Σώσε τη γάτα!

Μία αποδοτική λύση αυτού του προβλήματος ξεκινά με μια (σχεδόν) άμεση εφαρμογή του αλγόριθμου του flood fill (πλημμυρίσματος). Η μόνη προσθήκη που απαιτείται είναι αυτή της *χρονικής στιγμής* στην οποία συμβαίνει ένα γεγονός (δηλαδή που πλημμυρίζει κάποιο τετραγώνο του grid).

Σε ένα πρώτο πέρασμα:

1. τοποθετούμε στην ουρά τις πηγές του νερού, με χρόνο $t = 0$
2. οσο η ουρά δεν είναι κενή
   α. αφαιρούμε από την ουρά τις συντεταγμένες ενός τετραγώνου και τον αντίστοιχο χρόνο t
   β. αν το τετράγωνο αυτό δεν είναι ήδη πλημμυρισμένο
      - θέτουμε το χρόνο πλημμυρίσματος του τετραγώνου
      - προσθέτουμε τα γειτονικά τετράγωνα στην ουρά, με χρόνο $t + 1$

Στο τέλος αυτού του περάσματος, κάθε τετράγωνο του χάρτη που θα πλημμυρίσει θα είναι σημειωμένο με το χρόνο κατά τον οποίο αυτό θα συμβεί.

Σε ένα δεύτερο πέρασμα, κάνουμε ένα BFS ξεκινώντας από τη γάτα:

3. τοποθετούμε στην ουρά τις συντεταγμένες της γάτας, με χρόνο $t = 0$
4. όσο η ουρά δεν είναι κενή
   α. αφαιρούμε από την ουρά τις συντεταγμένες ενός τετραγώνου και τον αντίστοιχο χρόνο t
   β. βρίσκουμε τον ασφαλή χρόνο παραμονής σε αυτό το τετράγωνο (δηλαδή μία χρονική στιγμή πριν το τετράγωνο πλημμυρίσει)
   γ. αν αυτός είναι μεγαλύτερος από τον μέγιστο που έχουμε δει μέχρι τότε, ενημερώνουμε το μέγιστο και την αντίστοιχη θέση
   δ. για κάθε γειτονικό τετράγωνο που δεν είναι πλημμυρισμένο τη χρονική στιγμή $t + 1$
      - το προσθέτουμε στην ουρά, με χρόνο $t + 1$.

Για να ανακατασκευάσουμε τη λύση, φροντίζουμε να κρατάμε για κάθε τετράγωνο το γειτονικό του από το οποίο φθάσαμε σε αυτό και την κίνηση που κάναμε.

Σε περίπτωση που ο ασφαλής χρόνος παραμονής είναι ίσος με τον μέγιστο που έχουμε δει μέχρι τότε, στο βήμα 4β ελέγχουμε αν οι συντεταγμένες του νέου τετραγώνου είναι λεξικογραφικά μικρότερες. Για να πάρουμε τις λεξικογραφικά μικρότερες ακολουθίες κινήσεων που οδηγούν σε κάποιο τετράγωνο, φροντίζουμε να εξετάζουμε τις κινήσεις σε "αύξουσα" σειρά (`D`, `L`, `R` και `U`).

Στην παραπάνω περιγραφή έχουμε αγνοήσει λεπτομέρειες που μπορείτε να χειριστείτε εύκολα στην υλοποίηση (π.χ., εμπόδια, τετράγωνα που βρίσκονται στην άκρη του grid, κ.λπ.).

Η συνολική πολυπλοκότητα της λύσης είναι $O(NM)$, εφόσον και τα δύο βήματα (flood fill και BFS) έχουν αυτή την πολυπλοκότητα (το πλήθος των ακμών είναι κατά μέγιστο 4 για κάθε τετράγωνο). Σε γλώσσες αγνού συναρτησιακού προγραμματισμού, ίσως χρειαστεί να χρησιμοποιηθεί μία κατάλληλη δομή δεδομένων, π.χ. ordered map, που θα εισαγάγει έναν επιπλέον λογαριθμικό παράγοντα πολυπλοκότητας.

- Ordered map σε SML/NJ: BinaryMapFn
- Ordered map σε OCaml: Map