ΤΕΧΝΗΤΗ ΝΟΗΜΟΣΥΝΗ & ΕΜΠΕΙΡΑ ΣΥΣΤΗΜΑΤΑ

ΕΡΓΑΣΙΑ ΣΤΟΥΣ ΓΕΝΕΤΙΚΟΥΣ ΑΛΓΟΡΙΘΜΟΥΣ

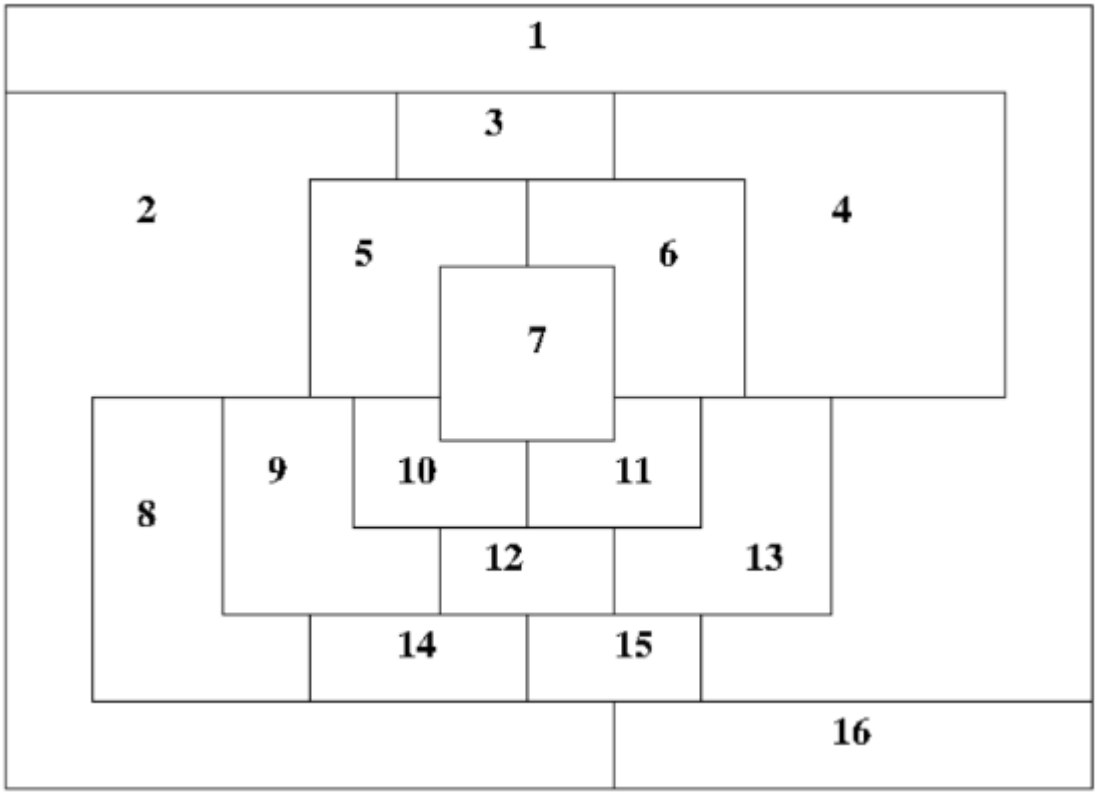
ΓΕΩΡΓΙΟΣ ΣΕΪΜΕΝΗΣ – Π19204
ΠΑΝΕΠΙΣΤΗΜΙΟ ΠΕΙΡΑΙΩΣ – ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ | ΜΑΪΟΣ ~ ΙΟΥΝΙΟΣ 2022

Περιεχόμενα

Εκφώνηση

Ως εργασία, ζητήθηκε να βρεθεί λύση στην άσκηση τύπου «Constraint Satisfaction Problem», με τη χρήση γενετικών αλγορίθμων.

Η άσκηση είχε τον παρακάτω γράφο με διάφορα κουτάκια να ενώνονται με τυχαία σειρά. **Τα κουτάκια πρέπει να χρωματιστούν με τρόπο τέτοιον, ώστε κανένα κουτάκι να έχει ίδιο χρώμα με αυτά που γειτονεύει.**



Οι φοιτητές έπρεπε να:

- Χρησιμοποιήσουν γενετικούς αλγορίθμους.
- Ορίσουν αρχικό πληθυσμό με τυχαίο αριθμό πλήθους.
- Φτιάξουν συνάρτηση καταλληλότητας.
- Φτιάξουν συνάρτηση επιλογής γονέων και συνάρτηση διασταύρωσης ενός σημείου.
- Φτιάξουν συνάρτηση μετάλλαξης και ανανέωσης πληθυσμού.

Υλοποίηση σε C#

Η λύση της εργασίας έγινε σε εφαρμογή τύπου Windows Forms, ώστε να υπάρχει γραφικοποίηση στον χρωματισμό του γράφου.

Το συγκεκριμένο πρόβλημα CSP (Constraint Satisfaction Problem) έχει πολλές βέλτιστες λύσεις, οπότε ο στόχος μας είναι να φτάσουμε σε μία από αυτές. Στην κύρια ροή, το πρόγραμμα παράγει έναν αρχικό πληθυσμό, μέσα στον οποίο υπάρχουν χρωμοσώματα με τυχαίες λύσεις. Χρησιμοποιώντας την συνάρτηση καταλληλότητας, γίνεται ταξινόμηση των χρωμοσωμάτων, κρατώντας, ως πρώτο, το καλύτερο χρωμόσωμα (δηλαδή αυτό που είναι πιο κοντά σε μία βέλτιστη λύση). Αυτό που καταφέρνουμε, κρατώντας το χρωμόσωμα αυτό, είναι ότι το πρόγραμμα παράγει κι άλλες λύσεις, βασιζόμενο σε αυτό. Έτσι, το πρόγραμμα φτάνει γρηγορότερα σε μία βέλτιστη λύση, χρησιμοποιώντας ταυτόχρονα μετάλλαξη, σε περίπτωση που κωλύεται. **Το πρόγραμμα δεν σταματάει, μέχρι να βρει μία βέλτιστη λύση.**

Πώς αναπαρίστανται οι λύσεις;

Οι λύσεις πρέπει να αναπαρασταθούν σε δυαδική μορφή. Από τη στιγμή, λοιπόν, που έχουμε **τέσσερα πιθανά χρώματα,** κατανείμονται ως εξής:

- 00 → μπλε
- 01 → κόκκινο
- 10 → πράσινο
- 11 → κίτρινο

Έπειτα, έχουμε 16 κουτάκια να χρωματίσουμε. Αυτό σημαίνει ότι έχουμε 16 × 2 = 32 bits για να ορίσουμε μία λύση. Οπότε, μία ομάδα των δύο bit καθορίζει και τη θέση κάθε κουτιού. Δηλαδή η πρώτη δυάδα καθορίζει το χρώμα του πρώτου κουτιού, η δεύτερη δυάδα του δεύτερου κουτιού, και ούτω καθεξής. Για παράδειγμα, η λύση `00000000000000000000000000000000` σημαίνει ότι όλα τα κουτάκια έχουν το χρώμα μπλε. Ωστόσο, η λύση `11010101010101010101010101010101` δίνει στο πρώτο κουτάκι το χρώμα κίτρινο και σε όλα τα υπόλοιπα το χρώμα κόκκινο.

Πώς λειτουργεί η συνάρτηση καταλληλότητας;

Χρησιμοποιώντας την κλάση Genome, αναπαρίσταται ένα χρωμόσωμα. Το χρωμόσωμα έχει δύο ιδιότητες: την λύση (όπως αυτή επεξηγήθηκε παραπάνω) και το σκορ καταλληλότητας. Η λύση έχει τύπο λίστας αλφαριθμητικών και **το σκορ καταλληλότητας είναι ένας ακέραιος αριθμός. Αυτός ο αριθμός ανεβαίνει κατά 1, όταν ένα κουτάκι έχει διαφορετικό χρώμα με κάποιον γείτονά του**. Άρα, όταν όλα τα κουτάκια έχουν το ίδιο χρώμα, το σκορ καταλληλότητας είναι 0. Αν όλα τα κουτάκια έχουν διαφορετικά χρώματα με τους γείτονές του, ο αριθμός φτάνει στο μέγιστο, που είναι το 42. **Οπότε, μία βέλτιστη λύση θα πρέπει να έχει σκορ καταλληλότητας 42.**

Πως ορίζεται ο αρχικός πληθυσμός;

Ο καλύτερος αριθμός, που ταιριάζει στον αρχικό πληθυσμό, είναι 15 χρωμοσώματα. Βέβαια, ο χρήστης μπορεί να πειραματιστεί, αλλάζοντάς τον με την χρήση ενός numeric up-down. **Περιέργως, η πρόσθεση χρωμοσωμάτων στον αρχικό πληθυσμό, δεν σημαίνει απαραίτητα ότι το πρόγραμμα θα φτάσει στην εύρεση μιας βέλτιστης λύσης γρηγορότερα.** Υπενθυμίζεται ότι βασιζόμαστε κυρίως στην τύχη. Έχοντας περισσότερα χρωμοσώματα, ναι μεν αυξάνει τις πιθανότητες να βρεθεί βέλτιστη λύση, αλλά σημαίνει και περισσότερη κατανάλωση υπολογιστικής ισχύος.

Πώς γίνεται η αναπαραγωγή;

Στην κύρια ροή, αφού παραχθούν οι λύσεις, τις ταξινομούμε με φθήνουσα σειρά βάσει του σκορ καταλληλότητάς τους. Κρατώντας απείραχτο το καλύτερο χρωμόσωμα, αναπαράγουμε λύσεις βάσει αυτού του χρωμοσώματος, διασταυρώνοντας σε ένα σημείο με τις υπόλοιπες λύσεις. Το σημείο στο οποίο γίνεται διασταύρωση είναι πάντοτε τυχαίο. Έχοντας, προηγουμένως, ταξινομήσει τα χρωμοσώματα, **φτάνουμε γρηγορότερα στην εύρεση μία λύσης, μια που διασταυρώνουμε πάντοτε τις καλύτερες λύσεις.**

Πόσο συχνά γίνεται κάποια μετάλλαξη;

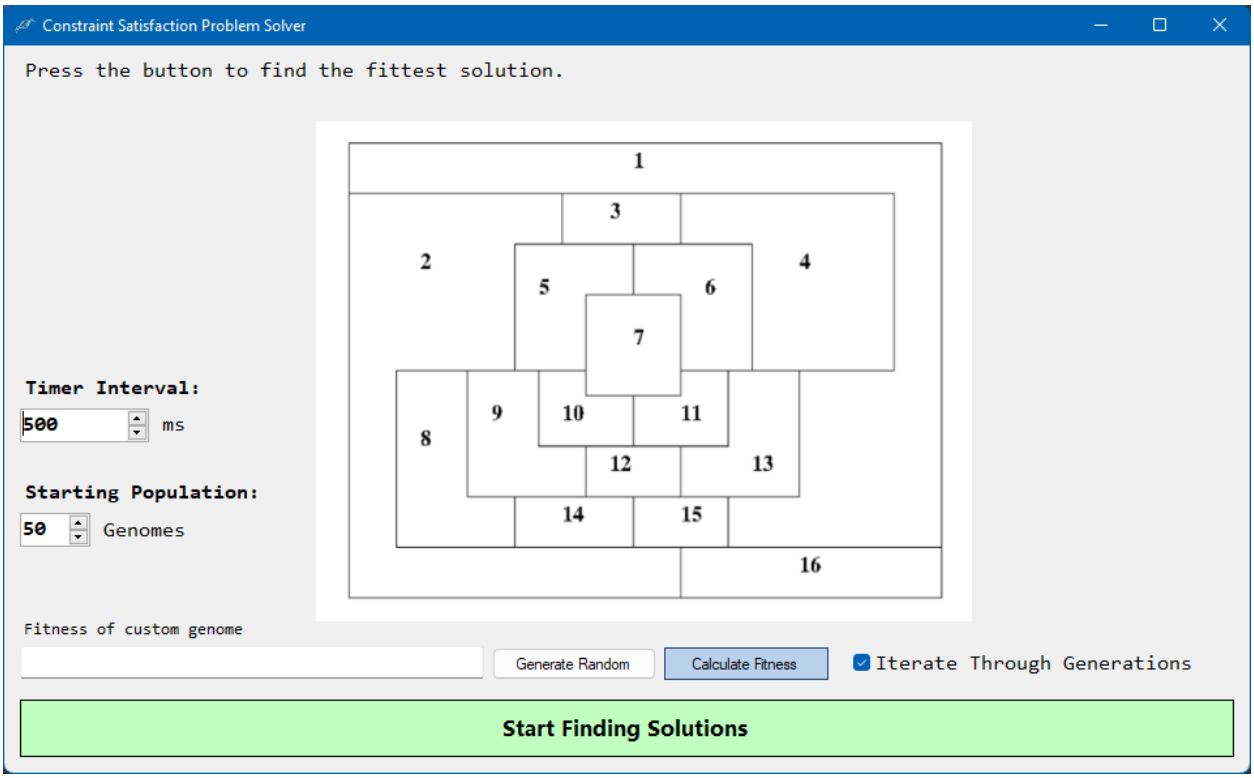
Μία μετάλλαξη έχει πιθανότητα 30% να συμβεί σε έναν απόγονο. Η μετάλλαξη δεν γίνεται στο καλύτερο χρωμόσωμα. Αυτή η μετάλλαξη συμβαίνει μόνο σε ένα bit. Όταν λέμε «30% πιθανότητα μετάλλαξης», εννοούμε ότι υπάρχει 30% πιθανότητα να αλλάξει ένα τυχαίο bit και 70% πιθανότητα να παραμείνει ίδια η λύση. **Βέβαια, υπάρχει η πιθανότητα, στην κύρια ροή, το πρόγραμμα να κωλύεται σε ένα τοπικό μέγιστο (local maxima).** Για να αποφύγουμε αυτό το ενδεχόμενο, ελέγχουμε αν το καλύτερο χρωμόσωμα παραμένει το ίδιο, για παραπάνω από οκτώ φορές στη σειρά. Σε αυτήν την περίπτωση παίρνουμε όλα τα χρωμοσώματα και κάνουμε **ολική μετάλλαξη με 50% πιθανότητα**. Αυτό, σημαίνει ότι για κάθε bit υπάρχει 50% πιθανότητα να αλλάξει ή να παραμείνει το ίδιο.

Παραδείγματα εκτέλεσης

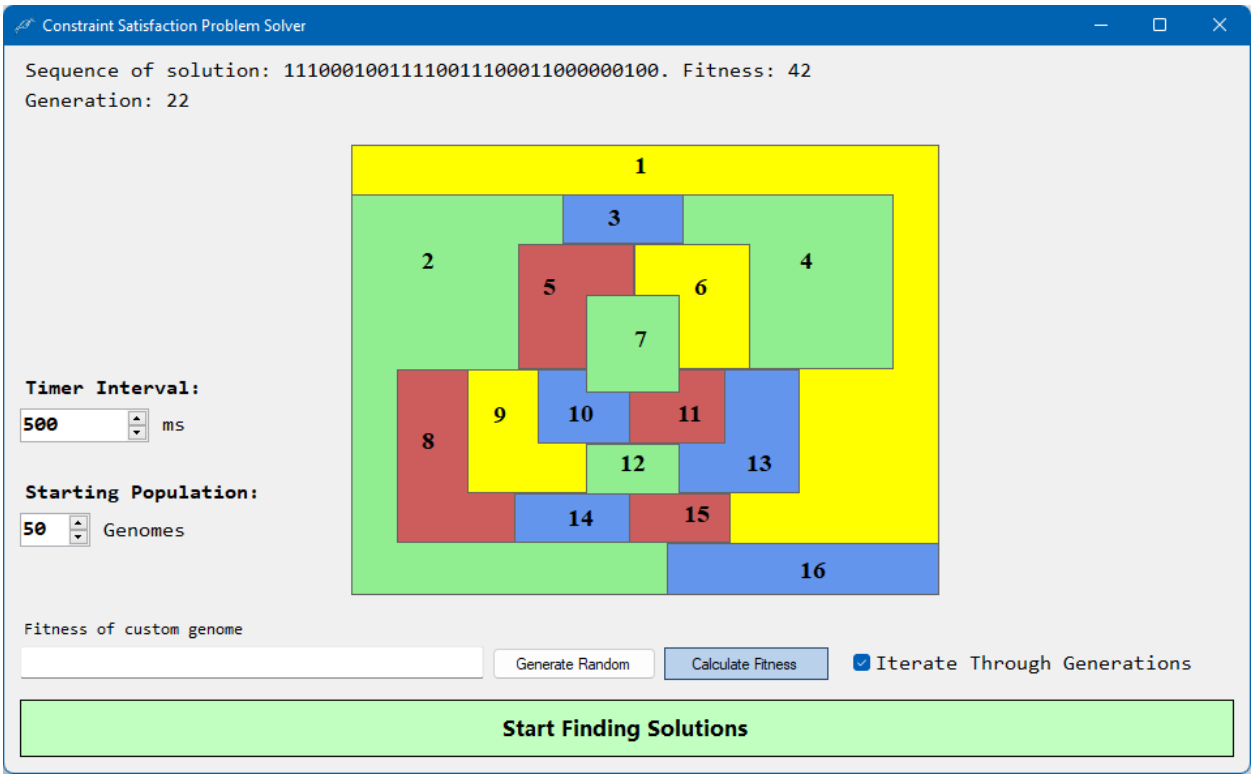
Υπόδειξη χρήσης της εφαρμογής και διάφορες ρυθμίσεις αναλύονται παρακάτω.

Αρχική Οθόνη

Η οθόνη, που αντικρίζει ο χρήστης με την πρώτη εκτέλεση, είναι η εξής:

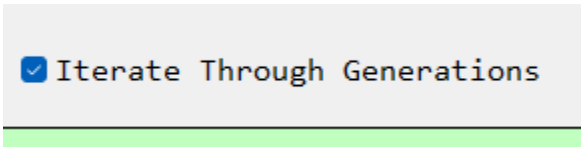


Για να δει ο χρήστης μία βέλτιστη λύση, θα πρέπει να πατήσει στο πράσινο κουμπί και ο γράφος θα αρχίσει να παίρνει τα διαθέσιμα χρώματα μέσα στα κουτάκια. Όταν γίνει αυτό, θα γραφεί η ακολουθία της λύσης και το σκορ καταλληλότητας (αφού είναι μία βέλτιστη λύση, θα είναι οπωσδήποτε 42), στο πάνω-αριστερά μέρος της φόρμας.

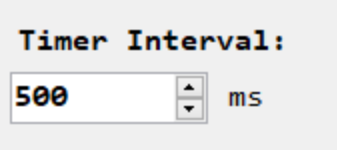


Παρακολουθώντας το Πέρας των Γενεών

Στα κάτω-αριστερά της φόρμας, ο χρήστης θα δει ένα check box, το οποίο στην αρχή θα είναι τσεκαρισμένο.

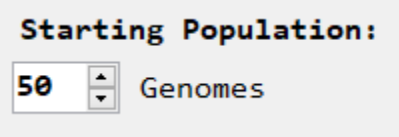


Υπάρχει αρκετό ενδιαφέρον στο πώς το πρόγραμμα φτάνει σε μία βέλτιστη λύση. **Ο χρήστης αποκτά τη δυνατότητα να δει τις ενδιάμεσες γενιές, πριν αυτές φτάσουν στο τέλος.** Στο πέρας κάθε γενιάς, αναπαρίσταται στον γράφο η λύση με το μεγαλύτερο σκορ καταλληλότητας αυτής της γενιάς. Ο χρήστης μπορεί να επιλέξει το πόσο γρήγορα θα περάσει στην επόμενη γενιά, επιλέγοντας τον χρόνο σε milliseconds, χρησιμοποιώντας την ρύθμιση στην δεξιά μεριά της φόρμας. Η ελάχιστη τιμή είναι μισό δευτερόλεπτο (500 ms) και η μέγιστη τιμή είναι 10 δευτερόλεπτα (10000 ms, δεν συνίσταται).

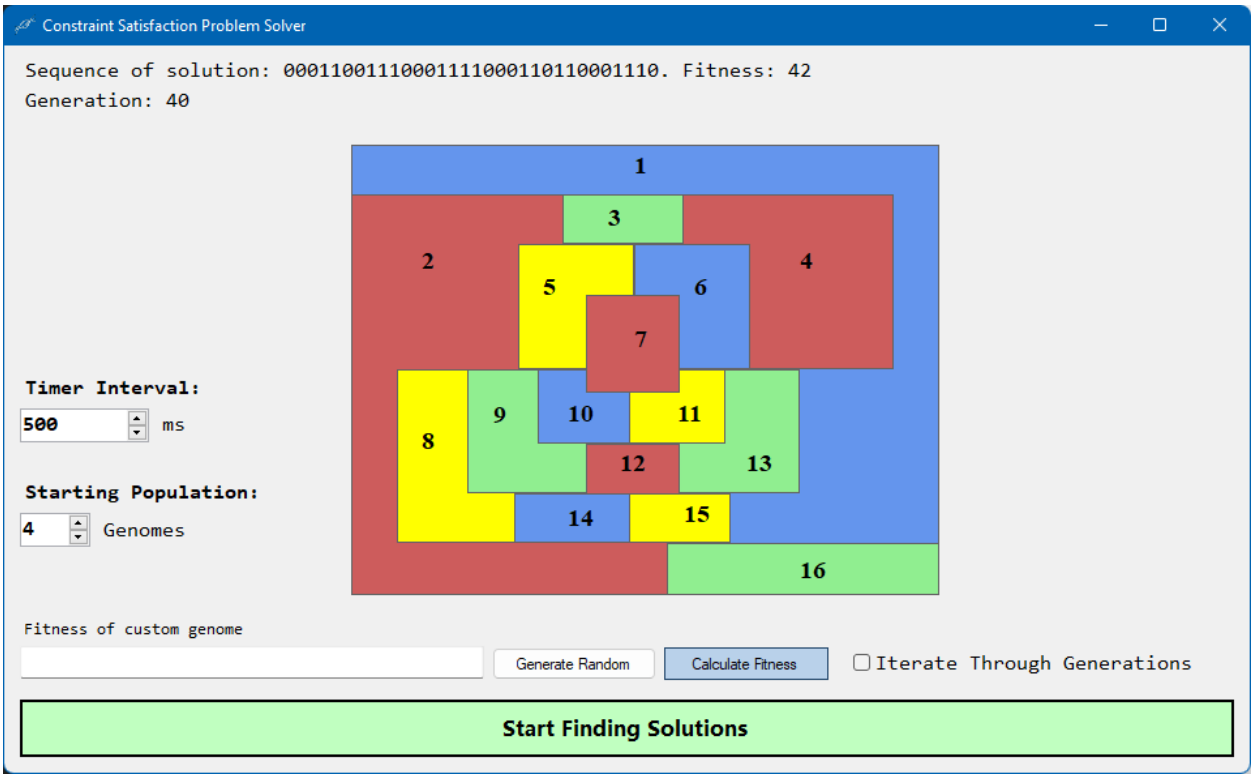


Πώς Επηρεάζει ο Αρχικός Πληθυσμός το Πρόγραμμα;

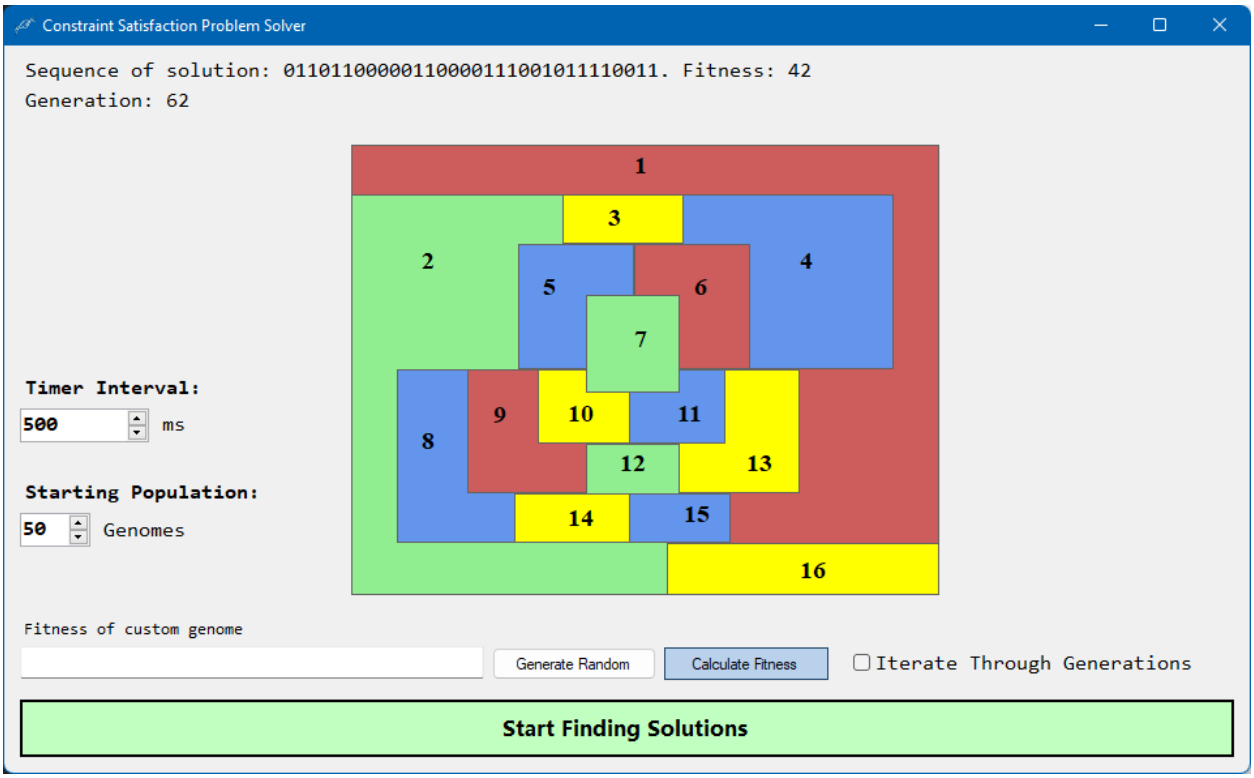
Η ρύθμιση του αρχικού πληθυσμού βρίσκεται στα αριστερά της φόρμας, κάτω από τον χειρισμό του χρόνου για το πέρας μια γενιάς.



Θα περίμενε κανείς ο αρχικός πληθυσμός να έχει τεράστια επίδραση στον χρόνο της εύρεσης της βέλτιστης λύσης. **Ωστόσο, όπως προαναφέραμε, δεν έχει σημασία ο αρχικός πληθυσμός σε αυτό,** καθώς βασιζόμαστε στην τυχαιότητα. Αυξάνοντας τον αρχικό πληθυσμό, αυξάνονται μονάχα οι πιθανότητες να ευρεθεί γρηγορότερα μια λύση. Ωστόσο, όπως παρατηρείται παρακάτω, υπάρχει περίπτωση να βρεθεί γρηγορότερα λύση, με μικρότερο αρχικό πληθυσμό.



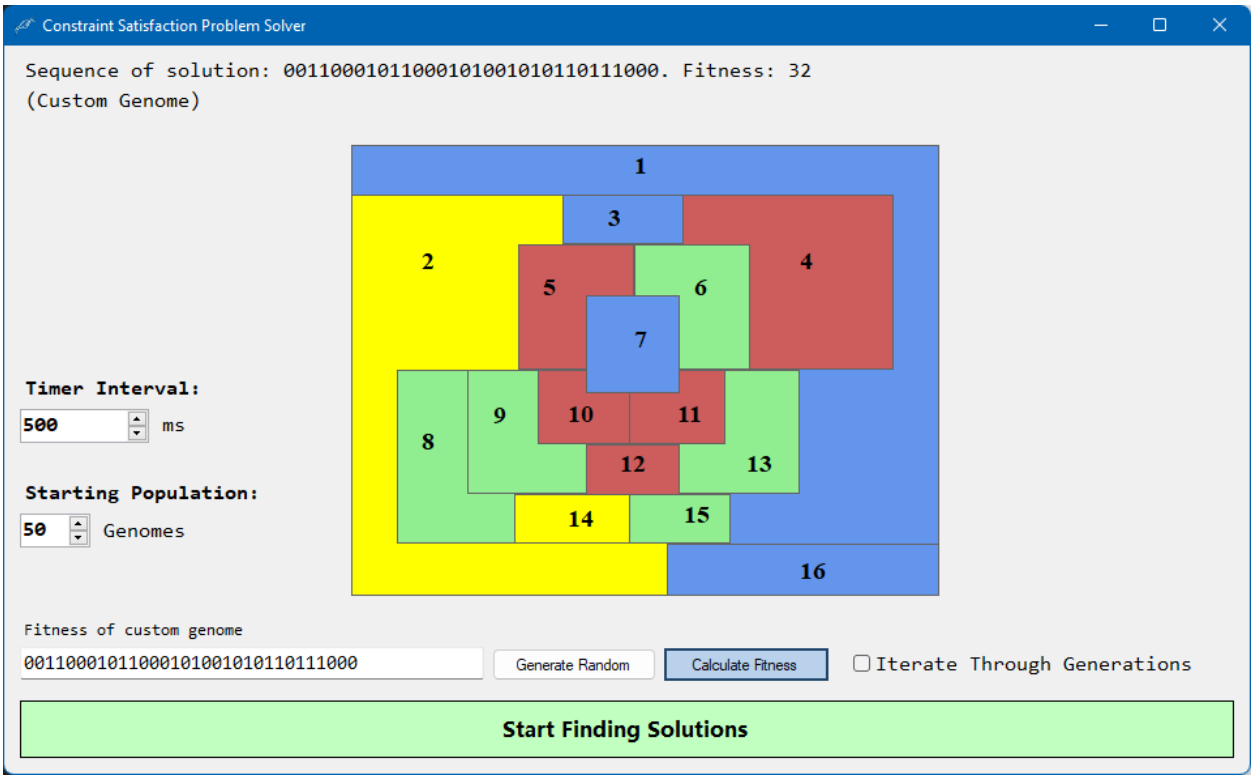
Εικόνα Α: 40 γενιές, με 4 αρχικά χρωμοσώματα



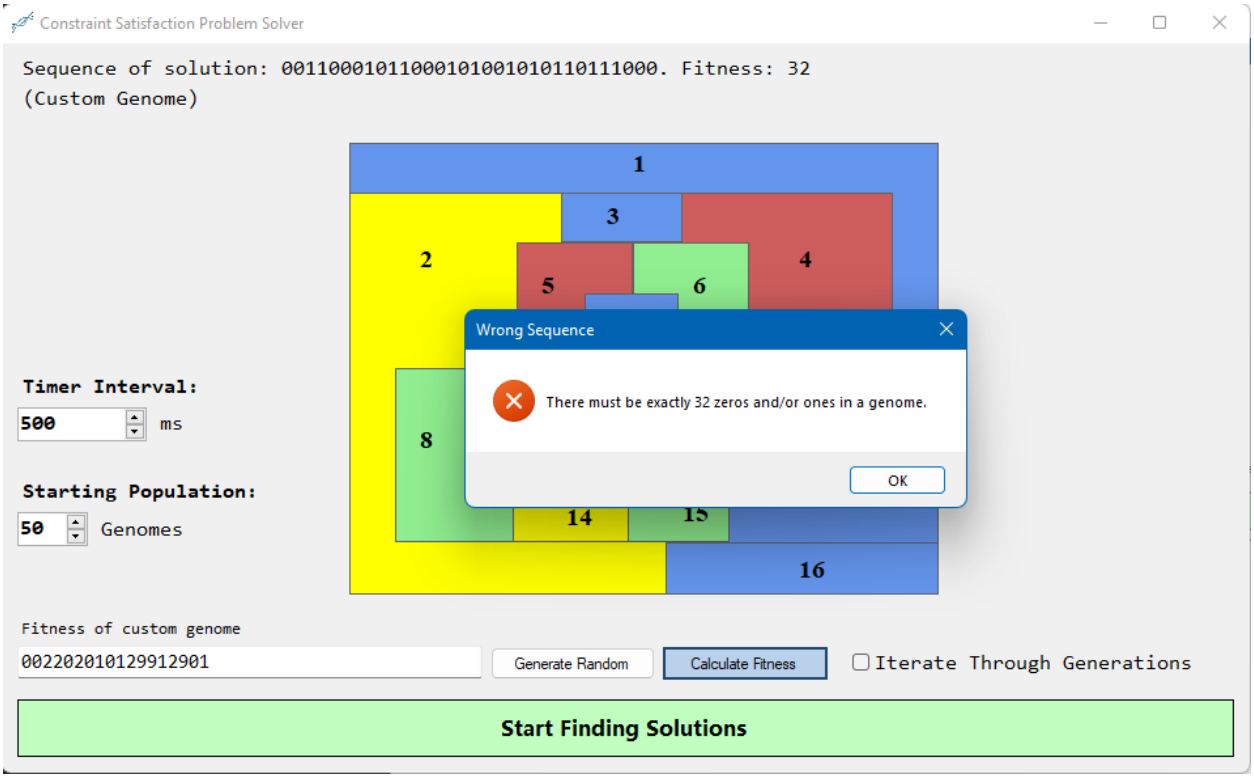
Εικόνα Β: 62 γενιές, με 50 αρχικά χρωμοσώματα

Χειροκίνητη Επίλυση

Αν ο χρήστης επιθυμεί να επιλύσει το πρόβλημα μόνος του, μπορεί να χρησιμοποιήσει τα δεδομένα που δόθηκαν και να την περάσει στο text box, κάτω-αριστερά της φόρμας. Ύστερα, θα πατήσει το μπλε κουμπί, που γράφει «Calculate Fitness» και θα δει αν η λύση που βρήκε είναι μία από τις βέλτιστες. **Ο χρήστης, βέβαια, οφείλει να περάσει ακριβώς 32 bit. Οποιοδήποτε άλλο αλφαριθμητικό δεν είναι αποδεκτό.**



Εικόνα C: Μία χειροκίνητη λύση (όχι βέλτιστη).



Εικόνα D: Λανθασμένη είσοδος.

Περιήγηση Αρχείων

Ιδανικά, οι παρακάτω μέθοδοι γίνονται στο λειτουργικό σύστημα των Windows 10 ή Windows 11 (64-bit), όπου έχουν δοκιμαστεί πλήρως.

I. Το πρόγραμμα αυτό έγινε στο προγραμματιστικό περιβάλλον του Rider. Επομένως, χρησιμοποιώντας είτε το Rider, είτε το Microsoft Visual Studio, μπορεί κανείς να ανοίξει το Project (επιλέγοντας από αυτά τα προγραμματιστικά περιβάλλοντα το .csproj αρχείο), να μεταγλωττίσει και να τρέξει κατ' ευθείαν το πρόγραμμα. Έχοντας, επίσης, ανοιχτό όλο το Project, φαίνεται όλος ο κώδικας πίσω από την Φόρμα, αλλά και από τις κλάσεις που χρησιμοποιήθηκαν, ώστε να φτιαχτεί ο αλγόριθμος. Αυτή είναι η ασφαλέστερη μέθοδος για να ανοίξει το πρόγραμμα χωρίς κανένα πρόβλημα.

II. Εναλλακτικά, ο χρήστης μπορεί να κατευθυνθεί στους εξς φακέλους: **Automatic-Color-Filler > bin > Debug > Automatic_Color_Filler.exe**, για το εκτελέσιμο.

Automatic-Color-Filler και ύστερα να ανοίξει με οποιονδήποτε text editor τα αρχεία Form1.cs, Genetic.cs, Population.cs και Genome.cs, ώστε να διαβαστεί ο κώδικας που τρέχει πίσω από το πρόγραμμα.