**Τεχνική Αναφορά**  
**Μάθημα:** Επιχειρησιακή Έρευνα  
**Φοιτητής:** Αναστάσιος Βήττας (ΑΜ: 197)  
**Ημερομηνία:** Απρίλιος 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 γνωστή βάση δεδομένων για προβλήματα βελτιστοποίησης που περιέχει έτοιμα, καλά διαμορφωμένα στιγμιότυπα (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. **Στη συνέχεια:** Ακολουθούν οι τιμές κόστους, μία για κάθε πιθανό συνδυασμό ανάθεσης (εργασία σε εργαζόμενο). Το σύνολο των τιμών είναι n×nn \times nn×n και είναι παρατεταγμένες **γραμμή-γραμμή**, δηλαδή πρώτα για την εργασία 0, μετά για την εργασία 1, κ.ο.κ.

A number grid with white text

AI-generated content may be incorrect.

Για παράδειγμα, η εργασία 0 έχει κόστος 52 αν ανατεθεί στον εργαζόμενο 0, 89 για τον εργαζόμενο 1, 40 για τον εργαζόμενο 2 κλπ.

**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, και
* 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(x)\_erotima1\_solution.txt.

Τα αποτελέσματα του erotima1.py είναι τα εξής:

A screen shot of a computer

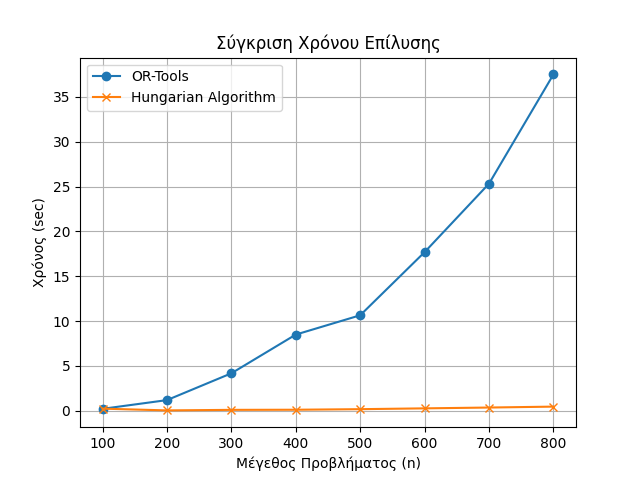
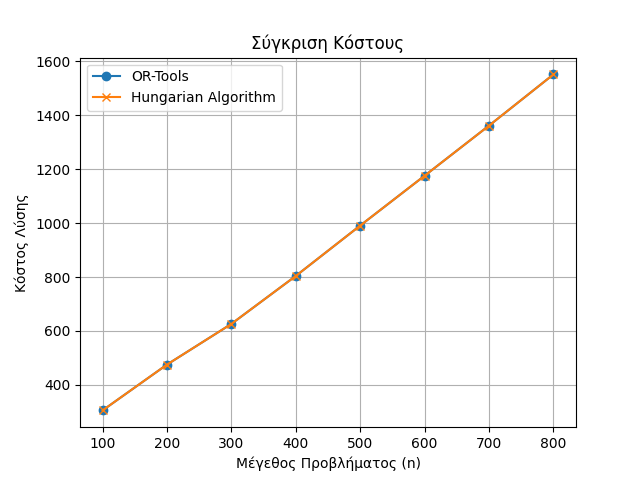
AI-generated content may be incorrect.

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

Στη συνέχεια, συνέκρινα τη μαθηματική μοντελοποίηση με τον Ουγγρικό αλγόριθμο, όπως αυτός είναι διαθέσιμος μέσα από τη βιβλιοθήκη NetworkX (συγκεκριμένα τη συνάρτηση minimum\_weight\_full\_matching). Και στις δύο περιπτώσεις, το κόστος ήταν ουσιαστικά το ίδιο ή πολύ κοντά στο βέλτιστο, αλλά η ταχύτητα διαφέρει, ειδικά για τα μικρότερα μεγέθη: ο Ουγγρικός αλγόριθμος τείνει να είναι ταχύτερος σε μικρά προβλήματα. Ωστόσο, η μαθηματική μοντελοποίηση έχει ευελιξία για να ενσωματώσει πιο περίπλοκους περιορισμούς, κάτι που ο Ουγγρικός αλγόριθμος δεν επιτρέπει τόσο εύκολα.

Για να αποτυπώσω αυτές τις διαφορές, κατέγραψα τους χρόνους εκτέλεσης και το κόστος της λύσης, δημιουργώντας παράλληλα γραφήματα με τη Matplotlib. Αυτά τα γραφήματα δείχνουν με σαφήνεια πώς κλιμακώνεται ο χρόνος σε σχέση με το πλήθος των εργασιών.

A screen shot of a computer program

AI-generated content may be incorrect.

**Ερώτημα 3 – Προσθήκη Νέων Περιορισμών**  
Τέλος, επεξεργάστηκα μια παραλλαγή του προβλήματος, όπου ορίστηκε ότι για κάθε ομάδα πέντε διαδοχικών εργασιών (π.χ. εργασίες με αναγνωριστικά 0 έως 4), τουλάχιστον δύο από αυτές πρέπει να ανατίθενται σε εργαζομένους της ίδιας πεντάδας (π.χ. εργαζόμενοι 0 έως 4). Για να ενσωματώσω αυτόν τον περιορισμό, τροποποίησα τη μαθηματική μοντελοποίηση, προσθέτοντας κατάλληλες ανισώσεις στο μοντέλο.

Η επιπλέον αυτή συνθήκη οδήγησε σε αύξηση της πολυπλοκότητας του προβλήματος, γεγονός που γίνεται αντιληπτό από τους αυξημένους χρόνους επίλυσης. Παρ’ όλα αυτά, οι solvers (SCIP/CBC) κατάφεραν να δώσουν βέλτιστες ή εφικτές λύσεις (ανάλογα με το μέγεθος του προβλήματος) μέσα σε ανεκτούς χρόνους.

**Συμπεράσματα**  
Μελετώντας τα αποτελέσματα, συμπεραίνω ότι η μαθηματική μοντελοποίηση με OR-Tools είναι μια στιβαρή προσέγγιση που λύνει αποτελεσματικά το βασικό πρόβλημα ανάθεσης, και μπορεί να επεκταθεί με πρόσθετους περιορισμούς όταν αυτό απαιτείται από το εκάστοτε σενάριο. Από την άλλη, ο Ουγγρικός αλγόριθμος είναι γρήγορος και ακριβής για την κλασική μορφή του προβλήματος, αλλά δεν προσαρμόζεται εύκολα σε νέους σύνθετους περιορισμούς.

Συνολικά, η επιλογή της μεθόδου επίλυσης εξαρτάται από τις απαιτήσεις του προβλήματος. Σε περίπτωση που χρειαζόμαστε μεγάλη ευελιξία για την ενσωμάτωση πολύπλοκων συνθηκών, προτιμάται η μαθηματική μοντελοποίηση. Αντίθετα, εάν θέλουμε γρήγορες λύσεις σε τυπικά προβλήματα ανάθεσης χωρίς ιδιαίτερους περιορισμούς, ο Ουγγρικός αλγόριθμος αποτελεί εξαιρετική επιλογή.

**Οδηγίες Εκτέλεσης Κώδικα**

1. Αρχικά, τοποθετούμε όλα τα αρχεία εισόδου (assign100.txt κ.λπ.) μέσα σε έναν φάκελο π.χ. dataset/.
2. Για το πρώτο ερώτημα, εκτελούμε το αρχείο erotima1.py χρησιμοποιώντας την εντολή:  
   python erotima1.py  
   Τα αποτελέσματα αποθηκεύονται στον φάκελο solutions/erotima1.
3. Για το δεύτερο ερώτημα, εκτελούμε το erotima2.py:  
   python erotima2.py  
   Τα γραφήματα κόστους και χρόνου δημιουργούνται και αποθηκεύονται στον φάκελο comparison\_img/.
4. Για το τρίτο ερώτημα, εκτελούμε το erotima3.py:  
   python erotima3.py  
   Οι λύσεις της παραλλαγής του προβλήματος αποθηκεύονται στον φάκελο solutions/erotima3.

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)