AI Freecell Solitair Solver

Όνομα: Χρυσοχοΐδης Ελευθέριος

Α/Μ: 7013

Το project AI Freecel Solitair Solver αναπτύχθηκε στα πλαίσια του μαθήματος της Τεχνητής Νοημοσύνης. Είναι φτιαγμένο σε C# χρησιμοποιώντας το Visual Studio 2015.

Η λειτουργικότητα του project είναι να προσπαθεί να βρει λύση σε παρτίδες του παιχνιδιού Freecell Solitair. Η αναζήτηση λύσης μπορεί να γίνει χρησιμοποιώντας έναν εκ των 4 παρακάτω αλγορίθμων: Πρώτα σε πλάτος, Πρώτα σε βάθος, Πρώτα στον καλύτερο, και Α άστρο.

Οι αλγόριθμοι αυτοί εξετάζουν διαδοχικές καταστάσεις του παιχνιδιού, μέχρι να βρουν λύση. Η διαφορές στους αλγορίθμους είναι στον τρόπο που επιλέγουν την επόμενη προς εξέταση κατάσταση.

Ο αλγόριθμος πρώτα σε πλάτος, επιλέγει να εξετάσει πρώτα όλα τα αδέλφια μιας κατάστασης, και μετά όλα τα παιδιά της κάθε κατάστασης. Μόλις τελειώσει με τα παιδιά εκείνα, προχωράει στα επόμενα παιδιά αλλά μένει σε συγκεκριμένο επίπεδο. Για να κατέβει επίπεδο δηλαδή, πρέπει να εξετάσει όλες τις αδελφικές καταστάσεις.

Ο αλγόριθμος πρώτα σε βάθος ψάχνει διαδοχικά όλους τους απογόνους μιας κατάστασης μέχρι αυτοί να εξαντληθούν. Όταν εξαντληθούν πηγαίνει στον πρόγονο και ψάχνει εκεινού τους απογόνους αντίστοιχα.

Ο αλγόριθμος πρώτα στον καλύτερο βαθμολογεί όλα τα παιδιά μιας κατάστασης και επισκέπτεται αυτήν που έχει την καλύτερη βαθμολογία. Η βαθμολόγηση γίνεται με βάση μια ευρετική συνάρτηση.

Ο αλγόριθμος Α άστρο, βαθμολογεί κι αυτός τα παιδιά μιας κατάστασης προσθέτοντας όμως και το κόστος για την επίσκεψη. Χρησιμοποιείται και εδώ ευρετική συνάρτηση.

Για να τρέξει η εφαρμογή μου, μέσω γραμμής εντολών χρησιμοποιήστε εντολή του τύπου "AI Freecell Solitair Solver.exe" <method> <inputFileName> <outputFileName> όπου:

* <method>, μπορείτε να βάλετε breadth, depth, best, astar που είναι για την επιλογή αλγορίθμου αναζήτησης,
* < inputFileName>, το όνομα του αρχείου που θα περιέχει την αρχική κατάσταση, και
* <outputFileName>, το αρχείου που θα περιέχει την λύση (αν βρεθεί).

Παράδειγμα εκτέλεσης:

"AI Freecell Solitair Solver.exe" best test1.txt test1SolBest.txt

Η αναζήτηση θα τελειώσει είτε όταν βρεθεί λύση είτε όταν περάσουν 600 δευτερόλεπτα. Ο χρόνος αυτός ορίζεται στην κλάση Αuxiliary με όνομα μεταβλητής TIMEOUT.

ΑΠΟΤΕΛΕΣΜΑΤΑ ΕΚΤΕΛΕΣΗΣ ΠΕΙΡΑΜΑΤΩΝ



Στο παραπάνω αρχείο φαίνονται τα αποτελέσματα των πειραμάτων. Τα πειράματα έγιναν σε 10 ίδια αρχεία (test1.txt – test10.txt) τα οποία δημιουργήθηκαν από τον Generator που δινόταν, με χρήση και των 4 αλγορίθμων στο κάθε αρχείο.

Τα αποτελέσματα γενικά ήταν εν μέρει αναμενόμενα. Κάποια από αυτά ήταν αναμενόμενα αλλά κάποια άλλα όχι.

* Πιο συγκεκριμένα για τον αλγόριθμο πρώτα σε πλάτος: Στις 10 προσπάθειες που έγιναν στα 10 αρχεία, δεν κατάφερε ποτέ να βρεθεί λύση σε 600 δευτερόλεπτα (10 λεπτά). Ίσως σε βάθος χρόνου να μπορούσε να λυθεί το πρόβλημα αλλά στο ορισμένο αυτό χρονικό διάστημα δεν βρέθηκε καμία φορά λύση. Στις περισσότερες περιπτώσεις εξετάστηκαν πάνω από 20 χιλιάδες κόμβοι (καταστάσεις) χωρίς επιτυχία. **Αποτέλεσμα**: εν μέρει αναμενόμενο. Ο αλγόριθμος πρώτα σε πλάτος είναι σίγουρο ότι αν υπάρχει λύση, θα την βρει, αλλά θα έχει ψάξει πιο μπροστά όλες τις υπόλοιπες λύσεις που είναι στο ίδιο ή σε υψηλότερο επίπεδο. Λογικά σε περισσότερο χρόνο θα έβρισκε λύση.
* Για τον αλγόριθμο πρώτα σε βάθος: Γενικά νομίζω ότι η τύχη ήταν με το μέρος μας. Στις 10 δοκιμές βρέθηκε λύση στις 6 από αυτές. Με ποσοστό επιτυχίας 60% καταφέρνει να βρει λύση στο παιχνίδι και σχετικά γρήγορα κιόλας. Οι περισσότερες λύσεις είναι κάτω από τα 15 δευτερόλεπτα κάτι το οποίο είναι πολύ ικανοποιητικό. **Αποτέλεσμα**: εν μέρει αναμενόμενο. Ο αλγόριθμος πρώτα σε βάθος είναι γενικά γρήγορος, και εξαντλεί μια κίνηση μέχρι το τέλος της, οπότε είναι λίγο θέμα τύχης αν βρεθεί λύση ή όχι και σε πόσο χρόνο θα βρεθεί.
* Για τον αλγόριθμο πρώτα στον καλύτερο: Νομίζω η μεγαλύτερη αποτυχία. Μπορεί να υπάρχει θέμα στην υλοποίηση του αλγορίθμου στο project, γιατί τα αποτελέσματα του δεν είναι καλά. Στα 10 πειράματα δεν βρήκε λύση σε κανένα από αυτά. Θεωρητικά θα έπρεπε να βρίσκει λύση και σχετικά γρήγορα γιατί από τις πιθανές επόμενες καταστάσεις επιλέγει να πάει σε αυτήν με την μεγαλύτερη βαθμολογία (βάση της ευρετικής συνάρτησης). Κάτι τέτοιο σημαίνει ότι θα έβρισκε την λύση σχετικά εύκολα. Στα 10 πειράματα όμως δεν βρήκε λύση καμία απολύτως φορά! Ο χρόνος είναι υπέρ αρκετός για αυτόν τον αλγόριθμο, οπότε τα αποτελέσματα του είναι δεν είναι καλά. **Αποτέλεσμα**: μη αναμενόμενο. Ίσως φταίει η ευρετική συνάρτηση. Θεωρητικά θα έπρεπε να βρει κάποιες λύσεις από τις 10 δοκιμές.
* Για τον αλγόριθμο Α άστρο: Ο πιο αποδοτικός αλγόριθμος (βάση των αποτελεσμάτων του). Βρήκε λύση στις 7 από τις 10 δοκιμές και μάλιστα στις περισσότερες από αυτές την βρήκε μέσα σε λιγότερο από 1 δευτερόλεπτο. **Αποτέλεσμα**: αναμενόμενο. Ο Α άστρο είναι αλγόριθμος που χρησιμοποιεί κι αυτός ευρετική συνάρτηση προσθέτοντας όμως ένα κόστος κάθε φορά. Το αξιοπερίεργο είναι ότι παρόλο που χρησιμοποιείται η ίδια ευρετική συνάρτηση όπως και στον πρώτα στον καλύτερο (με την διαφορά ότι εδώ προσθέτει το κόστος) ο Α άστρο βρίσκει λύσεις ενώ ο πρώτα στον καλύτερο δεν βρίσκει καμία.

Στον κώδικα υπάρχουν σχόλια για κάθε μέθοδο που χρησιμοποιείται, όσο το δυνατόν πιο επεξηγηματικά. Επίσης υπάρχουν οδηγίες εκτέλεσης της εφαρμογής, αν αυτή δεν εκτελεστεί σωστά.