Triadic-Optimization

Ioannis Anagnostakis
rizitis@gmail.com

Jun 5, 2024

1 Introduction (Εισαγωγή)

We execute the `createnums.py` script, which generates a file named `numbers.txt` in the current directory. This file contains one billion lines, each containing a number in the range $[1, 1,000,000,000]$.

Εκτελούμε το script `createnums.py`, το οποίο δημιουργεί ένα αρχείο `numbers.txt` στον τρέχοντα κατάλογο. Αυτό το αρχείο περιέχει ένα δισεκατομμύριο γραμμές, κάθε μία με έναν αριθμό από το $[1, 1,000,000,000]$.

2 Processing Steps (Βήματα Επεξεργασίας)

Next, we run either `teliko_nums.py` or `teliko_nums2.py`. These scripts perform the following operations:

Στη συνέχεια, εκτελούμε είτε το `teliko_nums.py` είτε το `teliko_nums2.py`. Αυτά τα scripts εκτελούν τις ακόλουθες διαδικασίες:

1. Randomly shuffle the lines in `numbers.txt`. (Ανακατεύουν τυχαία τις γραμμές στο `numbers.txt`).
2. Count the total number of lines (10^9). (Μετρούν το συνολικό αριθμό γραμμών 10^9).
3. Randomly divide the lines into three segments. (Διαχωρίζουν τυχαία τις γραμμές σε τρία τμήματα).
4. Compute the mean for each segment: (Υπολογίζουν τον μέσο όρο για κάθε τμήμα:)

$$\mu_i = \frac{\sum_{j=1}^{N_i} x_j}{N_i}, \tag{1}$$

where N_i is the number of lines in segment i and x_j is the number in line j. (όπου N_i είναι ο αριθμός γραμμών στο τμήμα i και x_j είναι ο αριθμός στη γραμμή j).

3 Results (Αποτελέσματα)

The resulting means exhibit small variations:
Οι προκύπτοντες μέσοι όροι εμφανίζουν μικρές αποκλίσεις:

```
python3 teliko_nums.py
The mean of the first segment is: 499985344.58548886
The mean of the second segment is: 499997023.58038485
The mean of the third segment is: 500017633.33407336
```

or (ή)

```
python3 teliko_nums2.py
The mean of the first segment is: 499981295.966643
The mean of the second segment is: 500010723.68831867
The mean of the third segment is: 500007981.8450144
```

4 Conclusion (Συμπέρασμα)

At such large scales, randomness ensures the accuracy of the mean. This is significant because:
Σε τέτοιες μεγάλες κλίμακες, η τυχαιότητα εξασφαλίζει την ακρίβεια του μέσου όρου. Αυτό είναι σημαντικό επειδή:

- A correct mean indicates a representative sample from the dataset. (Ένας σωστός μέσος όρος δείχνει ένα αντιπροσωπευτικό δείγμα από το dataset).
- The dataset, originally one billion numbers, could instead be a collection of files (e.g., text files, images, etc.). (Το dataset, αρχικά ένα δισεκατομμύριο αριθμοί, θα μπορούσε να είναι μια συλλογή αρχείων όπως κείμενα, εικόνες κ.λπ.).
- Using this method, we can train a model with only $\frac{1}{3}$ of the dataset and allocate less than 10% for testing and validation. (Με αυτή τη μέθοδο, μπορούμε να εκπαιδεύσουμε ένα μοντέλο με μόνο το $\frac{1}{3}$ του dataset και να διαθέσουμε λιγότερο από 10% για δοκιμή και επικύρωση).

The methodology for selecting the 10% subset will be discussed separately. (Η μεθοδολογία για την επιλογή του 10% συνόλου θα συζητηθεί ξεχωριστά.)

5 Summary - Final Stage (Σύνοψη - Τελικό Στάδιο)

We execute the `createnums.py` script, which generates a file named `numbers.txt` in the current directory. This file contains one billion lines, each containing a number in the range $[1, 1,000,000,000]$.

Εκτελούμε το script `createnums.py`, το οποίο δημιουργεί ένα αρχείο `numbers.txt` στον τρέχοντα κατάλογο. Αυτό το αρχείο περιέχει ένα δισεκατομμύριο γραμμές, κάθε μία με έναν αριθμό από το $[1, 1,000,000,000]$.

6 Processing Steps (Βήματα Επεξεργασίας)

Next, we run either `teliko_nums.py` or `teliko_nums2.py`. These scripts perform the following operations:

Στη συνέχεια, εκτελούμε είτε το `teliko_nums.py` είτε το `teliko_nums2.py`. Αυτά τα scripts εκτελούν τις ακόλουθες διαδικασίες:

1. Randomly shuffle the lines in `numbers.txt`. (Ανακατεύουν τυχαία τις γραμμές στο `numbers.txt`).
2. Count the total number of lines (10^9). (Μετρούν το συνολικό αριθμό γραμμών 10^9).
3. Randomly divide the lines into three segments. (Διαχωρίζουν τυχαία τις γραμμές σε τρία τμήματα).
4. Compute the mean for each segment: (Υπολογίζουν τον μέσο όρο για κάθε τμήμα:)

$$\mu_i = \frac{\sum_{j=1}^{N_i} x_j}{N_i}, \tag{2}$$

where N_i is the number of lines in segment i and x_j is the number in line j. (όπου N_i είναι ο αριθμός γραμμών στο τμήμα i και x_j είναι ο αριθμός στη γραμμή j).

7 Dataset Processing (Επεξεργασία Συνόλου Δεδομένων)

Let's move on to practice now. (Καλή η θεωρεία, ας περάσουμε στην πράξη τώρα.)

Assume that the `all_data` folder represents the dataset with all the files we want to use to train the model. (Ας υποθέσουμε ότι ο φάκελος `all_data` είναι το dataset με όλα τα αρχεία που θέλουμε να εκπαιδεύσουμε το μοντέλο.)

Running `step-4teliko_arheia.py` will perform the following: (Εκτελώντας το `step-4teliko_arheia.py` θα γίνουν τα παρακάτω:)

1. Read the names of the files from the `all_data` folder. (Ανάγνωση των ονομάτων των αρχείων από τον φάκελο `all_data`)
2. Randomly shuffle the files. (Τυχαία ανάκατεμα των αρχείων)
3. Calculate the total number of files. (Υπολογισμός του συνολικού αριθμού αρχείων)
4. Calculate the size of each segment (33%). (Υπολογισμός του μεγέθους κάθε τμήματος (33%))
5. Create folders for the 3 segments. (Δημιουργία φακέλων για τα 3 τμήματα)
6. Separate the files and move them to the respective folders. (Διαχωρισμός των αρχείων και μεταφορά στους αντίστοιχους φακέλους.)

Next, we run `step-5create_test.py`, where the following actions take place: (Στη συνέχεια τρέχουμε το `step-5create_test.py`, όπου εκεί θα γίνουν τα παρακάτω:)

1. Create the folders `merged`, `test_data`, and `val_data`. (Θα δημιουργηθούν οι φάκελοι `merged`, `test_data` και `val_data`)
2. Merge the `data1` and `data3` folders into one folder named `merged`. (Οι φάκελοι `data1` και `data3` θα συγχωνευτούν σε ένα φάκελο `merged`.)
3. Count the files in the `merged` folder and randomly select one fifth for testing and another one fifth for validation. (Μετράμε τα αρχεία στον φάκελο `merged` και τυχαία διαλέγουμε το 1/5 για τεστ και άλλο 1/5 για επικύρωση.)

Now, we can train our model and evaluate if the **Triadic Optimization** approach is effective! (Αυτό ήταν! Μπορούμε τώρα να δοκιμάσουμε να εκπαιδεύσουμε το μοντέλο μας και να αξιολογήσουμε αν η **Triadic Optimization** αξίζει τον κόπο!)