Άσκηση 3 Erlang για ραλλίστες

Ελευθέριος Καπελώνης Α.Μ. 03116163

Συνάρτηση `rally`

Η συνάρτηση `rally/3` λύνει το πρόβλημα. Χρησιμοποιεί BFS στο χώρο καταστάσεων του προβλήματος όπου ως κατάσταση θεωρείται το ζεύγος *unit* στο οποίο βρισκόμαστε μετά από κάποιο step και *ταχύτητα*.

Unit tests

Τα unit tests επαληθεύουν ότι η συνάρτηση `rally` παράγει το σωστό output σε συγκεκριμένα inputs. Η βιβλιοθήκη για τα unit tests είναι η `eunit` . Αρχικά γράφουμε τα 3 test cases (αυτά που δίνονται στην εκφώνηση). Για κάθε ένα χρησιμοποιούμε το `?_assertEqual` . Τελικά γράφουμε τη συνάρτηση `rally_test_` που εκτελεί τα 3 test cases.

Τρέχουμε τα unit tests:

```
8> c(rally).
{ok,rally}
9> eunit:test(rally).
  All 3 tests passed.
ok
```

Property based tests

Σε όλα τα property tests χρησιμοποιείται η συνάρτηση `generate_track` που παράγει ένα στιγμιότυπο του προβλήματος με τυχαίο τρόπο.

Το property `prop_upper_bound` ελέγχει ότι το output της συνάρτησης `rally` είναι μικρότερο ή ίσο του αριθμού των units στο track. Αυτό είναι ένα άνω φράγμα για το αποτέλεσμα.

```
15> proper:quickcheck(rally:prop_upper_bound()).
.....................................................................
.........................
OK: Passed 100 test(s).
true
```

Το property `prop_upper_bound` ελέγχει ότι το output της συνάρτησης `rally` είναι μεγαλύτερο ή ίσο του αριθμού των units στο track δια 24. Αυτό είναι ένα κάτω φράγμα για το αποτέλεσμα. Η ιδιότητα αυτή πρέπει να ισχύει καθώς στην καλύτερη περίπτωση η ταχύτητα θα είναι 240 καθόλη τη διάρκεια του ράλλυ.

```
16> proper:quickcheck(rally:prop_lower_bound()).
.....................................................................
.........................
OK: Passed 100 test(s).
true
```

Το πιο σημαντικό property είναι το `prop_correct_bfs` που ελέγχει την ορθότητα του BFS. Αρχικά

μετατρέπουμε τον ορισμό του προβλήματος σε ένα γράφο προσθέτοντας όλους τους κόμβους και τις επιτρεπόμενες ακμές μεταξύ τους. Στη συνέχεια, χρησιμοποιούμε τη συνάρτηση `digraph:get_short_path` στο συγκεκριμένο γράφο και βρίσκουμε το ελάχιστο μονοπάτι. Επειδή αυτή η συνάρτηση είναι της βιβλιοθήκης της Erlang είμαστε σίγουροι ότι είναι ορθή.

Αυτή η μέθοδος για να επιλύσουμε το πρόβλημα είναι χειρότερη, βέβαια, αλγοριθμικά από τον αλγόριθμο που χρησιμοποιήσαμε επειδή κατασκευάζει ολόκληρο τον γράφο. Αντίθετα, ο αρχικός αλγόριθμος κάνει expand (δηλαδή βρίσκει τις ακμές που πρόσκεινται) μόνο τους κόμβους που απαιτείται και έτσι δεν κατασκευάζει ολόκληρο το γράφο. Αλλά αυτό δεν είναι πρόβλημα εδώ αφού θέλουμε απλώς να ελέγξουμε την ορθότητα.

```
17> proper:quickcheck(rally:prop_correct_bfs()).
.....................................................................
.........................
OK: Passed 100 test(s).
true
```