

Tehisintellekti ja masinõppe alused: Lokaalne otsing

Liine Kasak

March 3, 2019

1 Lahenduskäik

Implementeeritakse lahenduse leidmiseks ühedimensioonilist masiivi, et vähendada keerukusklassi. Koodis paigutatakse lipud suvalisele reale, kuid iga lipp on eraldi veerus.

Algoritmi peamine töö peitub selles, et märke ronimise algoritmi jaoks arvutatakse välja, kui palju on konfliktis olevaid lippe ning selle põhjal otsustatakse lipu uue positsiooni asukoht.

1.1 Konfliktide arvutamine

Konfliktide arvutamiseks kontrollitakse masiivi, kus on algselt paigutatud lipud, ning kontrollitakse ridu ja diagonaale, kas kuskil tekib konflikte. Leitud konfliktide arv liidetakse kokku ning tagastatakse. Üks paar konfliktis olevaid lippe on üks konflikt.

1.2 Parima käigu leidmine

Parima käigu leidmiseks itereeritakse üle võimalikest käikudest. Võetakse järjest üks võimalikest käikudest ette ning kontrollitakse, kas esialgne konfliktide arv väheneb selle tulemusel või mitte. Kui liigutamise tulemusel saadakse parem tulemus, jäetakse see meelde ning salvestatakse. Pärast seda taastatakse seis. Funktsiooni lõpus tagastatakse muutujad, mis tähistavad parimat liigutust.

2 Random restart

Mäkkeronimise otsingu korral ei leidu alati lahendus, kuna otsing võib peatuda platool või lokaalses miinimumis.

Probleemi lahendamiseks kasutati *Random restarti*. Selle algoritmi implementeerimiseks tuli otsingut teha korduvalt, alustades iga kord juhuslikust algolekust. Kui lahendust ei leitud 200 korraga, katkestati otsing. Antud lisa mäkkeronimise algoritmi ei muutnud, parimad olekud salvestati samamoodi.