Το πρόβλημα του Knuth

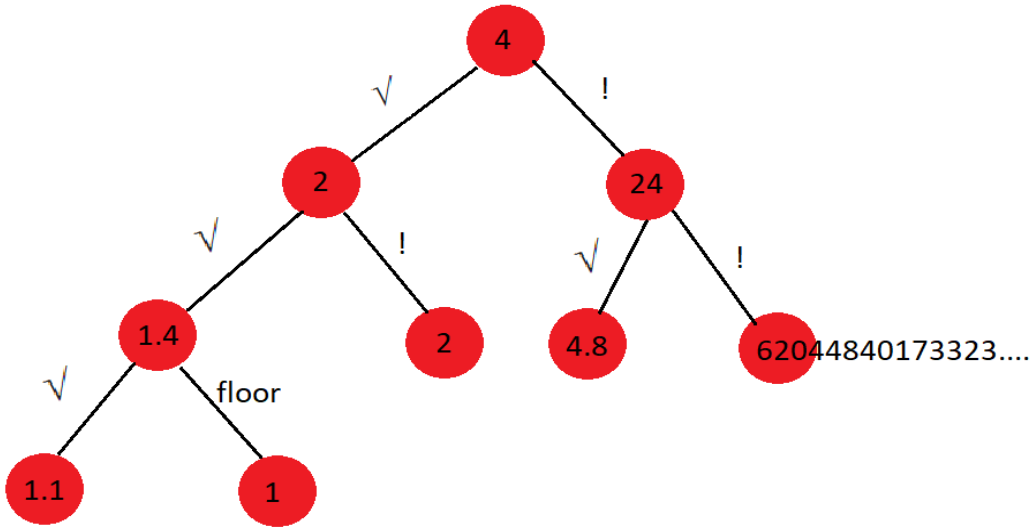
Το πρόβλημα του Knuth (Knuth, 1964) ορίζεται ως εξής: Ξεκινώντας με τον αριθμό 4, μια ακολουθία πράξεων εύρεσης τετραγωνικής ρίζας, ακέραιου μέρους (floor) και παραγοντικών μπορεί να φτάσει σε οποιονδήποτε επιθυμητό θετικό ακέραιο. Για παράδειγμα, μπορούμε να φτάσουμε στον αριθμό 5 με την παρακάτω ακολουθία πράξεων:

$$\left\lfloor \sqrt{\sqrt{\sqrt{\sqrt{\sqrt{(4!)!}}}} \right\rfloor = 5$$

Επίλυση:

Για την επίλυση του προβλήματος χρησιμοποιήθηκε η γλώσσα Python όπου υλοποιήθηκαν οι αλγόριθμοι πρώτα σε πλάτος (BFS) και επαναληπτική εκβάθυνση (IDDFS) σε δέντρο αναζήτησης. Στο δέντρο αναζήτησης ο αριστερός κόμβος ενός γονέα είναι η τετραγωνική ρίζα του αριθμού και ο δεξιός το παραγοντικό, αν ο αριθμός είναι ακέραιος. Σε περίπτωση που είναι δεκαδικός, ο δεξιός κόμβος είναι η πράξη floor.

Παρακάτω φαίνονται μερικοί από τους πρώτους κόμβους του δέντρου αναζήτησης



Ως δομή δεδομένων για το δέντρο χρησιμοποιήθηκε μια λίστα frontier όπου αξιοποιείται ως ουρά για τον BFS και ως στοίβα για τον IDDFS. Κάθε στοιχείο της λίστας frontier (δηλαδή κάθε κόμβος του δέντρου) είναι ένα dictionary με κλειδιά: την τιμή του αριθμού(value) , την πράξη που έδωσε αυτόν τον αριθμό (action) και την τιμή του προηγούμενου κόμβου (prev) . Επιπλέον, στον IDDFS υπάρχει ακόμα ένα κλειδί (depth) που δείχνει το βάθος του

κόμβου από τη ρίζα του δέντρου, με την ρίζα να έχει τιμή 0. Το βάθος στον BFS βρίσκεται ευκολότερα με βάση το πλήθος των actions .

Λόγω της πολυπλοκότητας των αλγορίθμων και της φύσης του προβλήματος έγιναν οι παρακάτω βελτιστοποιήσεις:

- Ο μέγιστος αριθμός που μπορεί να υπολογιστεί για το παραγοντικό είναι ο **2147483647.** Αυτός είναι περιορισμός που θέτει το math module της Python και δεν υπάρχει νόημα να τον παρακάμψουμε με τρίτα εργαλεία αφού είναι υπολογιστικά ασύμφορο. Οποιοσδήποτε μεγαλύτερος αριθμός πρέπει να υπολογιστεί θα πάρει την τιμή None . Γενικότερα όλες οι τιμές None αγνοούνται κατά τον έλεγχο για επέκταση ή λύση.
- Υπάρχει η σταθερά **FACTORIAL_MAX** που δεν επιτρέπει να υπολογιστεί παραγοντικό μεγαλύτερο αυτού του αριθμού. Ομοίως, οποιοσδήποτε μεγαλύτερος αριθμός, παίρνει την τιμή None με αποτέλεσμα να μην επεκτείνονται αρκετά κλαδιά του δέντρου για εξοικονόμηση. Αν οριστεί η σταθερά ως None ισχύει ο παραπάνω περιορισμός.
- Η τετραγωνική ρίζα ενός αριθμού μικρότερου του 1 δεν υπολογίζεται επειδή οι επόμενες 2 πιθανές πράξεις (ρίζα ή floor) θα δώσουν ως αποτέλεσμα το 0 ή ένα νέο δεκαδικό αριθμό μικρότερο του 1 αντίστοιχα. Το 0 δεν ανήκει σε κανέναν αριθμό-στόχο και ένας νέος δεκαδικός θα είναι και αυτός < 1 οπότε θα καταλήξει τελικά σε ατέρμονα βρόχο. Ομοίως, και εδώ κάθε τιμή < 1 παίρνει τιμή None.
- Στον BFS κάθε κόμβος ελέγχεται επιπλέον για τον αν είναι ο στόχος μόλις προστίθεται στο frontier

Το πρόγραμμα:

Τρέχοντας το αρχείο knuth.py το πρόγραμμα ζητάει από τον χρήστη τον αριθμό στόχο και τον αλγόριθμο που θα χρησιμοποιηθεί.

```
Enter the integer you want to search: 120
Choose the search algorithm
        1.breadth-first
        2.iterative deepening
>
```

Έπειτα εμφανίζονται τα παρακάτω αποτελέσματα αν βρεθεί λύση

```
PROBLEM SOLVED!!
Initial value (root):  4
Depth:  9
Actions:
         factorial
         factorial
         root
         root
         root
         root
         root
         floor
         factorial
Goal:  120
5.6005332469940186 s
```

Αν υπάρχουν πολλές τιμές που δεν έχουν υπολογιστεί (δηλαδή None) υπάρχει η περίπτωση να αδειάσει το frontier και να μην βρεθεί λύση. Τότε εμφανίζεται το παρακάτω μήνυμα

```
No solution found. Try increasing FACTORIAL_MAX
```

Για το testing του προγράμματος δοκιμάστηκαν 19 τυχαίοι θετικοί ακέραιοι . Η δοκιμή έγινε σε υπολογιστή με επεξεργαστή AMD Ryzen 7 3800X και 32GB RAM. Η έκδοση της Python ήταν η 3.11. Ως ανώτατο χρονικό όριο για την λύση ενός προβλήματος είναι τα 2 λεπτά. Αν ο αλγόριθμος δεν έδωσε καμία απάντηση πέραν των 2 λεπτών ή εμφανίστηκε το μήνυμα No solution found τότε δεν υπάρχουν μετρήσεις. Παρακάτω είναι τα αποτελέσματα των δοκιμών. Όλες οι τιμές είναι σε δευτερόλεπτα.

Αποτελέσματα:

FACTORIAL_MAX	None		5.000.000		500.000		
GOAL	BFS	IDDFS	BFS	IDDFS	BFS	IDDFS	DEPTH
1	0	0	0	0	0	0	3
3	5.62	-	5.6	100.03	0.003	0.008	12
5	5.6	27.81	5.5	27.96	0.0009	0.0009	8
8	-	-	-	-	1.37	-	31
10	5.6	77.8	5.4	78.02	0.002	0.003	11
12	-	-	-	-	-	-	-
16	-	-	-	-	-	-	-
20	-	-	-	-	7.03	-	62
22	-	-	-	-	6.7	-	56
24	0.0009	0	0	0	0	0	1
25	-	-	-	-	5.5	-	43
30	5.57	16.76	5.6	16.8	0.0009	0.0009	7
36	-	-	-	-	4.2	-	41
46	-	-	78.65	-	0.4	8	22
81	-	-	-	-	-	-	-
120	5.6	39.05	5.6	39	0.002	0.001	9
633	-	-	-	-	4.2	-	42
720	62.18	-	62	-	0.006	0.042	14
942	0.01	5.59	0.001	5.59	0.001	0	6

Κάθε αριθμός-στόχος δοκιμάστηκε για τρείς διαφορετικές τιμές του FACTORIAL_MAX . Όσο μικρότερο είναι το FACTORIAL_MAX τόσο περισσότεροι κόμβοι του δέντρου δεν επεκτείνονται άρα υπάρχει μεγαλύτερη πιθανότητα να βρεθεί λύση σε συντομότερο χρόνο. Πράγματι, όταν FACTORIAL_MAX = 500.000 οι 9 από τους 19 αριθμούς βρέθηκαν σε σχεδόν μηδενικό χρόνο, και από τους 2 αλγορίθμους, με το μέγιστο βάθος να είναι 14. Όσοι αριθμοί σε αυτήν την κατηγορία έχουν χρόνο πάνω από 1 δευτερόλεπτο (συνολικά 6) είναι και η μοναδική μέτρηση που υπήρξε αφού με μεγαλύτερο FACTORIAL_MAX δεν βρέθηκε λύση.

Επιπλέον, για μόλις 3 αριθμούς (12, 16, 81) δεν υπήρξε καμία λύση από κανέναν αλγόριθμο και για τα τρία όρια του FACTORIAL_MAX . Αυτό θα μπορούσε να σημαίνει ότι για να βρει ένας αλγόριθμος τον αριθμό, θα πρέπει να περάσει από κόμβους με τιμή μεγαλύτερη του FACTORIAL_MAX , κάτι που δεν επιτρέπεται. Επομένως το όριο που υπάρχει στον υπολογισμό του παραγοντικού μπορεί μεν να δώσει πιο γρήγορα μια λύση όμως ταυτόχρονα περιορίζεται και το σύνολο των αριθμών που μπορούν να βρεθούν.

Επίσης, είναι προφανές πως ο IDDFS χρειάζεται τουλάχιστον 3 φορές περισσότερο χρόνο να φτάσει στον αριθμό-στόχο για τον λόγο ότι συνεχώς επανεκκινεί την αναζήτηση μόλις φτάσει στο μέγιστο βάθος. Αξίζει να σημειωθεί επίσης πως το μέγιστο βάθος όπου ο IDDFS βρήκε λύση είναι 22 (GOAL=46) και μάλιστα μόνο με τον μεγαλύτερο περιορισμό του παραγοντικού.

Τέλος, αξίζει να σημειωθεί πως όταν το FACTORIAL_MAX μειώθηκε στο 5.000.000 δεν υπήρξε σχεδόν καμία αλλαγή στους χρόνους σε σύγκριση με το «απεριόριστο» παραγοντικό.