Задачи на Intel Summer Internship 2015

Нижний Новгород ***3***

Задача 1. **Разработка прототипа модуля проверки OpenCL приложений на наличие кода с неопределенным поведением** **3**

Задача 2. **Реализация алгоритма статического распределения задач по потокам** **3**

Задача 3. **Интеграция взаимодействия библиотеки Intel® MPI со сторонними инструментами на Windows** **3**

Задача 4. **Анализ и разработка коллективных операций MPI библиотеки на общей памяти** **4**

Задача 5. **Java интерфейсы для подмножества MPI-функций** **4**

Задача 6. **Автоматическое определение коммуникационного профиля MPI-приложениий и его оптимизация** **4**

Задача 7. **Сбор легковесной трассы с использованием информации только об имбалансе MPI** **4**

Задача 8. **Верификация алгоритмов синхронизации в библиотеке Intel® Threading Building Blocks (Intel® TBB)** **4**

Задача 9. **Разработка Business Intelligence решения для анализа статуса DPD SW Legal Compliance** **5**

Задача 10. **Оптимизация производительности пользовательского интерфейса Frame Analyzer for OpenGL на операционных системах Windows, Linux и Mac OS** **5**

Задача 11. **Реализация декодеров для разнообразных форматов текстур и Render Target стандартов OpenGL, OpenGL ES и расширений этих стандартов** **5**

Задача 12. **Реализация многопоточности в продукте Intel® DAAL с использованием библиотеки Intel® TBB** **6**

Задача 13. **Создание структуры оценки статистических функций** **6**

Задача 14. **Реализация методов для асинхронной загрузки данных.** **6**

Задача 15. **Разработка полностью автоматизированной системы пакетирования установочных пакетов для продуктов Intel®RealSense™, основанной на удобном структурированном списке с описанием содержания продуктов и компонент** **6**

Задача 16. **Разработка логической игры для Android с использованием передовой технологии Intel RealSense** **7**

Задача 17. **Создание кроссплатформенного мобильного приложения для обработки медиа, базирующегося на технологиях Intel, таких как INDE XRT и MediaForMobile** **7**

Задача 18. **Улучшенный HTML-отчет для утилиты MPI Performance Snapshot** **7**

Задача 19. **Настройка расширений системы рекомендаций в Vectorization Advisor** **7**

Задача 20. **Разработка демо, показывающего возможности системной платы Intel Edison** **7**

Задача 21. **Разработка демо-приложения для Intel® INDE Tamper Protection** **8**

Задача 22. **Разработка демонстрационных приложений для продукта Intel® Integrated Native Developer Experience** **8**

Задача 23. **Разработка лабораторной работы и демо-приложения для демонстрация возможностей оптимизации с использованием библиотеки Intel® Integrated Performance Primitives (IPP)** **8**

Задача 24. **Улучшение открытой библиотеки OpenCV с помощью библиотек повышения производительности Intel® Integrated Performance Primitives** **8**

Задача 25. **Интеграция библиотеки Intel® Integrated Performance Primitives (IPP) в Open Source библиотеку обработки изображений с высоко-уровневым интерфейсом 9**

Задача 26. **Разработка пользовательского интерфейса для модуля управления инфраструктурой, поддерживающей выпуск программных продуктов 9**

Новосибирск ***10***

Задача 1. **Определение ограничений производительности, связанных с пропускной способностью памяти 10**

Задача 2. **Автоматизация измерений производительности 10**

Задача 3. **Автоматизация анализа производительности 10**

Задача 4. **Расширение возможностей инструментов для отладки графического драйвера 10**

Задача 5. **Автоматизированный анализ результатов тестовых прогонов 10**

Задача 6. **Разработать плагин для WebUI инструмента Electric Commander, упрощающий настройку проверочных запусков для подтверждения качества библиотеки Intel® Math Kernel Library 10**

Задача 7. **Анализ/исследование/улучшение полноты тестовой базы гетерогенного компилятора для графического процессора Intel 11**

Задача 8. **Инструменты гетерогенного компилятора для графического процессора Intel 11**

Задача 9. **Автоматизация измерений производительности X32 компилятора 11**

Задача 10. **Разработка системы построения проектов для Intel® Visual Fortran Compiler 11**

Задача 11. **Автоматизация тестирования интегрированных сред разработки для Intel® C/C++ Compiler 12**

Задача 12. **Исследование функциональности и производительности для локальной оптимизации AOS-SOA 12**

Задача 13. **Модуль вывода внутреннего представления компилятора для инструмента оптимизации AOS шаблонов 12**

Задача 14. **Выделение производительных участков в приложениях и их приоритизация 12**

Задача 15. **Реализация алгоритмов автоматической выгрузки для сопроцессора Intel® Xeon Phi™ с использованием технологии hStreams 12**

Задача 16. **Оценка минимального и максимального собственного значения для обощённой задачи собственных значений и интеграция этой функциональности в Intel® Math Kernel Library Extended Eigensolver 12**

Задача 17. **Обзор решений для реализации глубоких и свёрточных нейронных сетей 13**

Задача 18. **Оценка ПО виртуализации для применения в тестах процессорных инструкций в сравнении с эмуляторами и аппаратным обеспечением 13**

Задача 19. **Исследование влияния шаблонов доступа в память на производительность параллельных приложений 13**

Задача 20. **Унификация тестов компоненты LAPACK для библиотеки Intel® MKL для процессоров Intel® Xeon® и Intel® Xeon Phi™ 13**

Нижний Новгород

Задача 1. Разработка прототипа модуля проверки OpenCL приложений на наличие кода с неопределенным поведением

Описание задачи. OpenCL – открытый стандарт для программирования гетерогенных систем (CPU, GPU), является быстроразвивающимся и динамичным стандартом разработки высокопроизводительных приложений для серверов, персональных компьютеров, мобильных и встроенных систем (OpenCL Khronos page).
Необходимо разработать прототип модуля для Intel реализации стандарта OpenCL, задачей которого будет обнаружение кода OpenCL C/C++, содержащего неопределенное поведение (согласно спецификации стандарта), в составе команды разработчиков Intel OpenCL CPU компилятора.
В рамках задачи необходимо:

- Изучить подходы современных компиляторов C, C++ к решению поставленной проблемы;
- Разработать новые компиляторные «проходы» (compiler passes), инструментирующих код программ OpenCL C/C++ и промежуточный машинный код;
- Добавить поддержку нового функционала модулем времени выполнения среды OpenCL;
- Разработать модуль генерации отчетов.

Требования к претенденту.

- Хорошее знание C/C++;
- Технический английский;
- Умение быстро учиться;
- Знание OpenCL и устройства компиляторов будет дополнительным преимуществом.

Задача 2. Реализация алгоритма статического распределения задач по потокам

Описание задачи. Необходимо изучить существующие в библиотеке Intel® Threading Building Blocks реализации и принципы работы алгоритмов распределения задач (простого, автоматического, и на основе вычисленной маски для каждой подзадачи, на которые разбивается исходная задача), используемых в таких параллельных алгоритмах библиотеки, как «parallel_for», «parallel_reduce» и «parallel_scan» (более подробную информацию можно найти по адресу: https://software.intel.com/en-us/node/506149). А также изучить прототип реализации алгоритма статического распределения задач. Определить, какие существуют возможные вариации алгоритма статического распределения задач, которые существенно влияют на производительность параллельных алгоритмов. Написать тесты производительности для этих вариаций, измерить производительность, сравнить и выбрать наиболее подходящий вариант алгоритма. Внедрить его в библиотеку TBB.

Участник познакомится с библиотекой Intel® Threading Building Blocks (https://www.threadingbuildingblocks.org/), получит опыт проведения исследовательских проектов по программированию многоядерных систем, освоит процесс разработки ПО, и сможет похвастаться своей собственной функциональностью в признанном во всём мире программном продукте.

Требования к претенденту.

- Знание языка программирования C++;
- Умение разбираться в существующем коде;
- Опыт многопоточного программирования;
- Желание учиться и быть успешным.

Задача 3. Интеграция взаимодействия библиотеки Intel® MPI со сторонними инструментами на Windows

Описание задачи. Библиотека Intel® MPI library предназначена для реализации обмена сообщениями в кластерных системах и обладает богатыми возможностями работы со сторонними утилитами. В некоторых случаях может потребоваться запустить сторонний инструмент (например, отладчик или профилировщик производительности) на определённом подмножестве процессов, передав ему в качестве параметра целевое приложение. Наиболее распространенным примером является необходимость осуществить анализ работы отдельно выбранных процессов. В случае ОС Linux задача была решена введением новой опции *gtool* для планировщика процессов MPI.
Требуется изучить текущую реализацию gtool на ОС Linux и перевести ее на платформу Windows.

Требования к претенденту.

- Знание языка программирования C;
- межпроцессное взаимодействие на Windows и Linux;
- представление о менеджере процессов Intel(R) MPI.

Задача 4. Анализ и разработка коллективных операций MPI библиотеки на общей памяти

Описание задачи. В настоящее время MPI (Message Passing Interface) является доминирующим стандартом для разработки распределенных приложений для вычислительных кластеров. С появлением многоядерных платформ встала задача оптимизации библиотеки не только для передачи сообщений между узлами кластера, но и для обмена данными внутри узла (с использованием общей памяти на узле). Одно из типичных «узких» мест при передаче сообщений – так называемые коллективные операции. Библиотека Intel® MPI library является одной из наиболее производительных реализаций стандарта MPI.

Требуется провести анализ существующей инфраструктуры коллективных операций на общей памяти, сделать детальный анализ их производительности и реализовать новые алгоритмы.

Требования к претенденту.

- Знание языка C на хорошем уровне;
- Знание примитивов межпроцессорного взаимодействия Linux и примитивов синхронизации;
- Знание алгоритмов коллективных операций – плюс.

Задача 5. Java интерфейсы для подмножества MPI-функций

Описание задачи. MPI является доминирующим стандартом для написания распределенных высокопроизводительных программ для вычислительных кластеров. В настоящее время Intel MPI не предоставляет Java интерфейсов к своей функциональности, что ограничивает его применение языками C, C++ и Fortran.

Задача состоит в том, чтобы выделить подмножество функций, наиболее часто используемых в реальных приложениях, и разработать для них соответствующие интерфейсы для вызова из Java.

Требования к претенденту.

- Понимание MPI;
- Хорошее знание C и Java;
- Преимуществом является опыт использования JNI.

Задача 6. Автоматическое определение коммуникационного профиля MPI-приложениий и его оптимизация

Описание задачи. В настоящее время MPI (Message Passing Interface) является доминирующим стандартом для разработки распределенных приложений для вычислительных кластеров. Для тонкой настройки и оптимизации библиотеки под конкретные приложения необходимо уметь выделять в них паттерны обмена сообщениями. Для этого командой Intel® MPI был реализован прототип «проигрывателя» коммуникационных схем приложений.

Задачи будут следующими:

1. Расширить функциональность существующего прототипа для снятия/воспроизведения MPI профиля под нужды приложения (Обязательная программа)
2. Исследовать поведение MPI профиля отдельно от приложения. (Дополнительная программа)

Требования к претенденту.

- Уверенное знание C/C++;
- Linux;
- Опыт работы с MPI и/или BOOST будут дополнительным плюсом.

Задача 7. Сбор легковесной трассы с использованием информации только об имбалансе MPI

Описание задачи. Intel® Trace Analyzer and Collector (ITAC) является средством для профилирования производительности распределенных программ, написанных с использованием стандарта передачи сообщений MPI (Message Passing Interface).В связи с большим количеством процессов в типичной MPI программе, как правило, информация, собираемая ITAC, является очень объемной и зачастую избыточной для неискушенного пользователя.

Требуется разработать прототип для Intel® Trace Analyzer and Collector, который будет записывать в трассу информацию только о тех функциях, в которых есть внутренний дисбаланс (с помощью новых функций-перехватчиков MPI вызовов). Прототип будет основан на текущей реализации ITAC с целью последующей интеграции в продукт.

Требования к претенденту.

C, perl, MPI

Задача 8. Верификация алгоритмов синхронизации в библиотеке Intel® Threading Building Blocks (Intel® TBB)

Описание задачи. Библиотека Intel® Threading Building Blocks (Intel® TBB) предоставляет набор высокоуровневых алгоритмов для параллельных вычислений, позволяя существенно упростить процесс разработки параллельных приложений для многоядерных процессоров. Библиотека Intel TBB содержит ряд нетривиальных параллельных алгоритмов, активно использующих атомарные операции.

Необходимо изучить алгоритмы синхронизации, реализованные в Intel TBB, и разработать модели изученных алгоритмов в средах автоматической верификации (например, верификатор программных моделей SPIN). Существенная часть такой работы – понять, являются ли обнаруженные проблемы ошибками в реализации TBB или ограничениями модели. Особый интерес представляет изучение корректности работы алгоритмов с различными моделями консистентности памяти. Больше о предметной области можно узнать из обзора M. Desnoyers «Proving the Correctness of Nonblocking Data Structures».

Требования к претенденту.

- Знание C++ и умение разбираться в существующем коде;
- Опыт многопоточного программирования, представление о неблокирующей синхронизации;
- Способность быстро изучать требуемые инструменты моделирования.

Задача 9. Разработка Business Intelligence решения для анализа статуса DPD SW Legal Compliance

Описание задачи. Задача состоит из 3-х составляющих:

1) разработка модуля в Data Warehouse: сбор и предварительная обработка данных из нескольких источников, привязка их к существующим SW продуктам.
2) анализ требования по составлению отчета на основе полученных данных, выбор наиболее подходящей платформы для разработки отчета: Microsoft Business Intelligence (MS BI) или SAP Business Objects (BObj).
3) создание самих отчетов и предоставление их заказчику.

Требования к претенденту.

- Знание SOAP;
- Знание MS SQL (умение писать запросы и процедуры).

Дополнительным плюсом могут быть:

- Владение языками программирования Java или C#;
- Знание MS BI или SAT Bobj;
- WSDL, XSD.

Задача 10. Оптимизация производительности пользовательского интерфейса Frame Analyzer for OpenGL на операционных системах Windows, Linux и Mac OS

Описание задачи. Удобный пользовательский интерфейс - это один из ключевых компонентов успешности продукта. Очень важно, чтобы интерфейс имел приемлемое время отклика на действия пользователя, был доступен и стабилен вне зависимости от того, насколько сложные запросы обрабатываются в более низких слоях архитектуры продукта. Целью данной задачи является улучшение производительности пользовательского интерфейса анализатора производительности графических приложений Frame Analyzer for OpenGL на наборе операционных систем.

Необходимо:

- изучить Frame Analyzer for OpenGL и базовые подходы к профилированию игр;
- сделать анализ производительности Frame Analyzer for OpenGL на Windows, Linux и Mac OS;
- идентифицировать узкие места, ухудшающие производительность;
- предложить способы их устранения;
- реализовать хотя бы один из предложенных способов;
- показать улучшение производительности.

Требования к претенденту.

- Двухлетний опыт прогаммирования на C++.

Желательные требования:

- Опыт программирования на QML;
- Опыт программирования на Windows, Linux и Mac OS.

Задача 11. Реализация декодеров для разнообразных форматов текстур и Render Target стандартов OpenGL, OpenGL ES и расширений этих стандартов

Описание задачи. Современные стандарты графических API декларируют большое количество форматов текстур и Render Target, которые варьируются для разных версий API, вида API, зависят от производителя графического процессора и т.д. Для того, чтобы анализаторы производительности графических приложений были конкурентно способными и могли профилировать современные игры, необходимо поддерживать наиболее востребованные форматы текстур и Render Target. Эта задача нацелена на расширение поддерживаемого набора форматов для анализатора производительности графических приложений Frame Analyzer for OpenGL.

В рамках этой задачи нужно

- изучить Frame Analyzer for OpenGL и базовые подходы к профилированию игр;

- определить набор форматов текстур и Render Target, которые нужно поддержать;
- изучить спецификации этих форматов;
- реализовать декодеры для этих форматов;
- проверить качество реализации декодеров;
- интегрировать новые декодеры в код продукта.

Требования к претенденту.

- Опыт программирования на C++ от 1 года;
- Желательно иметь опыт программирования на OpenGL или OpenGL ES.

Задача 12. Реализация многопоточности в продукте Intel® DAAL с использованием библиотеки Intel® TBB

Описание задачи. Intel® DAAL - это новый продукт, который выходит на рынок в 2015 году. Продукт представляет из себя набор блоков, оптимизированных для работы на всех платформах Intel, которые можно будет использовать на всех стадиях анализа данных: сбора данных из различных источников, предварительной обработки, преобразования, анализа, моделирования, проверки и принятия решений.

Требуется провести анализ реализованных алгоритмов на предмет многопоточности, сделать оценку прироста производительности при применении библиотеки Intel® TBB и оптимизировать алгоритмы для достижения высокой степени параллельного выполнения.

Требования к претенденту.

- Хороший математический аппарат (численные методы, теория вероятности, математический анализ);
- Умение программировать на C++;
- Знание библиотеки Intel® TBB будет дополнительным плюсом.

Задача 13. Создание структуры оценки статистических функций

Описание задачи. Intel® DAAL - это новый продукт, который выходит на рынок в 2015 году. Продукт представляет из себя набор блоков, оптимизированных для работы на всех платформах Intel, которые можно будет использовать на всех стадиях анализа данных: сбора данных из различных источников, предварительной обработки, преобразования, анализа, моделирования, проверки и принятия решений.

Требуется изучить алгоритмы, которые используются в библиотеке Intel® DAAL и реализовать алгоритмы для оценки точности вычислений в различных условиях. Изучить существующие продукты в области аналитики и разработать методы для оценки их работы.

Требования к претенденту.

- Хороший математический аппарат (численные методы, теория вероятности, математический анализ);
- Умение программировать на C++, желательно знание скриптовых языков Python/Perl/bash;
- Умение совершать/программировать операции с плавающей точкой (знание IEEE 754 стандарта) будет дополнительным плюсом.

Задача 14. Реализация методов для асинхронной загрузки данных.

Описание задачи. Intel® DAAL - это новый продукт, который выходит на рынок в 2015 году. Продукт представляет из себя набор блоков, оптимизированных для работы на всех платформах Intel, которые можно будет использовать на всех стадиях анализа данных: сбора данных из различных источников, предварительной обработки, преобразования, анализа, моделирования, проверки и принятия решений.

Требуется изучить работу системы асинхронной передачи данных в Apache* Spark, создать прототип выгрузки этих данных и их передачи в алгоритм для дальнейшего анализа.

Требования к претенденту.

- Хороший математический аппарат (численные методы, теория вероятности, математический анализ);
- Умение программировать на C++/Java;
- Знание Apache* Spark будет дополнительным плюсом.

Задача 15. Разработка полностью автоматизированной системы пакетирования установочных пакетов для продуктов Intel®RealSense™, основанной на удобном структурированном списке с описанием содержания продуктов и компонент

Описание задачи. Задача включает расширение существующей функциональности пакетирования (построенной на Python, Batch, perl скриптах, а также Microsoft TFS.) путем добавления новой функциональности: централизованный полноценный список с описанием содержания продуктов и компонент, а также добавление функциональности по автогенерации всех конфигурационных файлов установочных пакетов (BOM).

Требования к претенденту.

Обязательные навыки:
- Уверенное владение Python;
- Отладка сложных программных систем;
- Основы написания скриптов на Batch;
- Английский на уровне чтения технической документации.

Желательные навыки:
- Javascript;
- C/C++;
- Опыт работы с виртуальными машинами;
- Знакомство с Windows Installer и WiX Toolset;
- XAML, TFS workflows.

Задача 16. Разработка логической игры для Android с использованием передовой технологии Intel® RealSense™

Описание задачи. Разработать для Android устройств на базе архитектуры Intel логическую игру (для развития логического мышления у детей), которая будет использовать распознавание жестов и мимики.

Требования к претенденту.
- Знание C/C++ или Java;
- Знание архитектуры Android приложений

Задача 17. Создание кроссплатформенного мобильного приложения для обработки медиа, базирующегося на технологиях Intel, таких как INDE XRT и MediaForMobile

Описание задачи. Специально для разработчиков ПО под мобильные платформы Intel предоставляет набор инструментов и библиотек INDE (Intel Integrated Native Developer Experience). В его состав входит кроссплатформенная библиотека для обработки медиа данных, называемая Media For Mobile (M4M). Имеется набор примеров приложений, использующих M4M на таких операционных системах, как iOS и Android. Также в скором времени в INDE будет добавлена XRT технология позволяющая запускать Java код под iOS.

Задача состоит в создании Java-приложения, базирующегося на технологии INDE XRT, запускающегося на Android и iOS и использующего INDE M4M библиотеку для работы с медиа. В качестве прототипов можно использовать существующие примеры для M4M.

Требования к претенденту.
- Знание Java;
- Опыт разработки под Android;
- Знакомство с Obj-C.
- Желательно опыт работы с iOS API.

Задача 18. Улучшенный HTML-отчет для утилиты MPI Performance Snapshot

Описание задачи. MPI Performance Snapshot (MPS) является легковесным масштабируемым средством для профилирования производительности распределенных программ, написанных с использованием стандарта передачи сообщений MPI (Message Passing Interface). Основным форматом вывода для этой утилиты является простой текстовый формат при работе с утилитой из командной строки. Но некоторые данные визуализируются с помощью генерируемого HTML5-отчета. Данная задача предполагает расширение этого HTML5-отчета – добавление новых интерактивных представлений данных, базируясь на интерфейсе командной строки данной утилиты.

Требования к претенденту.

HTML, JS, C++

Задача 19. Настройка расширений системы рекомендаций в Vectorization Advisor

Описание задачи. Используя рекомендации экспертов по оптимизации, основывающиеся на диагностиках компилятора и статическом анализе кода, написать примеры кода для тестирования текущих рекомендаций и выработки критериев для неисследованных проблем производительности и векторизации. Разработать алгоритмы детектирования проблем для автоматической системы рекомендаций.

Требования к претенденту.
- Понимание различных техник оптимизации кода;
- Желательно хорошее знание C++ и знакомство с Python;
- Будет плюсом хорошее знание английского из-за потенциальной необходимости общаться с англоязычными экспертами.

Задача 20. Разработка демо, показывающего возможности системной платы Intel Edison

Описание задачи. Придумать и разработать демонстрацию, показывающую функциональность и преимущества платы Intel Edison.

Требования к претенденту.

Знание C/C++ или Java. Базовые знания схемотехники.

Задача 21. Разработка демо-приложения для Intel® INDE Tamper Protection

Описание задачи. Intel® INDE Tamper Protection – это новый продукт компании Intel, который защищает секреты в пользовательском коде от статического и динамического реверс-инжиниринга и обеспечивает целостность кода во время исполнения.
Необходимо разработать Windows-приложение с использованием Intel® Tamper Protection, взяв за основу свою оригинальную идею.

Требования к претенденту.

- Хорошее знание основ криптографии (протоколы, алгоритмы шифрования и подписи и т.д.);
- Уверенное программирование на языке C/C++;
- Высокий уровень навыков письменного и разговорного английского языка.

Задача 22. Разработка демонстрационных приложений для продукта Intel® Integrated Native Developer Experience

Описание задачи. Задача заключается в разработке демонстрационных приложений для продукта Intel® Integrated Native Developer Experience. Приложения должны объединить в себе (где это возможно) все компоненты этого программного продукта для демонстрации преимуществ разработчикам, работающим в мобильном сегменте.

Требования к претенденту.

- Знание операционных систем Windows, OSX и Adnroid, хорошие навыки разработки под данные операционные системы;
- Знание C++ и Java;
- Знание сред разработки Visual Studio и Android Studio, набора инструментов Android SDK;
- Уверенное знание английского языка.
- Знание алгоритмов работы с видео и аудио и популярных форматов будет дополнительным преимуществом.

Задача 23. Разработка лабораторной работы и демо-приложения для демонстрация возможностей оптимизации с использованием библиотеки Intel® Integrated Performance Primitives (IPP)

Описание задачи. Библиотека Intel® IPP постоянно эволюционирует, стараясь отвечать изменяющимся требованиям экосистемы. Необходимо подготовить пошаговую лабораторную работу, рассказывающую о том, как правильно использовать Intel® IPP и о том какие улучшения получит пользователь библиотеки. Кроме этого необходимо будет разработать демо-приложения, которые бы демонстрировали возможности библиотеки на каких-то конкретных примерах.

Требования к претенденту.

- Хорошее знание C/C++;
- Знакомство с IPP;
- Умение понимать чужой код;
- Умение разрабатывать под Windows.
- Дополнительный плюс: умение работать в Linux, OS X, Android

Задача 24. Улучшение открытой библиотеки OpenCV с помощью библиотек повышения производительности Intel® Integrated Performance Primitives

Описание задачи. Необходимо повысить производительность открытой библиотеки OpenCV на архитектурах компании Intel с помощью библиотеки повышения производительности Intel® Integrated Performance Primitives (IPP). Версия OpenCV 3.0 уже имеет интегрированную библиотеку IPP for OpenCV (ICV), являющуюся небольшим подмножеством IPP. Необходимо увеличить число OpenCV функций, ускоренных с помощью IPP. Для этого нужно будет вставить предложенные IPP функции в соответствующие места OpenCV, провести все необходимые тесты и подготовить все для интеграции изменений в публично доступную версию OpenCV.

Требования к претенденту.

- Хорошее знание C/C++;
- Знакомство с OpenCV и IPP;
- Умение понимать чужой код;
- Умение разрабатывать под Windows.
- Дополнительный плюс: умение работать в Linux, OS X, Android.

Задача 25. Интеграция библиотеки Intel® Integrated Performance Primitives (IPP) в Open Source библиотеку обработки изображений с высоко-уровневым интерфейсом

Описание задачи. Необходимо:

- Провести анализ существующих Open Source библиотек обработки изображений на языке C++ и Java.
- Интегрировать IPP в выбранную библеотеку
- Разработать демо-приложения базе выбранной библиотеки, показывающие преимущества использования библиотеки IPP.

Требования к претенденту.

- Хорошее знание C/C++;
- Знакомство с IPP;
- Умение понимать чужой код;
- Умение разрабатывать под Windows.
- Дополнительный плюс: умение работать в Linux, OS X, Android.

Задача 26. Разработка пользовательского интерфейса для модуля управления инфраструктурой, поддерживающей выпуск программных продуктов

Описание задачи. Разработать веб-интерфейс для администраторов, позволяющий им настраивать компоненты распределённой инфраструктуры и связи между ними, а также отслеживать состояние системы. Решение должно быть написано на языке JavaScript и обращаться к сервисам посредством REST API.

Требования к претенденту.

- Опыт разработки на JavaScript (или диалекте);
- Базовые представления об HTML и CSS;
- Базовые представления о REST API;
- Базовые представления о ReactJS приветствуются.

Новосибирск

Задача 1. Определение ограничений производительности, связанных с пропускной способностью памяти

Описание задачи. Оптимизации размещений в памяти имеют важное значение для высокопроизводительных приложений. Задача включает в себя изучение основных факторов ограничений производительности, связанных с доступом к памяти и автоматизации выполняемых действий анализа для обнаружения таких проблем и их соотношению к определенной части исходного кода. Еще одним направлением является оценка модели доступа памяти на основе динамического профилирования. В рамках данного проекта вы получите представление о том, как работа с памятью влияет на производительность кода, написанного на языке высокого уровня.

Требования к претенденту.

- C/C++ Linux, языки сценариев, x86 ассемблер;
- Знакомство с компиляторными оптимизациями и инструментами профилирования является дополнительным преимуществом.

Задача 2. Автоматизация измерений производительности

Описание задачи. Команда измерения производительности производит большое количество замеров. Для своевременного предоставления результатов требуется управлять конфигурациями тестирования гибко и эффективно. Предлагается реализовать соответствующий инструментарий на C#. В рамках этого проекта вы на практике сможете научиться создавать комплексное программное обеспечение.

Требования к претенденту.

- Хорошие знания математики и программирования на C# и SQL;
- Знание XML, Linq, CI и классических QA процессов желательны.

Задача 3. Автоматизация анализа производительности

Описание задачи. В команде тестирования производительности еженедельно автоматически анализируются сотни падений. Учитывая специфику тестирования, во многих случаях требуется аналитика и сбор дополнительной информации (часто вручную) для принятия решения о существовании проблемы. Автоматизация сбора дополнительной информации и есть задача для летней школы, требующая понимания работы программно-аппаратного комплекса в целом. В рамках этого проекта вы познакомитесь с подходами к созданию автоматических инструментов обработки неоднородных данных для поиска проблем в сложном программном обеспечении.

Требования к претенденту.

- Хорошее знание математики и техническое образование, навыки программирования на C#, SQL, Shell, Perl;
- понимание языка разметки XML и технологии доступа к данным Linq;.
- Знание систем continuous integration и умение работать с инструментами Intel для профилировки (Intel® Vtune™ Amplifier) является плюсом;
- Желательно знание организации классических процессов тестирования.

Задача 4. Расширение возможностей инструментов для отладки графического драйвера

Описание задачи. Усовершенствовать инструменты, используемые при тестировании и отладке графического драйвера. Главная задача - автоматизировать локализацию проблем в драйвере. В рамках этого проекта вы познакомитесь с подходами к созданию автоматических инструментов поиска проблем в сложном программном обеспечении.

Требования к претенденту.

C++, OpenGL

Задача 5. Автоматизированный анализ результатов тестовых прогонов

Описание задачи. Разработать анализатор логов тестовых прогонов, выявляющий регрессии, уже описанные дефекты, и представляющий результаты в удобном для использования виде. В рамках этого проекта вы познакомитесь с подходами к созданию автоматических анализаторов неоднородных данных и созданию удобных в использовании интерфейсов.

Требования к претенденту.

Знание языков Python, JavaScript

Задача 6. Разработать плагин для WebUI инструмента Electric Commander, упрощающий настройку проверочных запусков для подтверждения качества библиотеки Intel® Math Kernel Library

Описание задачи. Разработчикам библиотеки математических функций Intel® Math Kernel Library необходима возможность запускать любой этап проверки продукта, чтобы быть уверенным в

качестве интегрируемого кода. Чтобы эту возможность предоставить, необходимо написать расширение к текущему плагину WebUI для инструмента Electric Commander, позволяющее гибкую конфигурацию подобных задач с подтверждением правильности введенных данных «на лету» с целью избежать ненужных и неправильных запусков проверки библиотеки. В рамках этого проекта вы приобретёте навыки автоматического тестирования сложного программного обеспечения на множестве разнородных платформ и создания удобного для использования интерфейса работы с комплексом автоматического тестирования.

Требования к претенденту.

- Знание Java/GWT;
- Eclipse;
- базовые знания скриптовых языков (shell, Perl).

Задача 7. Анализ/исследование/улучшение полноты тестовой базы гетерогенного компилятора для графического процессора Intel

Описание задачи. В данной задаче требуется провести анализ, оптимизацию и расширение тестовой базы, используемой для тестирования гетерогенного компилятора для графического процессора Intel. Во время работы над задачей студент получит знания об организации процесса разработки и тестирования гетерогенного компилятора, научится создавать и усовершенствовать тесты для гетерогенной вычислительной среды, приобретет начальные знания по ускорению программ на графических процессорах.

Требования к претенденту.

- Знакомство со следующими языками и технологиями: C/C++, perl, sh/csh/bash, python, xml;
- Знание OS Windows 7, 8+, Linux.

Задача 8. Инструменты гетерогенного компилятора для графического процессора Intel

Описание задачи. В данной задаче необходимо создать инструменты для улучшения эффективности разработчика. В процессе работы над инструментами, студент получит базовые навыки по организации разработки компилятора, процесса тестирования и специфику разработки гетерогенного компилятора.

Детали задачи:

changeset-manager. Инструмент представляет из себя Java GUI приложение для формирования комманд настройки компиляторов на архивные версии (changesets).

Необходима настройка параметров (тип изменения/changeset'а, архитектура, OS и т.д.) и функция поиска последнего исправного изменения. Полный список функций приложения и требования к нему будут описаны в техническом задании.

Универсальный кроссплатформенный скрипт для тестирования и анализа результатов тестирования GEN-компилятора.

Требования к претенденту.

- Знакомство со следующими языками и технологиями: java, perl, sh/csh/bash, python, xml;
- Знание OS Win 7, 8+, Linux.

Задача 9. Автоматизация измерений производительности X32 компилятора

Описание задачи. В рамках данной задачи студент научится использовать стандартизованные бенчмарки для измерения производительности генерируемого кода, адаптировать их код для X32, произведет замеры и сравнение производительности реального продуктового компилятора и сможет включить эти измерения в автоматическую систему отслеживания производительности.

Требования к претенденту.

- Умение работать в Linux OS,
- Знание основ языка программирования C.

Задача 10. Разработка системы построения проектов для Intel® Visual Fortran Compiler

Описание задачи. Работая над этой задачей, студент научится создавать плагины для Microsoft* Visual Studio*, изучит механизм работы MSBuild, получит практический опыт разработки, отладки и тестирования сложных сценариев MSBuild, который может быть применен при работе с любой современной системой автоматичекой сборки проектов, такой как ant, maven или gradle.

Требования к претенденту.

- Опыт разработки в Microsoft* Visual Studio;
- Знание операционной системы Windows* (системный реестр, глобальный кэш сборок);
- Уровень владения английским языком – intermediate.

Будет плюсом:

- Опыт разработки на C#, C++ или Java;
- Знание систем автоматической сборки проектов, таких как ant, maven или gradle.

Задача 11. Автоматизация тестирования интегрированных сред разработки для Intel® C/C++ Compiler

Описание задачи. В ходе работы над задачей студент познакомится с методами автоматизации тестирования графического пользовательского интерфейса на примере интегрированных сред разработки Eclipse* и Microsoft* Visual Studio*. Также студент получит опыт организации процесса тестирования и создания автоматизированных тестов для графических приложений на OS Windows* и Linux.

Требования к претенденту.

Минимальные навыки:

- Опыт разработки на C#, Java или C++;
- Уровень владения английским языком - intermediate.

Будет плюсом:

- Знакомство с системами контроля версий;
- Знакомство со скриптовыми языками (Shell, Batch).

Задача 12. Исследование функциональности и производительности для локальной оптимизации AOS-SOA

Описание задачи. Студент приобретет навыки проверки качества преобразований, осуществляемых компиляторами, а также узнает некоторые нюансы архитектур процессоров Intel. Используя это знание, студент научится создавать надежные kernel-тесты, позволяющие измерять прирост производительности от конкретных оптимизаций компилятора Intel.

Требования к претенденту.

- Знание C/C++;
- Linux shell;
- Microsoft Excel;
- Знание x86 ассемблера желательно.

Задача 13. Модуль вывода внутреннего представления компилятора для инструмента оптимизации AOS шаблонов

Описание задачи. Студент будет ответственен за создание модуля вывода для существующего приложения оптимизации AOS шаблонов, который будет производить описание оптимизированных последовательностей инструкций в виде внутреннего языка компилятора Intel. Студент познакомится с SIMD интринсиками Intel компилятора и набором инструкций процессоров Intel.

Требования к претенденту.

Знание C/C++, SVN, Linux shell

Задача 14. Выделение производительных участков в приложениях и их приоритизация

Описание задачи. Студент научится пользоваться такими инструментами разработки от Intel, как компилятор и функциональные симуляторы процессоров. Студенту необходимо будет исследовать и решить сложную задачу сопоставления профиля исполнения программы на симуляторе с отчетами об оптимизации, создаваемыми компилятором Intel.

Требования к претенденту.

- Знание Perl или другого производительного скриптового языка;
- Linux shell;
- Знание x86 ассемблера желательно.

Задача 15. Реализация алгоритмов автоматической выгрузки для сопроцессора Intel® Xeon Phi™ с использованием технологии hStreams

Описание задачи. Задача включает изучение технологии hStreams для выгрузки вычислений на сопроцессор Intel® Xeon Phi™ и реализация гибридных алгоритмов компоненты LAPACK с использованием этой технологии. В рамках этого проекта вы получите навыки создания эффективных параллельных программ для современных процессоров, состоящих из сотен ядер.

Требования к претенденту.

- Опыт программирования на языке C;
- Знание инструментов разработки для OS Linux;
- Опыт разработки для Intel® Xeon Phi™ будет плюсом.

Задача 16. Оценка минимального и максимального собственного значения для обощённой задачи собственных значений и интеграция этой функциональности в Intel® Math Kernel Library Extended Eigensolver

Описание задачи. Задача включает разработку функциональности, позволяющей вычислить N минимальных или максимальных собственных значений, и интеграцию этой функциональности в компоненту Extended Eigensolver. В рамках этого проекта вы получите представление о том, как на

практике реализовывать программное обеспечение, которое способно надёжно и быстро решать одну из сложнейших проблем линейной алгебры.

Требования к претенденту.

- Математическое образование;
- Базовое знание OS Linux;
- Навыки программирования на языках C и Fortran.

Задача 17. Обзор решений для реализации глубоких и свёрточных нейронных сетей

Описание задачи. Задача включает обзор существующих решений для глубоких и свёрточных нейронных сетей с целью идентификации ключевых вычислительных ядер и возможностей для расширения функциональности математической библиотеки Intel® Math Kernel Library. В рамках этого проекта вы получите представление об одной из наиболее востребованных и перспективных областей для создания искусственного интеллекта и автоматической обработки разнородных данных.

Требования к претенденту.

- Опыт разработки для OS Linux;
- Владение языками C и C++;
- Базовое понимание архитектуры Intel;
- Опыт работы с нейросетями будет плюсом.

Задача 18. Оценка ПО виртуализации для применения в тестах процессорных инструкций в сравнении с эмуляторами и аппаратным обеспечением

Описание задачи. Задача включает в себя создание шаблонов виртуальных машин с ОС Linux и определёнными типами процессоров для дальнейшей оценки совместимости программного обеспечения Intel® Math Kernel Library и инструкций данных процессоров. Оценка включает в себя запуск специфичных для процессоров тестовых наборов для определения корректности виртуализационного окружения. Также в задачу включены измерения производительности данного решения в сравнении с эмуляторами и аппаратными процессорами.

Требования к претенденту.

- Базовые знания в Linux,
- Базовые знания bash скриптов,
- Первоначальные познания в командной строке Linux,
- Первоначальные познания в средствах разработки под Linux (компилятор, утилита make),
- Опционально – знание Windows

Задача 19. Исследование влияния шаблонов доступа в память на производительность параллельных приложений

Описание задачи. Одной из критически важных для производительности характеристик приложений параллельной обработки данных является шаблон доступа в память. В общих чертах это соответствие между итерациями параллельного гнезда циклов и индексами доступа в массивы внутри гнезда. Это соответствие может сильно влиять на поведение кэш-памяти и, соответственно, производительность. При распределении итераций параллельного гнезда между потоками исполнения имеется шанс “поправить” это соответствие в пользу лучшей производительности. Задача состоит в (1) изучении понятия шаблона доступа и его формального представления (2) написании (или поиске в интернете) тестов, имеющих разные шаблоны доступа в память, но одинаковые по сложности и рабочему объёму данных и (3) нахождении экспериментальным способом влияния шаблонов доступа и распределения итераций по потокам на производительность.

Требования к претенденту.

- Хороший C/C++;
- Базовое понимание параллельной обработки данных.

Задача 20. Унификация тестов компоненты LAPACK для библиотеки Intel® MKL для процессоров Intel® Xeon® и Intel® Xeon Phi™

Описание задачи. Задача заключается в объединении тестовых баз компонента LAPACK для библиотеки Intel® Math Kernel Library для процессоров Intel® Xeon® и Intel® Xeon Phi™, а также оптимизации тестового покрытия для указанной компоненты. В рамках работы над проектом вы познакомитесь с новейшими подходами к организации и автоматизации тестирования сложного программного обеспечения.

Требования к претенденту.

- Математическое образование, опыт разработки для OS Linux;
- Владение языками C (желательно также Fortran).