



# *Анализ и оптимизация на софтуерни приложения*

Александър Пенев

Васил Василев

## Пример – Векторизация

# Съдържание

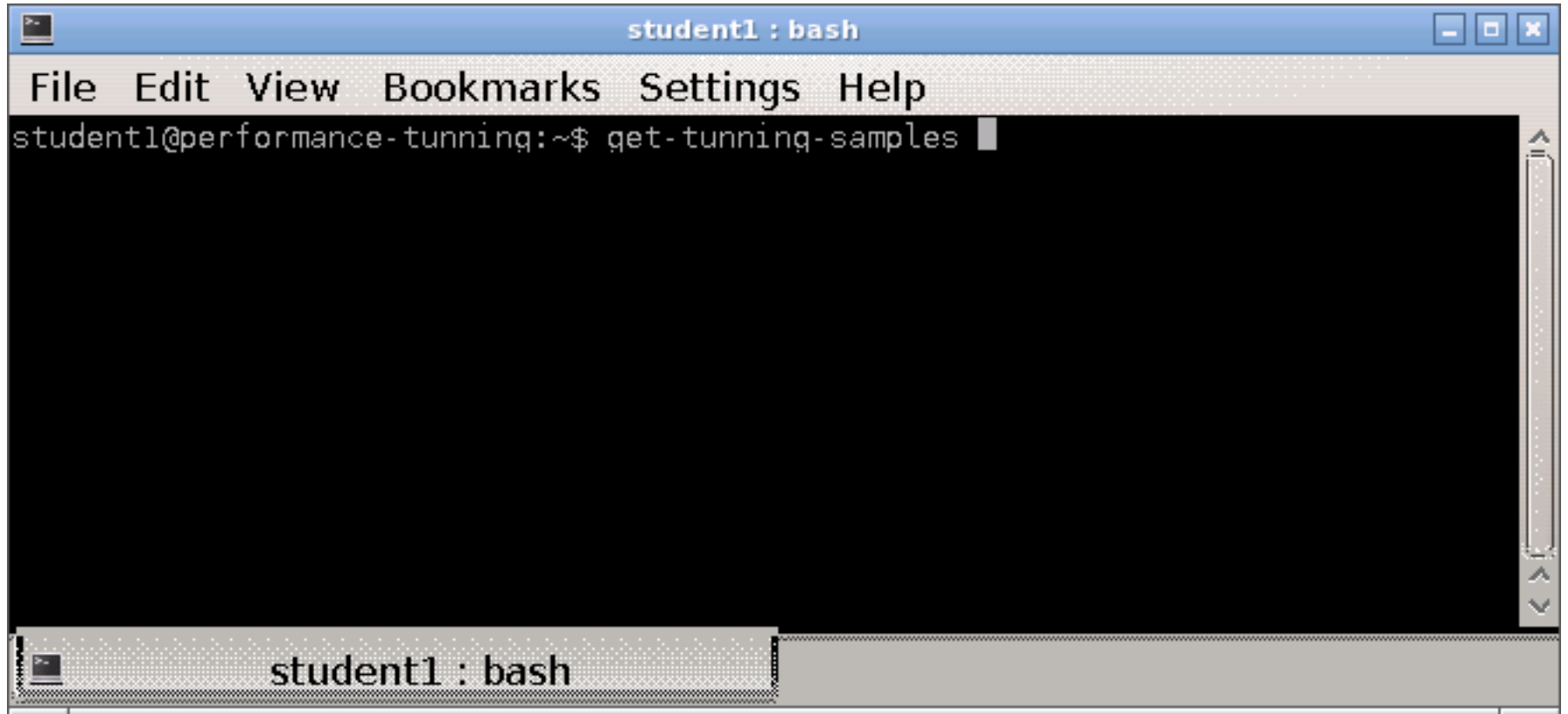
1. Сваляне на примерите
2. Компилиране
3. Ключове за задаване и контрол на автоматичната векторизация
4. Тестове на производителността на векторизацията
5. Флагове на компилаторите

# Как да свалите примерите във вашата работна среда

1. Свързвате се със сървъра по SSH и/или VNC, използвайки вашия акаунт
2. В SSH конзолата напишете **get-tuning-samples**
3. Получавате копие на сорс кода на примерите за упражненията във подпапка `samples` на вашата потребителска папка
4. Примерите в `~/samples/tuning/vectorization` са подходящи за експериментиране на векторизиращите възможности на компилаторите



# Допълнете примерите си



```
student1 : bash
File Edit View Bookmarks Settings Help
student1@performance-tuning:~$ get-tuning-samples
```

# *Нов пример vectorization/fixnoise*

- ❖ Реализиран на C
- ❖ Намира се в подпапката  
~/samples/tuning/vectorization/fixnoise
- ❖ Компилира се с:  
**make**



# Компиляция с различни компилатори (gcc)

1. От терминала изчистете старата компиляция с  
**make clean**
2. Компилиайте наново с  
**make gcc**
3. Разгледайте генерирания сорс код в `fixnoise.s`



# Компиляция с различни компилатори (icc)

1. От терминала изчистете старата компиляция с  
**make clean**
2. Компилиайте наново с  
**make icc**
3. Разгледайте генерирания сорс код в `fixnoise.s`



# Компиляция на *fixnoise1*

1. Влезте в директорията `fixnoise1`
2. От терминала изчистете старата компиляция с **`make clean`**
3. Компилиайте наново с **`make iss`** или **`make gcc`**
4. Разгледайте генерирания сорс код в `fixnoise.s`





# Флагове на компилаторите за векториз.

gcc:

1. **-ftree-vectorize** ВКЛЮЧВА ВЕКТОРИЗАЦИИТЕ
2. **-ftree-vectorizer-verbose=2** ВКЛЮЧВА ПОКАЗВАНЕТО НА УСПЕШНИТЕ И НЕУСПЕШНИ ВЕКТОРИЗАЦИИ

icc:

1. **-O2** и нагоре ВКЛЮЧВА И ВЕКТОРИЗАЦИИТЕ
2. **-vec** И **-no-vec** ВКЛЮЧВАТ И ИЗКЛЮЧВАТ ВЕКТОРИЗАЦИЯТА
3. **-vec-report3** ВКЛЮЧВА ПОКАЗВАНЕТО НА УСПЕШНИТЕ И НЕУСПЕШНИ ВЕКТОРИЗАЦИИ
4. и други



# Сравнение на производителността

1. Преработете `fixnoise.c` програмите, така че да изпълняват основната си функция върху повече данни (увеличете  $N$  поне 100 пъти). Изпълнявайте функцията в цикъл (поне  $10^9$  пъти)
2. Тествайте с `time`, `perf` или `VTune` производителността на двете програми
3. Използвайте `gcc` и `icc` и сравнете резултатите
4. Сравнете резултатите и със и без векторизация



# *Прилагане на векторизациите в Tachyon*

1. Използвайте векторизациите в проекта за оптимизиране на RayTracer-а Tachyon
2. Вижете къде може да се промени кода (ако има такива места) за да се приложат повече векторизации
3. Обмислете и експериментирайте с промяна на основните структури данни с цел по-добра векторизация

