A bird with a head

AI-generated content may be incorrect.

**Τεχνική Αναφορά**  
Επιχειρησιακή Έρευνα  
Αναστάσιος Βήττας (ΑΜ: 197)  
21 Απριλίου 2025

**Εισαγωγή**  
Το πρόβλημα ανάθεσης είναι ένα κλασικό πρόβλημα βελτιστοποίησης, στο οποίο έχουμε n εργασίες και n εργαζόμενους, και πρέπει να αντιστοιχίσουμε κάθε εργασία σε έναν μόνο εργαζόμενο, με τέτοιο τρόπο ώστε το συνολικό κόστος ανάθεσης να είναι το ελάχιστο δυνατό. Το κόστος κάθε πιθανής ανάθεσης δίνεται με έναν πίνακα τιμών. Πρόκειται για ένα πρόβλημα με πολλές εφαρμογές, από τη βιομηχανία και τις μεταφορές μέχρι το χρονοπρογραμματισμό.

**Dataset**  
Για την παρούσα εργασία χρησιμοποιήθηκαν δεδομένα από το OR-Library[[1]](#footnote-1), μια βάση δεδομένων για προβλήματα βελτιστοποίησης που περιέχει έτοιμα datasets και instances για το πρόβλημα ανάθεσης. Τα datasets που χρησιμοποιήθηκαν ονομάζονται assign\*.txt, όπου ο αριθμός δείχνει το πλήθος των εργασιών και των εργαζομένων. Για παράδειγμα, το αρχείο assign100.txt περιέχει πρόβλημα ανάθεσης με 100 εργασίες και 100 εργαζόμενους, ενώ το assign800.txt περιέχει πρόβλημα με 800×800 πίνακα κόστους.

Η δομή κάθε αρχείου είναι η εξής:

1. Πρώτη γραμμή: Περιέχει έναν αριθμό, ο οποίος δηλώνει πόσες εργασίες και εργαζόμενοι υπάρχουν στο στιγμιότυπο (π.χ. 100 για 100×100).
2. Υπόλοιπες Γραμμές: Ακολουθούν οι τιμές κόστους, μία για κάθε πιθανό συνδυασμό ανάθεσης εργασίας σε εργαζόμενο. Για παράδειγμα, η εργασία 0 έχει κόστος 52 αν ανατεθεί στον εργαζόμενο 0, 89 για τον εργαζόμενο 1, 40 για τον εργαζόμενο 2 κλπ.

A number grid with white text

AI-generated content may be incorrect.

**Ερώτημα 1 – Χρήση Μαθηματικής Μοντελοποίησης**

Στο πρώτο ερώτημα (erotima1.py) υλοποιήθηκε ένα μοντέλο που επιλύει το πρόβλημα ανάθεσης σύμφωνα με το πρωτότυπο που παρουσιάζεται στην σελίδα των Google OR-Tools[[2]](#footnote-2). Πιο συγκεκριμένα για την μοντελοποίηση χρησιμοποιήθηκε ο επιλυτής SCIP, ο οποίος υποστηρίζει ακέραιο και γραμμικό προγραμματισμό.

Το πρόβλημα διατυπώθηκε ως εξής:

* Κάθε εργασία μπορεί να ανατεθεί σε έναν μόνο εργαζόμενο
* Κάθε εργαζόμενος μπορεί να αναλάβει μόνο μία εργασία.

Η συνάρτηση στόχου του μοντέλου αθροίζει τα κόστη ανάθεσης για κάθε ζευγάρι εργασίας i και εργαζομένου j, όπου υπάρχει ανάθεση, και επιδιώκει την ελαχιστοποίησή τους. Σκοπός είναι να βρεθεί η κατανομή των εργασιών στους εργαζόμενους που δίνει το ελάχιστο συνολικό κόστος.

A diagram of a network

AI-generated content may be incorrect.

Η λύση που προκύπτει είναι βέλτιστη με βάση τον δεδομένο πίνακα κόστους, και καταγράφεται σε αρχείο assign\*\_erotima1\_solution.txt.

Τα αποτελέσματα του erotima1.py είναι τα εξής:

|  |  |  |
| --- | --- | --- |
| **Μέγεθος Προβλήματος** | **Συνολικό Κόστος** | **Χρόνος Επίλυσης (δευτ.)** |
| assign100.txt | 305 | 0.19 |
| assign200.txt | 475 | 0.98 |
| assign300.txt | 626 | 2.92 |
| assign400.txt | 804 | 5.99 |
| assign500.txt | 991 | 10.32 |
| assign600.txt | 1176 | 17.44 |
| assign700.txt | 1362 | 24.00 |
| assign800.txt | 1552 | 36.26 |

A screen shot of a computer screen

AI-generated content may be incorrect.

**Ερώτημα 2 – Σύγκριση με τον Ουγγρικό Αλγόριθμο**

Στη συνέχεια, συνέκρινα τα αποτελέσματα του πρώτου ερωτήματος (Μαθηματική Μοντελοποίηση) με τον Ουγγρικό αλγόριθμο, μέσα από τη βιβλιοθήκη NetworkX[[3]](#footnote-3) και πιο συγκεκριμένα τη συνάρτηση minimum\_weight\_full\_matching. Θα μπορούσα να υλοποιήσω επίσης τον αλγόριθμο με την βιβλιοθήκη που βρήκα η οποία ονομάζεται Munkres[[4]](#footnote-4).

Για να αποτυπώσω αυτές τις διαφορές, κατέγραψα τους χρόνους εκτέλεσης και το κόστος της λύσης, δημιουργώντας παράλληλα γραφήματα με την βιβλιοθήκη Matplotlib. Αυτά τα γραφήματα δείχνουν πολύ καλύτερα από τους πίνακες την διαφορά τον δυο αλγορίθμων.

|  |  |  |
| --- | --- | --- |
| **Μέγεθος Προβλήματος** | **OR-Tools (sec)** | **Hungarian (sec)** |
| assign100.txt | 0.19 | 0.20 |
| assign200.txt | 1.00 | 0.02 |
| assign300.txt | 2.80 | 0.06 |
| assign400.txt | 6.88 | 0.13 |
| assign500.txt | 11.66 | 0.18 |
| assign600.txt | 20.05 | 0.28 |
| assign700.txt | 25.78 | 0.37 |
| assign800.txt | 37.88 | 0.47 |

A screen shot of a computer program

AI-generated content may be incorrect.

A graph with a line and a dotted line

AI-generated content may be incorrect.A graph with a line and a point

AI-generated content may be incorrect.

Από τα παραπάνω φαίνεται ότι ο Ουγγρικός αλγόριθμος είναι πολύ ταχύτερος για μεγάλα προβλήματα, ειδικά από 200 εργασίες και πάνω. Και οι δύο μέθοδοι δίνουν το ίδιο κόστος, δηλαδή βρίσκουν την ίδια βέλτιστη λύση, όμως ο Ουγγρικός είναι πιο αποδοτικός χρονικά σε μεγάλα δεδομένα.

**Ερώτημα 3 – Προσθήκη Νέων Περιορισμών**

Στο τρίτο ερώτημα προστέθηκε ένας επιπλέον περιορισμός: για κάθε ομάδα 5 συνεχόμενων εργασιών, τουλάχιστον 2 από αυτές πρέπει να ανατεθούν σε εργαζόμενους της αντίστοιχης ομάδας. Αυτό έκανε το πρόβλημα πιο σύνθετο και το μοντέλο λύθηκε με Integer Programming μέσω SCIP (μέσα από OR-Tools).

Τα αποτελέσματα του Ερώτηματος 3 παρουσιάζονται στον πίνακα που ακολουθεί και αποθηκεύονται ως assign\*\_erotima3\_solution.txt.

|  |  |  |
| --- | --- | --- |
| **Μέγεθος Προβλήματος** | **Συνολικό Κόστος** | **Χρόνος Επίλυσης (δευτ.)** |
| assign100.txt | 551.0 | 0.17 |
| assign200.txt | 1063.0 | 0.82 |
| assign300.txt | 1296.0 | 2.40 |
| assign400.txt | 1872.0 | 5.16 |
| assign500.txt | 2092.0 | 9.05 |
| assign600.txt | 2906.0 | 23.07 |
| assign700.txt | 3163.0 | 22.12 |
| assign800.txt | 3601.0 | 56.92 |

**A screen shot of a computer

AI-generated content may be incorrect.**

Η προσθήκη του περιορισμού αύξησε σημαντικά το συνολικό κόστος ανάθεσης, σχεδόν στο διπλάσιο. Παρ’ όλα αυτά, ο χρόνος επίλυσης παρέμεινε σε αποδεκτά επίπεδα. Αυτό δείχνει ότι το μοντέλο είναι εφαρμόσιμο ακόμα και σε πιο πολύπλοκες εκδοχές του προβλήματος.

**Συμπεράσματα**

Η μαθηματική μοντελοποίηση με την χρήση των εργαλείων OR-Tools λύνει το πρόβλημα με ακρίβεια αλλά είναι πιο αργή σε μεγάλα δεδομένα σε αντίθεση με τον Ουγγρικό αλγόριθμο που είναι πολύ πιο γρήγορος. Το μοντέλο με περιορισμό ανάθεσης σε ομάδες έχει δικαίως πιο αυξημένο κόστος καθώς οι περιορισμοί «δυσκολεύουν» τον επιλυτή.

**Πόροι Συστήματος**

CPU: AMD Ryzen 7 7730U

RAM: 16GB unified DDR4

**Οδηγίες Εκτέλεσης Κώδικα**

Προαπαιτούμενα:

Python >= 3.8

git clone <https://github.com/tasosvittas/Operations-Research.git>

Δημιουργία και ενεργοποίηση περιβάλλοντος:

Windows:

python -m venv opSearch

opSearch\Scripts\activate

Linux / macos:

python3 -m venv opSearch

source opSearch/bin/activate

Εγκατάσταση βιβλιοθηκών:

pip install -r requirements.txt

Εκτέλεση των ερωτημάτων:

cd Operations-Research\Ergasia1\_OS

python erotima1.py  
python erotima2.py  
python erotima3.py

1. <https://people.brunel.ac.uk/~mastjjb/jeb/orlib/files/> [↑](#footnote-ref-1)
2. https://developers.google.com/optimization/assignment/assignment\_example [↑](#footnote-ref-2)
3. <https://networkx.org/documentation/stable/reference/algorithms/generated/networkx.algorithms.bipartite.matching.minimum_weight_full_matching.html> [↑](#footnote-ref-3)
4. https://software.clapper.org/munkres/ [↑](#footnote-ref-4)