图如下所示：

VPC

ECS

EIP

PC

openEuler 20.03操作系统；

毕昇编译器；

每套实验环境可供一名学员上机操作。

软件方面，本实验需要一台终端电脑与弹性云服务器(ECS)链接以输入操作命令或/和传输文件。对于Windows 10 / macOS / Linux，我们可以用命令行工具ssh和scp完成这个过程。

如果在有些Windows系统下不能运行ssh工具，可以使用MobaXterm工具软件。MobaXterm下载地址：<https://mobaxterm.mobatek.net/download.html>

下文若无特殊说明，均以命令行工具ssh和scp为例进行讲解。

# 构建实验环境

## 购买ECS

在浏览器地址栏输入华为云控制台网址[console.huaweicloud.com](https://console.huaweicloud.com/)并按回车键，这时页面将跳转至登录页。



按要求输入账号密码，进行登录。

注意：在此之前您需要在华为云主页注册华为云账号。

登录成功后会自动进入控制台页面，本例中我们将区域选在 “华北-北京四”。



将鼠标悬停于左侧导航栏图标处展开服务列表。然后在服务列表中点击“弹性云服务器ECS”项。



点击弹性云服务器ECS控制台页面右上角的“购买弹性云服务器ECS”按钮进入购买页面。



按照下表内容配置弹性云服务器ECS的参数。

|  |  |
| --- | --- |
| 参数 | 配置 |
| 计费模式 | 按需计费 |
| 区域 | 华北-北京四 |
| CPU架构 | 鲲鹏计算 |
| 规格 | 鲲鹏通用计算增强型 | kc1.6xlarge.2 | 24vCPUs | 48GB |
| 镜像 | 公共镜像 | openEuler | openEuler 20.03 64bit with ARM(40GB) |
| 系统盘 | 通用型SSD | 40GB |

注意：这里“区域”的配置是和VPC的区域配置保持一致的。

配置完成后点击“下一步：网络配置”，进入网络配置，按下表配置网络参数。

|  |  |
| --- | --- |
| 参数 | 配置 |
| 网络 | vpc-default | subnet-default | 自动分配IP地址 |
| 安全组 | Sys-WebServer |
| 弹性公网IP | 现在购买 |
| 线路 | 全动态BGP |
| 公网带宽 | 按流量计费 |
| 带宽大小 | 10Mbit/s |

配置完成后，点击“下一步：高级配置”，按下表配置ECS高级配置参数。

|  |  |
| --- | --- |
| 参数 | 配置 |
| 云服务器名称 | openEuler（输入符合规则名称） |
| 登录凭证 | 密码 |
| 密码 | 请输入8位以上包含大小写字母、数字和特殊字符的密码，如Euler@123 |
| 确认密码 | 请再次输入密码 |
| 云备份 | 暂不购买 |
| 云服务器组 | 不配置 |
| 高级选项 | 不勾选 |

配置完成后点击右下角“下一步：确认配置”。勾选同意协议，然后点击：“立即购买”。

在提交任务成功后，点击“返回云服务器列表”，返回ECS控制台。

## 通过ssh登录系统

在ECS控制台查看ECS弹性公网IP地址。



在客户端机器操作系统里的Console控制台或Terminal终端里运行ssh命令：

$ ssh root@*119.8.238.181*

（注意：此处的IP地址*119.8.238.181*即是刚刚购买的弹性公网IP地址。）

在客户端（本地PC）第一次登录时会有安全性验证的提示：

The authenticity of host '119.8.238.181 (119.8.238.181)' can't be established.

ECDSA key fingerprint is SHA256:RVxC1cSuMmqLtWdMw4n6f/VPsfWLkT/zDMT2q4qWxc0.

Are you sure you want to continue connecting (yes/no)? yes

在这里输入yes并按回车键继续：

Warning: Permanently added '119.8.238.181' (ECDSA) to the list of known hosts.

Authorized users only. All activities may be monitored and reported.

root@119.8.238.181's password:

输入密码(注意这里不会有任何回显)并回车，登录后的界面如下所示：

Welcome to Huawei Cloud Service

Last login: Wed Aug 19 20:10:20 2020 from 119.3.119.18

Welcome to Huawei Cloud Service

Last login: Wed Aug 19 20:10:20 2020 from 119.3.119.18

Welcome to 4.19.90-2003.4.0.0036.oe1.aarch64

System information as of time: Wed Aug 19 20:19:56 CST 2020

System load: 0.00

Processes: 118

Memory used: 11.9%

Swap used: 0.0%

Usage On: 12%

IP address: 192.168.1.88

Users online: 1

[root@openeuler ~]#

## 安装毕昇编译器

### yum源安装方式

在/etc/yum.repos.d/目录下增加配置文件bisheng-compiler.repo，运行如下命令：

cat > /etc/yum.repos.d/bisheng-compiler.repo << EOF

[bisheng-compiler]

name=bisheng-compiler

baseurl=https://repo.oepkgs.net/bisheng/aarch64/

enabled=1

gpgcheck=0

priority=100

EOF

从yum源仓库下载和安装毕昇编译器rpm包，运行如下命令：

yum update

yum install bisheng-compiler

（可选）清空当前窗口的hash缓存表。

如果系统中有其他版本的 LLVM 编译器，请在安装毕昇编译器之后立即运行如下命令：

hash -r

防止clang命令被hash缓存表捕获，出现毕昇编译器或开源LLVM编译器无法使用的问题。

验证毕昇编译器版本，运行如下命令：

clang -v

若返回结果已包含bisheng compiler版本信息，说明安装成功。

### 软件包安装方式

创建毕昇编译器安装目录（这里以/opt/compiler为例），运行如下命令：

mkdir -p /opt/compiler

下载毕昇编译器软件包，运行如下命令：

wget https://mirrors.huaweicloud.com/kunpeng/archive/compiler/bisheng\_compiler/bisheng-compiler-2.1.0-aarch64-linux.tar.gz

生成当前文件的哈希值，运行如下命令：

sha256sum bisheng-compiler-2.1.0-aarch64-linux.tar.gz

下载毕昇编译器软件包sha256文件，运行如下命令：

wget https://mirrors.huaweicloud.com/kunpeng/archive/compiler/bisheng\_compiler/bisheng-compiler-2.1.0-aarch64-linux.tar.gz.sha256

查看bisheng-compiler-2.1.0-aarch64-linux.tar.gz.sha256文件的内容，运行如下命令：

cat bisheng-compiler-2.1.0-aarch64-linux.tar.gz.sha256

将bisheng-compiler-2.1.0-aarch64-linux.tar.gz.sha256文件的内容与步骤 3中打印的内容进行比较，一致则表示下载的软件包与网站发行的软件包相同。

解压软件包，运行如下命令：

cd /opt/compiler

tar -zxvf bisheng-compiler-2.1.0-aarch64-linux.tar.gz

解压完成后在当前目录下出现名为bisheng-compiler-2.1.0-aarch64-linux的目录。

配置毕昇编译器的环境变量，运行如下命令：

export PATH=/opt/compiler/bisheng-compiler-2.1.0-aarch64-linux/bin:$PATH

export LD\_LIBRARY\_PATH=/opt/compiler/bisheng-compiler-2.1.0-aarch64-linux/lib:$LD\_LIBRARY\_PATH

注意：以上步骤是以/opt/compiler目录举例，若您的安装目录不同，请以实际目录为准。

（可选）清空当前窗口的hash表。

如果系统中有其他版本的 LLVM 编译器，请在安装毕昇编译器之后立即运行如下命令：

hash -r

防止clang命令被hash捕获，出现毕昇编译器或开源LLVM编译器无法使用的问题。

验证毕昇编译器版本，运行如下命令：

clang -v

若返回结果已包含bisheng compiler版本信息，说明安装成功。

# 使用毕昇编译器进行自动并行化

## 实验目的

探究如何使用毕昇编译器进行自动并行化，以获得更好的程序性能表现。

## 实验设备

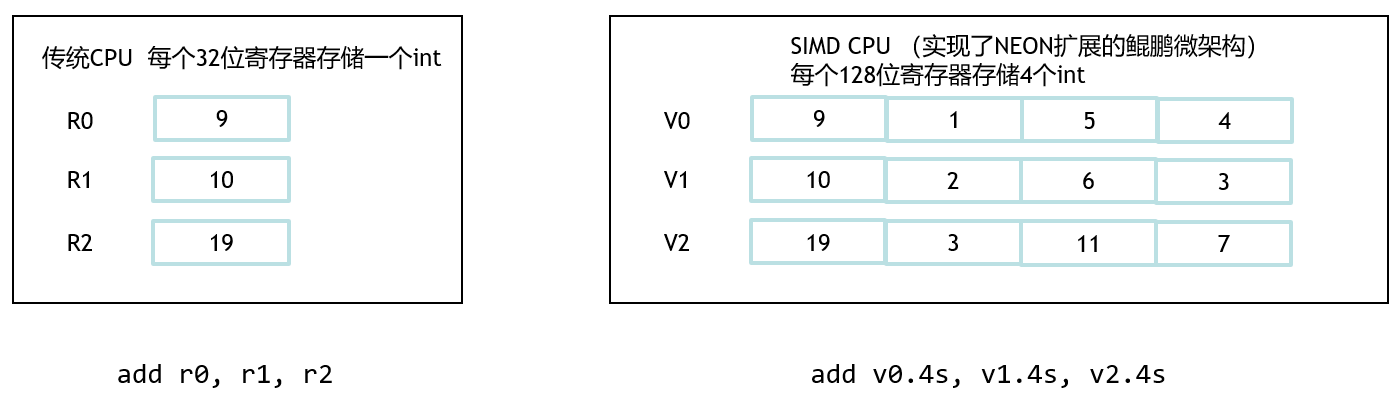
实验设备清单

|  |  |
| --- | --- |
| 操作系统 | CPU类型 |
| openEuler 20.03 LTS SP1 | 鲲鹏920处理器 |
| CentOS Linux release 7.6 aarch64 | 鲲鹏920处理器 |
| CentOS Linux release 8.1 aarch64 | 鲲鹏920处理器 |
| Ubuntu 18.04 | 鲲鹏920处理器 |

## 实验原理

### SIMD, 数据级并行与矢量化

SIMD (Single Instruction Multiple Data) 即单指令流多数据流，是一种采用一个控制器来控制多个处理器，同时对一组数据（又称“数据向量”）中的每一个分别执行相同的操作从而实现空间上的并行性的技术。简单来说就是一个指令能够同时处理多个数据，属于数据级并行（data level parallelism）



硬件基础：ARM NEON 指令集扩展

NEON是ARM平台上的一种基于SIMD思想的技术，可以提供128bit的向量运算能力。NEON技术从ARMv7开始被采用，鲲鹏920微架构已经使能ARM NEON指令集，可以进行矢量化计算操作。

### 编译器自动矢量化

在并行计算中，自动矢量化是自动并行化的一种特殊情况。在编译过程中，可以自动将计算机程序中一次计算操作处理一个数据的标量实现，转换为一个操作同时处理多个数据的矢量实现，以实现自动矢量优化。这种自动矢量优化在一些具有循环计算的场景下提升显著。

### 线程级并行

在并行计算中，线程级并行是另一种并行模式。对于没有依赖的任务，可以利用计算机的多个CPU核来并行执行任务。多线程编程有多种编程模型，本次实验以OpenMP为例，来介绍一种简单快速使能多线程并行化的方法。

## 实验任务操作指导

### 开启毕昇编译器自动矢量化

使用毕昇编译器时添加-O2或-O3选项可以开启自动矢量优化，使用 #pragma clang loop vectorize(disable)可以强制关闭相应矢量优化。

test.c：

#include <math.h>

#include <time.h>

void \_\_attribute\_\_ ((noinline)) f(int N, float \*arr1, float \*arr2, float \*result){

// 删除或注释pragma行即可开启自动矢量化

#pragma clang loop vectorize(disable)

for(int i = 0; i < N; i++) {

result[i] = arr1[i] + arr2[i];

}

}

int main()

{

clock\_t start, finish;

int loop = 10000;

int len = 1000000;

float \*a = (float\*)malloc(sizeof(float) \* len);

float \*b = (float\*)malloc(sizeof(float) \* len);

float \*d = (float\*)malloc(sizeof(float) \* len);

for (int i = 0; i < len; i++) {

//随机生成数组

a[i] = rand() \* 1.8570f - 2.0360f;

b[i] = rand() \* 8.7680f - 6.3840f;

}

start = clock();

for (int i = 0; i < loop; i++) {

f(len, a, b, d);

}

finish = clock();

double time = (double)(finish - start) / CLOCKS\_PER\_SEC;

printf("time: %f s\n", time);

a[0] = d[0];//调用结果d防止编译器将计算过程优化掉

free(a);

free(b);

free(d);

return 0;

}

编译运行

clang test.c -O2 && ./a.out

去掉pragma语句编译运行，比较运行时间差异。

$ clang test.c -O2 && ./a.out

time: 13.160768 s

$ clang test.c -O2 && ./a.out

time: 6.378453 s

### 使用OpenMP使能线程级并行

使用#pragma omp parallel for导语可以对循环部分进行线程级并行优化。此时将会有多个线程并行跑循环的不同迭代，同时每个线程执行的汇编代码使能了自动向量化。

test.c：

#include <math.h>

#include <time.h>

#include <omp.h>

void \_\_attribute\_\_ ((noinline)) f(int N, float \*arr1, float \*arr2, float \*result){

// 在开启自动向量化的基础上添加线程级并行导语

#pragma omp parallel for num\_threads(4)

for(int i = 0; i < N; i++) {

result[i] = arr1[i] + arr2[i];

}

}

int main()

{

clock\_t start, finish;

int loop = 10000;

int len = 1000000;

float \*a = (float\*)malloc(sizeof(float) \* len);

float \*b = (float\*)malloc(sizeof(float) \* len);

float \*d = (float\*)malloc(sizeof(float) \* len);

for (int i = 0; i < len; i++) {

//随机生成数组

a[i] = rand() \* 1.8570f - 2.0360f;

b[i] = rand() \* 8.7680f - 6.3840f;

}

start =omp\_get\_wtime(); // 多线程场景下不能使用clock()函数计算时间

for (int i = 0; i < loop; i++) {

f(len, a, b, d);

}

finish =omp\_get\_wtime();

double time = (double)(finish - start);

printf("time: %f s\n", time);

a[0] = d[0];//调用结果d防止编译器将计算过程优化掉

free(a);

free(b);

free(d);

return 0;

}

编译运行（添加OpenMP参数）

clang test.c -O2 -fopenmp && ./a.out

与只开启矢量化的例子比较运行时间差异。

$ clang -fopenmp test.c -O2 -fopenmp && ./a.out

time: 1.977864 s

## 思考题

假设每个NEON指令可以处理128 bit（4个float类型）的数据，如果一个循环需要处理的float类型数据长度不是4的倍数，这部分数据该如何处理？

参考答案：

编译器会为循环的尾块生成标量的循环，即单步迭代运算。毕昇编译器的后续版本将会支持SVE指令集，支持对变长数据的矢量化，这样对于循环尾块依然可以矢量化。

# **云资源清理**

## ECS关机

实验完成后，回到ECS控制台，勾选openEuler虚拟机，先进行关机。



在弹出的对话框中点击“是”按钮：



## 删除云资源

待关机完成后点击“更多”🡪“删除”。



在弹出的对话框中勾选“释放云服务器绑定的弹性公网IP地址”和“删除云服务器挂载的数据盘”，然后点击“是”，删除ECS。



您可以在控制台点击“更多 | 资源 | 我的资源”菜单项，检查资源是否全部删除。



注意：

(1) 虚拟私有云VPC和安全组可以不删除，以留下次使用。

(2) 若在除“华北-北京四”之外区域（如“亚太-香港”）购买了ECS和EIP，请切换到那个区域查看。

# **术语和缩略语表**

|  |  |  |
| --- | --- | --- |
| 缩略语 | 英文全称 | 中文释义 |
| SIMD | Single Instruction Multiple Data | 单指令流多数据流 |