ΕΛΛΗΝΙΚΗ ΔΗΜΟΚΡΑΤΙΑ

Εθνικό και Καποδιστριακό Πανεπιστήμιο Αθηνών

Λογικός Προγραμματισμός

Ασκήσεις

Παναγιώτης Σταματόπουλος

Τμήμα Πληροφορικής και Τηλεπικοινωνιών

Περιεχόμενα

1. Β΄ Ομάδα Ασκήσεων "Λογικού Προγραμματισμού" Ακαδημαϊκού Έτους 2012-13

Οι ασκήσεις της ομάδας αυτής πρέπει να αντιμετωπισθούν με τη βοήθεια της τεχνολογίας του **προγραμματισμού με περιορισμούς**. Ένα σύστημα λογικού προγραμματισμού που υποστηρίζει την τεχνολογία αυτή είναι η **ECL^{i}PS^e** (**http://www.eclipseclp.org**). Μπορείτε να χρησιμοποιήσετε είτε την παλιότερη βιβλιοθήκη **`fd`** είτε την νεώτερη **`ic`**. Η βιβλιοθήκη **`fd`** τεκμηριώνεται στο κεφάλαιο 2 του "Obsolete Libraries Manual" και η **`ic`** στα κεφάλαια 3 και 4 του "Constraint Library Manual". Αν χρησιμοποιήσετε την **`ic`**, θα σας χρειαστεί και η βιβλιοθήκη **`branch_and_bound`**, ιδιαιτέρως το κατηγόρημα `bb_min/3`, το οποίο τεκμηριώνεται, όπως και όλα τα κατηγορήματα που παρέχει η **ECL^{i}PS^e**, στο "Alphabetical Predicate Index".

Εναλλακτικά συστήματα λογικού προγραμματισμού με περιορισμούς που μπορείτε να χρησιμοποιήσετε για τις ασκήσεις αυτής της ομάδας είναι η **GNU Prolog** (**http://www.gprolog.org**) και η **SWI-Prolog** (**http://www.swi-prolog.org**).

Σε κάθε περίπτωση, στα αρχεία που θα παραδώσετε, θα πρέπει να αναφέρεται στην αρχή τους, σαν σχόλιο, για ποιο σύστημα Prolog έχουν υλοποιηθεί τα αντίστοιχα προγράμματα.

1.1 Άσκηση 4

Μία εκδοχή του προβλήματος διαμέρισης αριθμών είναι η εξής. Δεδομένου κάποιου θετικού ακεραίου N, να διαμερισθεί το σύνολο $S = \{1,2,3,\ldots,N\}$ σε δύο υποσύνολα S_1 και S_2 ($S_1 \cap S_2 = \emptyset$, $S_1 \cup S_2 = S$) τέτοια ώστε τα S_1 και S_2 να έχουν ίδιο πλήθος στοιχείων ($|S_1| = |S_2|$), το άθροισμα των στοιχείων του S_1 να ισούται με το άθροισμα των στοιχείων του S_2 ($\sum_{i \in S_1} i = \sum_{j \in S_2} j$) και το άθροισμα των τετραγώνων των στοιχείων του S_1 να ισούται με το άθροισμα των τετραγώνων των στοιχείων του S_2 ($\sum_{i \in S_1} i^2 = \sum_{j \in S_2} j^2$). Για παράδειγμα, για $N = 8$, το πρόβλημα έχει μία λύση, την $S_1 = \{1,4,6,7\}$, $S_2 = \{2,3,5,8\}$. Είναι προφανές ότι το πρόβλημα δεν έχει λύση αν το N είναι περιττός. Επίσης, φαίνεται ότι υπάρχουν λύσεις μόνο για N που είναι πολλαπλάσια του 4 και μεγαλύτερα ή ίσα του 8, αλλά κάτι τέτοιο δεν έχει αποδειχθεί μαθηματικά.

Ορίστε στο σύστημα λογικού προγραμματισμού της επιλογής σας ένα κατηγόρημα `numpart/3`, το ποίο όταν καλείται σαν `numpart(N,L1,L2)`, για δεδομένο N, να επιστρέφει στα L1 και L2, μία διαμέριση του συνόλου {1,2,3,…,N}, σύμφωνα με τον παραπάνω ορισμό, και τελικά, μέσω οπισθοδρόμησης, όλες τις διαφορετικές λύσεις. Κάποια παραδείγματα εκτέλεσης είναι τα εξής:

```
?- numpart(2,L1,L2).

no


?- numpart(4,L1,L2).

no


?- numpart(8,L1,L2).

L1 = [1, 4, 6, 7]

L2 = [2, 3, 5, 8]  --> ;

no
```

```
?- numpart(9,L1,L2).
no

?- numpart(10,L1,L2).
no

?- numpart(12,L1,L2).
L1 = [1, 3, 7, 8, 9, 11]
L2 = [2, 4, 5, 6, 10, 12]  --> ;
no

?- numpart(16,L1,L2).
L1 = [1, 2, 7, 8, 11, 12, 13, 14]
L2 = [3, 4, 5, 6, 9, 10, 15, 16]  --> ;

L1 = [1, 3, 6, 8, 10, 12, 13, 15]
L2 = [2, 4, 5, 7, 9, 11, 14, 16]  --> ;

L1 = [1, 3, 6, 9, 10, 11, 12, 16]
L2 = [2, 4, 5, 7, 8, 13, 14, 15]  --> ;

L1 = [1, 4, 5, 8, 10, 11, 14, 15]
L2 = [2, 3, 6, 7, 9, 12, 13, 16]  --> ;

L1 = [1, 4, 6, 7, 9, 12, 14, 15]
L2 = [2, 3, 5, 8, 10, 11, 13, 16]  --> ;

L1 = [1, 4, 6, 7, 10, 11, 13, 16]
L2 = [2, 3, 5, 8, 9, 12, 14, 15]  --> ;
```

```
L1 = [1, 5, 6, 7, 8, 11, 14, 16]

L2 = [2, 3, 4, 9, 10, 12, 13, 15]  --> ;

no


?- findall((L1,L2),numpart(20,L1,L2),L), length(L,N).

.........

N = 24


?- findall((L1,L2),numpart(24,L1,L2),L), length(L,N).

.........

N = 296




?- findall((L1,L2),numpart(28,L1,L2),L), length(L,N).

.........

N = 1443


?- findall((L1,L2),numpart(32,L1,L2),L), length(L,N).

.........

N = 17444
```

Παραδοτέο για την άσκηση είναι **ένα πηγαίο αρχείο Prolog**.

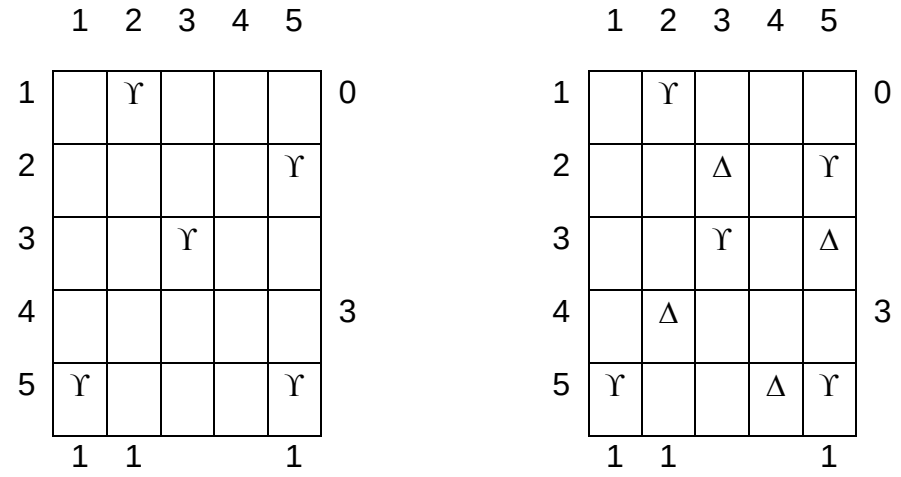
1.2 Άσκηση 5

Θεωρούμε έναν ορθογώνιο αγρό δεδομένων διαστάσεων, έστω NxM. Σε συγκεκριμένες θέσεις (i,j) του αγρού, με το i από 1 έως N και το j από 1 έως M, υπάρχουν K δέντρα. Θέλουμε να τοποθετήσουμε στον αγρό ένα πλήθος από τέντες σε θέσεις που πρέπει να βρεθούν έτσι ώστε:

- Σε τουλάχιστον μία από τις γειτονικές θέσεις κάθε δέντρου, οριζόντια, κάθετα ή διαγώνια, να υπάρχει τέντα.
- Δύο τέντες δεν πρέπει να βρίσκονται σε γειτονικές θέσεις, ούτε οριζόντια, ούτε κάθετα, ούτε διαγώνια.
- Δεν μπορεί να υπάρχει τέντα σε θέση που υπάρχει δέντρο.

- Για κάποιες, όχι απαραίτητα όλες, από τις γραμμές και τις στήλες του αγρού δίνονται μέγιστοι αριθμοί τεντών που μπορεί να υπάρχουν στη γραμμή ή τη στήλη, αντίστοιχα.
- Οι τέντες που θα τοποθετηθούν να είναι οι ελάχιστες δυνατές.

Παρακάτω, στο σχήμα αριστερά, φαίνεται ένας αγρός με $N = 5$ και $M = 5$, στον οποίο βρίσκονται $K = 5$ δέντρα, που σημειώνονται με το σύμβολο Υ, στις θέσεις $(1,2)$, $(2,5)$, $(3,3)$, $(5,1)$ και $(5,5)$. Επίσης, δίνονται οι περιορισμοί ότι στην 1η και στην 4η γραμμή πρέπει να υπάρχουν έως 0 και έως 3 τέντες, αντίστοιχα, και ότι σε καθεμία από την 1η, την 2η και την 5η στήλη πρέπει να υπάρχει έως 1 τέντα. Στο σχήμα δεξιά φαίνεται μία λύση του προβλήματος, με τον ελάχιστο δυνατό αριθμό τεντών, οι θέσεις των οποίων σημειώνονται με το σύμβολο Δ.



Γράψτε ένα κατηγόρημα tents/4, το οποίο να καλείται σαν `tents(RowTents, ColumnTents, Trees, Tents)`, όπου `RowTents` είναι μία λίστα με τα επιθυμητά συνολικά μέγιστα πλήθη τεντών ανά γραμμή, `ColumnTents` είναι μία λίστα με τα επιθυμητά συνολικά μέγιστα πλήθη τεντών ανά στήλη και Trees είναι μία λίστα από συντεταγμένες της μορφής Row-Column, στις οποίες βρίσκονται τα δέντρα. Το πλήθος των γραμμών N του αγρού ισούται με το μήκος της λίστας `RowTents`, το πλήθος των στηλών M ισούται με το μήκος της λίστας `ColumnTents` και το πλήθος των δέντρων K ισούται με το μήκος της λίστας Trees. Αν δεν θέλουμε να δώσουμε περιορισμό μεγίστου πλήθους για τις τέντες σε κάποια γραμμή ή κάποια στήλη, αρκεί στο αντίστοιχο στοιχείο της λίστας `RowTents` ή `ColumnTents`, αντίστοιχα, να βάλουμε κάποιο αρνητικό αριθμό. Το κατηγόρημα που θα γράψετε να επιστρέφει στη μεταβλητή Tents μία λίστα με τις συντεταγμένες των θέσεων στις οποίες πρέπει να τοποθετηθούν οι τέντες, ώστε να ισχύουν οι περιορισμοί που έχουν τεθεί. **Αν το πρόβλημα έχει περισσότερες από μία λύση με το ελάχιστο δυνατό πλήθος τεντών, να βρίσκονται όλες μέσω οπισθοδρόμησης.** Κάποιες ενδεικτικές εκτελέσεις είναι οι εξής:

```
?- tents([0, -1, -1, 3, -1], [1, 1, -1, -1, 1],
         [1-2, 2-5, 3-3, 5-1, 5-5], Tents).
Tents = [2 - 3, 3 - 5, 5 - 2, 5 - 4]  --> ;
Tents = [2 - 3, 3 - 5, 4 - 2, 5 - 4]  --> ;
Tents = [2 - 3, 3 - 5, 4 - 1, 5 - 4]  --> ;
Tents = [2 - 2, 3 - 5, 4 - 1, 5 - 4]  --> ;
.......................

?- findall(Tents, tents([0, -1, -1, 3, -1], [1, 1, -1, -1, 1],
           [1-2, 2-5, 3-3, 5-1, 5-5], Tents), AllTents).
AllTents = [[2 - 3, 3 - 5, 5 - 2, 5 - 4],
```

```
            [2 - 3, 3 - 5, 4 - 2, 5 - 4],
            [2 - 3, 3 - 5, 4 - 1, 5 - 4],
            [2 - 2, 3 - 5, 4 - 1, 5 - 4],
            [2 - 2, 3 - 4, 4 - 1, 5 - 4],
            [2 - 2, 2 - 4, 4 - 1, 5 - 4],
            [2 - 2, 2 - 4, 4 - 1, 4 - 5],
            [2 - 2, 2 - 4, 4 - 1, 4 - 4],
            [2 - 1, 3 - 5, 4 - 2, 5 - 4],
            [2 - 1, 3 - 4, 5 - 2, 5 - 4],
            [2 - 1, 3 - 4, 4 - 2, 5 - 4],
            [2 - 1, 2 - 4, 5 - 2, 5 - 4],
            [2 - 1, 2 - 4, 4 - 5, 5 - 2],
            [2 - 1, 2 - 4, 4 - 4, 5 - 2],
            [2 - 1, 2 - 4, 4 - 2, 5 - 4],
            [2 - 1, 2 - 4, 4 - 2, 4 - 5],
            [2 - 1, 2 - 4, 4 - 2, 4 - 4]]

?- tents([-1, -1, -1, 2, -1, -1, 2, 1],
         [2, 1, -1, 1, 1, -1, 1, -1, -1, 1, 2, -1],
         [1-4, 1-9, 1-12, 2-1, 2-5, 2-8, 3-1, 3-6, 3-8, 3-12,
          4-5, 4-7, 4-11, 5-3, 5-9, 6-1, 6-7, 6-11, 7-5,
          8-10], Tents).
Tents = [2 - 4, 2 - 9, 2 - 12, 3 - 2, 4 - 6, 5 - 10, 6 - 3,
         7 - 1, 7 - 6, 8 - 11]  --> ;
Tents = [2 - 4, 2 - 9, 2 - 12, 3 - 2, 4 - 6, 5 - 10, 6 - 3,
         7 - 1, 7 - 6, 8 - 9]  --> ;
Tents = [2 - 4, 2 - 9, 2 - 12, 3 - 2, 4 - 6, 5 - 10, 6 - 3,
         6 - 6, 7 - 1, 8 - 11]  --> ;
.......................

?- findall(Tents, tents([-1, -1, -1, 2, -1, -1, 2, 1],
           [2, 1, -1, 1, 1, -1, 1, -1, -1, 1, 2, -1],
           [1-4, 1-9, 1-12, 2-1, 2-5, 2-8, 3-1, 3-6, 3-8,
            3-12, 4-5, 4-7, 4-11, 5-3, 5-9, 6-1, 6-7, 6-11,
            7-5, 8-10], Tents), AllTents),
   length(AllTents, N).
AllTents = .......................
N = 1262

?- tents([1, -1, -1, -1, -1, -1, 3, 1, -1, 1, 2, 2, 1],
         [2, 1, -1, 1, 4, 1, 3, -1, 2, 1, 1, 0],
         [2-3, 1-5, 5-4, 4-5, 7-7, 10-6, 2-2, 4-8, 8-5, 9-9,
          1-8, 9-2, 3-3, 1-1, 9-8, 8-7, 10-10, 2-7, 8-6, 4-4,
          9-1], Tents).
Tents = [2 - 1, 2 - 4, 2 - 9, 3 - 7, 5 - 5, 8 - 8, 9 - 6,
         10 - 2, 11 - 11]  --> ;
.......................

[1]?- tents([1, -1, -1, -1, -1, -1, 3, 1, -1, 1, 2, 2, -1],
           [2, 1, -1, 1, 4, -1, 3, -1, 2, 1, 1, -1],
           [2-3, 1-5, 5-4, 4-5, 7-7, 10-6, 2-2, 4-8, 8-5, 9-9,
            1-8, 9-2, 3-3, 1-1, 9-8, 8-7, 10-10, 2-7, 8-6,
            4-4, 9-1, 11-4, 11-8, 12-5, 12-9, 12-12, 6-11,
            9-11, 6-12], Tents).
Tents = [2 - 1, 2 - 4, 2 - 9, 3 - 7, 5 - 5, 7 - 12, 8 - 8,
        9 - 6, 9 - 12, 10 - 2, 11 - 5, 11 - 11, 12 - 8]  --> ;
```

[1] Η ερώτηση αυτή είναι πιθανό να αργεί πάρα πολύ.

. . . . . .

Παραδοτέο για την άσκηση είναι **ένα πηγαίο αρχείο Prolog**.

1.3 Άσκηση 6

Αντιμετωπίστε τις προηγούμενες ασκήσεις της Β΄ ομάδας (4 και 5) με τη βοήθεια της βιβλιοθήκης C++ για προγραμματισμό με περιορισμούς, τον Naxos Solver (http://www.di.uoa.gr/~pothitos/naxos/), που έχει σχεδιάσει και αναπτύξει ο υποψήφιος διδάκτωρ του Τμήματός μας **Νίκος Ποθητός** (pothitos@di.uoa.gr).

Παραδοτέο για την άσκηση είναι **ένα αρχείο zip**, που θα περιλαμβάνει όλη τη δουλειά σας για τα δύο προβλήματα, κατάλληλα οργανωμένη σε καταλόγους ανά πρόβλημα. Μην περιλάβετε μέσα στο αρχείο zip εκτελέσιμα ή αντικειμενικά προγράμματα.

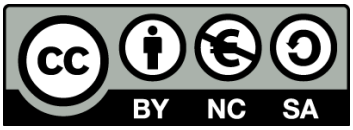
Σημειώματα

Σημείωμα Ιστορικού Εκδόσεων Έργου

Το παρόν έργο αποτελεί την έκδοση 1.0.

Σημείωμα Αναφοράς

Σημείωμα Αδειοδότησης



[1] http://creativecommons.org/licenses/by-nc-sa/4.0/

Ως **Μη Εμπορική** ορίζεται η χρήση:

- που δεν περιλαμβάνει άμεσο ή έμμεσο οικονομικό όφελος από την χρήση του έργου, για το διανομέα του έργου και αδειοδόχο
- που δεν περιλαμβάνει οικονομική συναλλαγή ως προϋπόθεση για τη χρήση ή πρόσβαση στο έργο
- που δεν προσπορίζει στο διανομέα του έργου και αδειοδόχο έμμεσο οικονομικό όφελος (π.χ. διαφημίσεις) από την προβολή του έργου σε διαδικτυακό τόπο

Ο δικαιούχος μπορεί να παρέχει στον αδειοδόχο ξεχωριστή άδεια να χρησιμοποιεί το έργο για εμπορική χρήση, εφόσον αυτό του ζητηθεί.

Διατήρηση Σημειωμάτων

- Οποιαδήποτε αναπαραγωγή ή διασκευή του υλικού θα πρέπει να συμπεριλαμβάνει:
- το Σημείωμα Αναφοράς
- το Σημείωμα Αδειοδότησης
- τη δήλωση Διατήρησης Σημειωμάτων

- το Σημείωμα Χρήσης Έργων Τρίτων (εφόσον υπάρχει)

μαζί με τους συνοδευόμενους υπερσυνδέσμους.

Σημείωμα Χρήσης Έργων Τρίτων

Το Έργο αυτό κάνει χρήση των ακόλουθων έργων:

Εικόνες/Σχήματα/Διαγράμματα/Φωτογραφίες

Χρηματοδότηση

- Το παρόν εκπαιδευτικό υλικό έχει αναπτυχθεί στο πλαίσιο του εκπαιδευτικού έργου του διδάσκοντα.
- Το έργο «**Ανοικτά Ακαδημαϊκά Μαθήματα στο Πανεπιστήμιο Αθηνών**» έχει χρηματοδοτήσει μόνο τη αναδιαμόρφωση του εκπαιδευτικού υλικού.
- Το έργο υλοποιείται στο πλαίσιο του Επιχειρησιακού Προγράμματος «Εκπαίδευση και Δια Βίου Μάθηση» και συγχρηματοδοτείται από την Ευρωπαϊκή Ένωση (Ευρωπαϊκό Κοινωνικό Ταμείο) και από εθνικούς πόρους.