Approximate Computing and Data Analysis

Jian-Jia Chen

TU Dortmund

Dec.,10,2019

Introduction

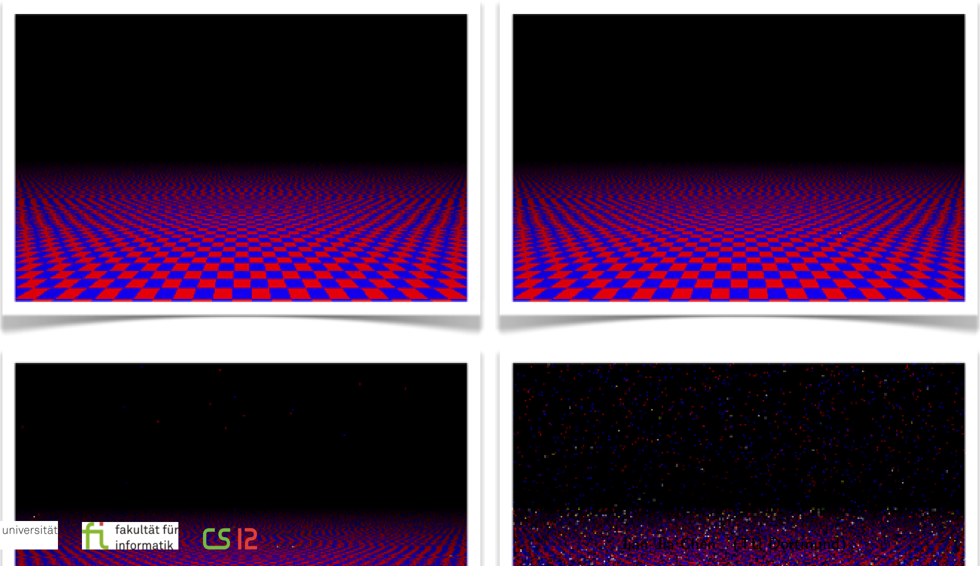
- Sometimes, computing the best possible output of some algorithm requires a significant amount of resources
- For some applications, the best possible output is not actually needed, since minor degradations will possibly not even be recognized by users.
- This can be exploited in a resource-constrained environment in order to trade-off the quality of the output against resources.
- A certain deviation of the actual output is accepted, for example, for lossy audio, video and image encoding.

Introduction

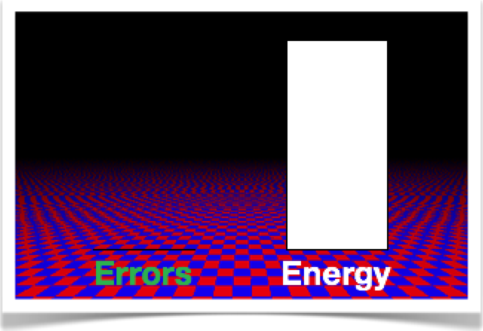
- Sometimes, computing the best possible output of some algorithm requires a significant amount of resources
- For some applications, the best possible output is not actually needed, since minor degradations will possibly not even be recognized by users.
- This can be exploited in a resource-constrained environment in order to trade-off the quality of the output against resources.
- A certain deviation of the actual output is accepted, for example, for lossy audio, video and image encoding.

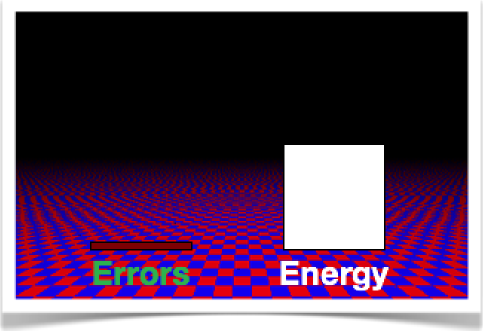
This leads us to consider approximate computing

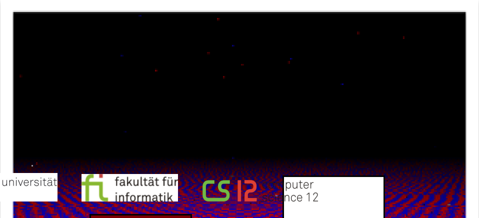
An Example

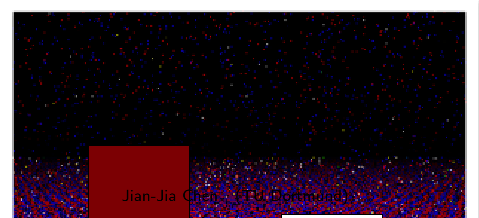


An Example

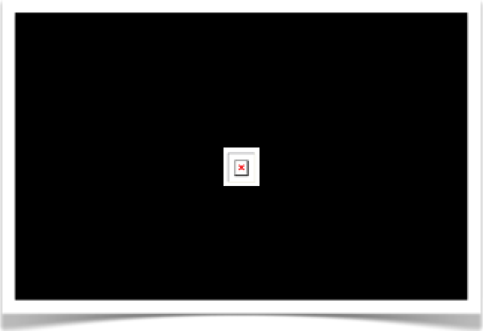


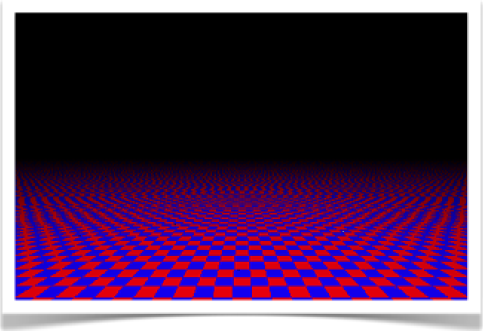


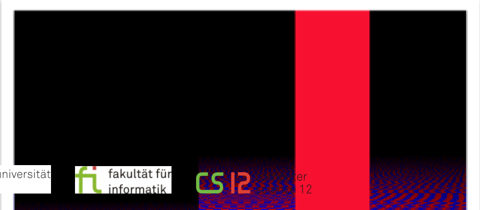




An Example







Definition

According to, Mittal, S.: "A survey of techniques for approximate computing." ACM Comput. Surv. 48(4), 62:1-62:33 (2016).

Definition

Computing which tolerates a certain deviation of generated output of some algorithm from the best possible result is called approximate computing

Definition

According to, Mittal, S.: "A survey of techniques for approximate computing." ACM Comput. Surv. 48(4), 62:1-62:33 (2016).

Definition

Computing which tolerates a certain deviation of generated output of some algorithm from the best possible result is called approximate computing

It is essential to compare the best possible output (real) values of $\vec{x} = \{x_1, x_2, \ldots, x_n\}$ with the approximated output (signal) values of $\vec{y} = \{y_1, y_2, \ldots, y_n\}$, for n samples.

Possible Metrics to Compare $\vec{x}$ and $\vec{y}$

Definition

The Mean-Squared Error (MSE) is defined as

$$MSE(\vec{x}, \vec{y}) = \frac{1}{n} \sum_{i=1}^{n} (x_i - y_i)^2$$

Possible Metrics to Compare $\vec{x}$ and $\vec{y}$

Definition

The Mean-Squared Error (MSE) is defined as

$$MSE(\vec{x}, \vec{y}) = \frac{1}{n} \sum_{i=1}^{n} (x_i - y_i)^2$$

Definition

The Root-Mean-Squared Error (RMSE) is defined as

$$RMSE(\vec{x}, \vec{y}) = \sqrt{\frac{1}{n} \sum_{i=1}^{n} (x_i - y_i)^2}$$

Possible Metrics to Compare $\vec{x}$ and $\vec{y}$ (cont.)

Definition

The Mean-Absolute Error (MAE) is defined as

$$MAE(\vec{x}, \vec{y}) = \frac{1}{n} \sum_{i=1}^{n} |x_i - y_i|$$

For identical deviations of the measured signal y from real values x, the MAE is equal to the RMSE. However, the RMSE emphasizes large deviations between real and measured values (so-called outliers).

Peak Signal to Noise Ratio

Definition

The Peak-Signal-to-Noise Ratio (PSNR) is defined as

$$PSNR(\vec{x}, \vec{y}) = 10 \log_{10} \left(\frac{x_{max}^2}{MSE(x, y)} \right) = 20 \log_{10} \left(\frac{x_{max}}{RMSE(x, y)} \right)$$

where x_{max} is defined as the $\max_{i=1}^{n} x_i$

Peak Signal to Noise Ratio

Definition

The Peak-Signal-to-Noise Ratio (PSNR) is defined as

$$PSNR(\vec{x}, \vec{y}) = 10 \log_{10} \left(\frac{x_{max}^2}{MSE(x, y)} \right) = 20 \log_{10} \left(\frac{x_{max}}{RMSE(x, y)} \right)$$

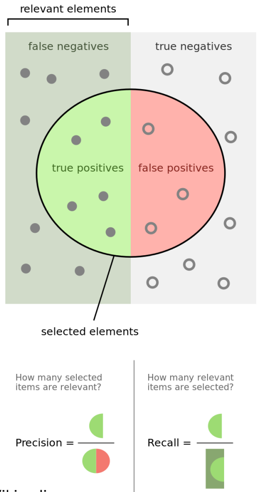
where x_{max} is defined as the $\max_{i=1}^{n} x_i$

- There are several other metrics, especially for images
- None of these metrics is really superior to others
- Several of these metrics should be computed and a careful comparison should be performed

Data Analysis in Approximating Computing

- For data analysis, classification of objects is a frequent goal
- Suppose that we restrict ourselves to binary classification
- Four cases are possible
  - True positives (TP): we classify some object as a cat and it is actually a cat
  - False positives (FP): we classify some object as a cat and it is not a cat
  - True negatives (TN): we classify some object as not a cat and it is actually not a cat
  - False negatives (FN): we classify some object as not a cat and it is actually a cat.

Precision and Recall



Wikipedia

Definition

The precision is defined as

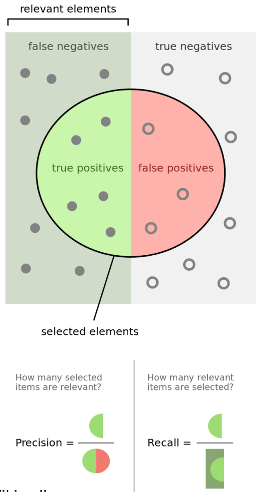
$$\frac{\text{True Positives}}{\text{True Positives} + \text{False Positives}}$$

Definition

The recall is defined as

$$\frac{\text{True Positives}}{\text{True Positives} + \text{False Negatives}}$$

Precision and Recall



Wikipedia

Definition

The precision is defined as

$$\frac{\text{True Positives}}{\text{True Positives} + \text{False Positives}}$$

Definition

The recall is defined as

$$\frac{\text{True Positives}}{\text{True Positives} + \text{False Negatives}}$$

Definition

The F1 score or F-measure is defined as the harmonic mean of precision and recall

Accuracy and Specificity

Definition

The accuracy is defined

$$\frac{\text{True Positives} + \text{True Negatives}}{\text{True Positives} + \text{False Positives} + \text{True Negatives} + \text{False Negatives}}$$

Definition

The specificity is defined

$$\frac{\text{True Negatives}}{\text{False Positives} + \text{True Negatives}}$$

Approximate Computing: Some Examples

- Qualifiers of data types
  - @approx int a := ...;
  - @precise int p := ...;
- Variable a is not accurate and variable p is accurate
- Statements
  - p := a; (this is problematic)
  - a := p; (this is okay)
- Approximate Conditions

  if (a = 10) { p :=2;} (this can be problematic, approximate bool)

Controlling Approximation

- Approximate should not interfere with precise
- Semantically, approximate results are unspecified best effort
- Only higher levels can measure quality, but application specific
- Lower (hardware or system software) levels can make monitoring convenient
- Offline: Profile, auto-tune
- Online: React, i.e., recompute or decrease the approximation level

Approximation-aware ISA

- An example (in MIPS ISA):

```
lw r1, 0x04($0)
lw r2, 0x08($0)
add r3, r1, r2
sw r3, 0x0c($0)
```

- An example (in Approximate MIPS ISA):

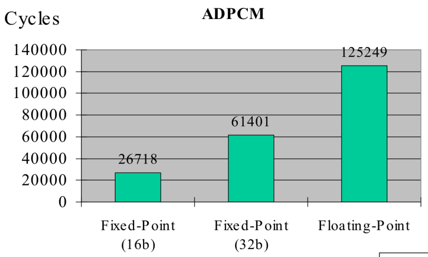
```
lw r1, 0x04($0)
lw r2, 0x08($0)
add.a r3, r1, r2
sw.a r3, 0x0c($0)
```

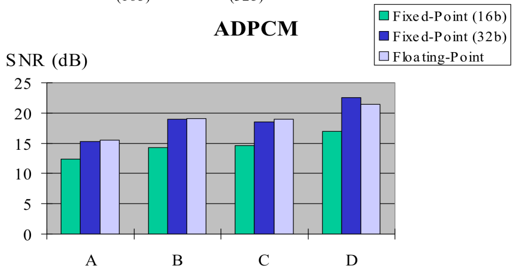
- add.a and sw.a need approximate ALU and approximate storage, respectively.

Floating-Point to Fixed-Point Conversion

- Pros:
  - Lower cost
  - Faster
  - Lower power consumption
  - Sufficient SNR, if properly used
  - Suitable for portable applications
- Cons:
  - Decreased dynamic range
  - Finite word-length effect, unless properly scaled
  - Overflow and excessive quantization noise
  - Extra programming effort

An Example: ADPCM





Ki-Il Kum, et al.