Impact Factor: 9.9 ISSN-L: 2544-980X

Eng Qisqa Yo`L Masalasini Hal Qiluvchi Zamonaviy Li Algoritmi Samaradorligi

Gulnora Yodgorovna Buronova [1], Fayzullayev Azizbek Otabekovich [2]

Annotatsiya: Mazkur maqola zamonaviy eng qisqa yo`l masalalarni hal qilishda qo`llaniladigan algoritmlarini tahlil qilishga bag'ishlangan. Kirish BFS, DFS, Prim,Li kabi eng keng tarqalgan graf algoritmlari haqida umumiy ma'lumot beradi. Maqolaning asosiy qismi zamonaviy graf algoritmlarining zaif tomonlarini tahlil qilishni o'z ichiga oladi va turli xil hujum usullarini ko'rib chiqadi. Umumiy qilib aytganda, graf ma`lumotlar strukturasida eng qisqa yo`lmasalarini hal qilishning kompleks usullarini qo'llash va Li algoritmini qadan ba qadam ishlash prinsipi sodda tilda ifoda etilgan.

**Kalit so`zlar:** node, tugun, edge, qirra, Li algoritmi, to`lqin algoritmi, marshrutlash.

Labirintdagi eng qisqa yo'l (Shortest Path) algoritmi grafda ikki tugun (vertex) orasidagi eng qisqa yo'li topish uchun ishlatiladi. Masalan, Dijkstra, BFS, DFS, Prim hamda Bellman-Ford va Li algoritmlari bu maqsadga erishish uchun foydalaniladi. Eng qisqa yo‘l muammosini quyidagilarda kuzatish mumkin:

- Navigatsiya (Google Maps, Waze, Yandex Maps)
- Aloqa tarmoqlarida
- Ip routing
- Ijtimoiy tarmoqlarda odamlarni topishda
- Robot yoki dron uchun yo‘l topishda

Li 1961 yilda ikkita terminalli tarmoqni, tarmoqqa yo'naltirish algoritmini taqdim etdi. Bunda asosiy algoritm ham tezlik, ham xotira talablari uchun yaxshilandi. Li algoritmi va uning turli takomillashtirilgan versiyalari labirintlarni marshrutlash algoritmlari sinfini tashkil qiladi. Ushbu algoritm tekis to'rtburchaklar to'rning istalgan ikkita cho'qqisi orasidagi yo'lni topish uchun eng ko'p qo'llaniladigan algoritmdir.

Labirintlarni marshrutlash, tekis to'rtburchaklar panjarali grafikda mos ravishda manba(lar) va maqsad(lar) deb ataladigan juft nuqtalar orasidagi yo'lni topish uchun ishlatiladi. Marshrutlash uchun mavjud maydonlar bloklanmagan cho'qqilar sifatida, to'siqlar esa bloklangan cho'qqilar sifatida ifodalanadi. Labirintni marshrutlash algoritmining maqsadi hech qanday bloklangan cho'qqidan foydalanmasdan manba va maqsad cho'qqisi o'rtasidagi yo'lni topishdir. Yo'lni topish jarayoni qidiruv bosqichidan boshlanadi, bunda bir nechta yo'llar manbadan boshlanadi va ulardan biri nishonga yetguncha kengaytiriladi. Maqsadga erishilgandan so'ng, yo'lni aniqlash uchun cho'qqilarni manbaga qaytarish kerak. Tekshirish bosqichida har bir cho'qqining ota-onasi haqidagi ma'lumotlar saqlanib qolsa, qayta tiklash bosqichi osongina amalga oshirilishi mumkin. Bu yo'llarni o'rganishning bir qancha usullari ishlab chiqilgan. Li algoritmining mashhurligining kaliti uning soddaligi va agar mavjud bo'lsa, optimal yechimni topish kafolatidir.

Li algoritmining kashfiyot bosqichi kenglikdagi birinchi qidiruvning takomillashtirilgan versiyasidir. Qidiruvni manbadan tarqaladigan to'lqin sifatida ko'rish mumkin. Li algoritmi yo'lni topish algoritmi

[1] Bukhara State University, p.f.f.d. dotsent
[2] Bukhara State University 1-2KIDT-22 guruh, 2-bosqich talabasi

va labirintlarni yo'naltirishning mumkin bo'lgan yechimi bo'lib, u ko'pincha bosilgan elektron platalardagi simlarni yo'naltirish uchun kompyuter quvvatli dizayn tizimlarida qo'llaniladi. Li algoritmining parallellashuvi muhokama qilinadi, bu uning samaradorligi va ishlov berish vaqtini yaxshilash uchun muayyan kontekstlarda ishlatilishi mumkin. Li algoritmi parallellashtirilgan va bir nechta ishlov berish birliklarida amalga oshiriladi. Bizning algoritmimiz oldingi to'lqinni kengaytirish sxemasini parallellashtiradi va oldingi to'lqin kengayishlarini bir vaqtning o'zida amalga oshiradi, shu bilan protsessordan foydalanishni oshiradi va ishlov berish vaqtini qisqartiradi. Biroq, bu bir qatlamli marshrutlash yordamida amalga oshiriladi.

Marshrutlash - bu bosilgan elektron plata yoki VLSI chipidagi turli modullarning terminallarini bir-biriga bog'laydigan ulanishlar to'plamini topish vazifasi. Har bir ulanish manba terminalini maqsad terminalga ulaydi. Li algoritmi ikkita terminal orasidagi eng qisqa yo'lni qidiradi va agar ulanish mavjud bo'lsa, ikkita nuqta orasidagi marshrutni topishni kafolatlaydi. Ushbu algoritm manba va maqsad terminallari orasidagi eng qisqa yo'lni ham kafolatlaydi. Marshrutlash algoritmlari avtomatlashtirilgan simlar, global marshrutlash, batafsil marshrutlash, SAPR va robot yo'lini rejalashtirish kabi boshqa muammolarga qo'llanilishi mumkin. Labirintni marshrutlash algoritmi, agar shunday yo'l mavjud bo'lsa, bitta sim uchun labirintdagi ikkita nuqta orasidagi eng qisqa yo'lni topishga harakat qiladi. Ushbu sxemada manba tugun o'zining to'rtta qo'shnisiga xabar yuboradi. Xabar to'lqin shaklida boshqa tugunlarga tarqaladi. Belgilangan joyga etib boradigan birinchi to'lqin jabhasi ulanish yo'lini belgilaydi. Ushbu algoritmda ikki bosqich mavjud. Birinchi bosqichda tugunlar manbadan masofalari bilan belgilanadi. Keyingi bosqichda masofalar manbadan eng kam masofaga ega bo'lgan yo'lni tanlab, manbaga o'tish uchun ishlatiladi.

Li algoritmi tavsifi uning xusisiyatlari

Li algoritmi marshrutlash muammolarining mumkin bo'lgan yechimlaridan biridir. Ushbu algoritm marshrutlash qatlamini panjara sifatida ifodalaydi, bu yerda har bir tarmoq nuqtasi qo'shni tarmoq nuqtalariga ulanishlarni o'z ichiga olishi mumkin.

Kuchli tomonlari:

- Agar mavjud bo'lsa, 2 nuqta o'rtasida ulanishni topish kafolati.
- Minimal yo'lni kafolatlash.

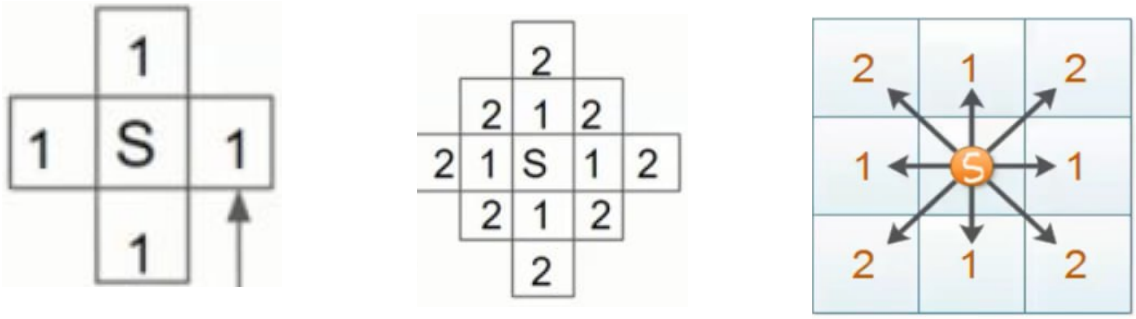
Kamchiliklari:

- Zich joylashtirish uchun katta xotira talab qilinadi.
- Sekin harakatlanadi.

Bu algoritmning asosiy maqsadi ikkita nuqtani (S manbadan T maqsadga) ulash uchun eng qisqa yo'lni (to'r tugunlari ketma-ketligini) aniqlash(1-rasm). Yo‘l kesishishi yoki egilishi mumkin.

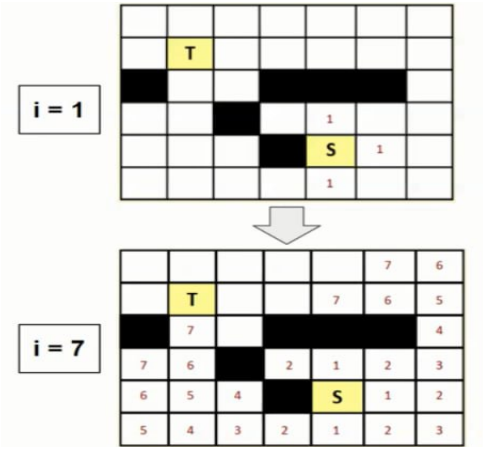
1-rasm.Yo`llarni aniqlash

Jarayon manbaning eng qo'shni kataglaridan boshlanadi. Birinchi yo'lda erishish mumkin bo'lgan barcha yangi katakchalarni topish kerak (ya'ni, umumiy uzunligi atigi 1 birlik faqat 1 katak bo'lgan barcha yo'llar). 1-yo'l uzunligidagi tugunlar yordamida 2-yo'lda erishish mumkin bo'lgan barcha yangi tugunlarni topish mumkin. Jarayon T ga erishilgunga qadar takrorlanadi.



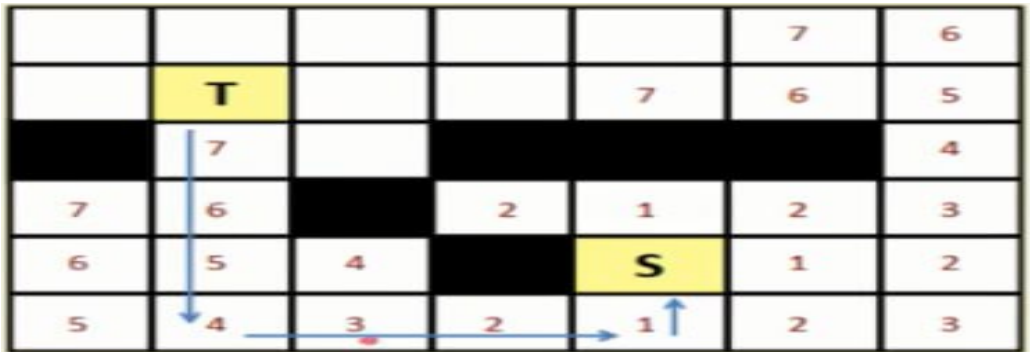
2-rasm. Li algoritmini ishlash 1-bosqichi

1-qadam: Yorliqlash maqsadli tarmoq yacheykasi T L iteratsiyasida belgilanmaguncha davom etadi (ya'ni, i=L bo'lganda).



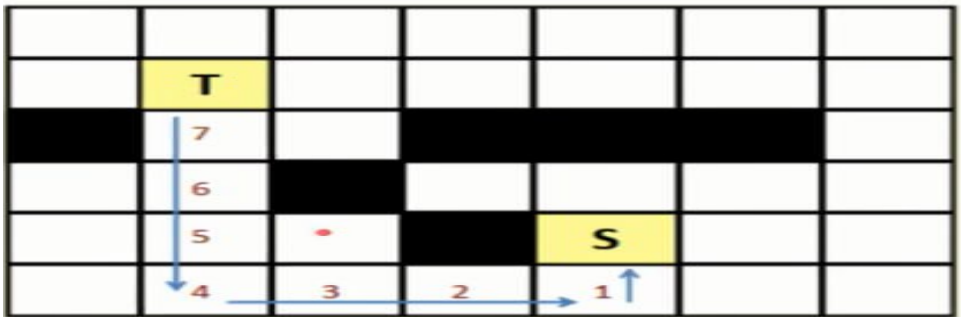
3-rasm. L - eng qisqa yo'lning uzunligi. Bu jarayonda i=L=7.

2-qadam: Qayta kuzatish , maqsadli T dan S manba tuguniga muntazam ravishda orqaga qayting. Agar i-bosqichda T ga erishilgan bo'lsa, unga qo'shni bo'lgan kamida bitta katakcha i-1 deb etiketlanadi va hokazo. Raqamlangan kataklarni kamayish tartibida kuzatib, biz eng qisqa yo'l L bo'ylab S ga erisha olamiz. Amalda, asosiy qoida, agar buni qilish kerak bo'lmasa, orqaga qaytish yo'nalishini o'zgartirmaslikdir.



4-rasm.

3-qadam: Yorliqlarni tozalash, hozirgina topilgan yo'lga mos keladiganlardan tashqari barcha etiketlangan tugunlar tozalanadi. Yo'l bo'ylab tugunlar to'siqlar sifatida belgilangan. Qidiruvning murakkabligi to'lqin tarqalish bosqichining o'zi kabidir.



5-rasm.

U kenglik bo'yicha birinchi qidiruvni amalga oshirish va har bir tarmoq nuqtasini manbadan masofa bilan belgilash orqali ulanishning manba va maqsad tugunlari o'rtasidagi eng qisqa yo'l ulanishini qidiradi. Agar ulanish mumkin bo'lsa, bu kengaytirish bosqichi oxir-oqibat maqsad tuguniga yetib boradi. Ikkinchi orqaga qaytish fazasi keyin yorliqlar kamaygan har qanday yo'lni bo'ylab ulanishni tashkil qiladi. Ushbu algoritm ma'lum ulanish uchun manba va maqsad o'rtasidagi eng qisqa yo'lni topishi kafolatlangan. Biroq, bir nechta ulanishlar amalga oshirilganda, bitta ulanish boshqa ulanishlarni bloklashi mumkin. To'lqin kengayishi bloklarda yoki allaqachon simli qismlarda emas, balki faqat chipning yo'naltiriladigan maydonidagi nuqtalarni belgilaydi. Segmentatsiyani minimallashtirish uchun iloji boricha bir yo'nalishda ushlab turish yaxshidir. Uning sturukturasi quyidagicha:

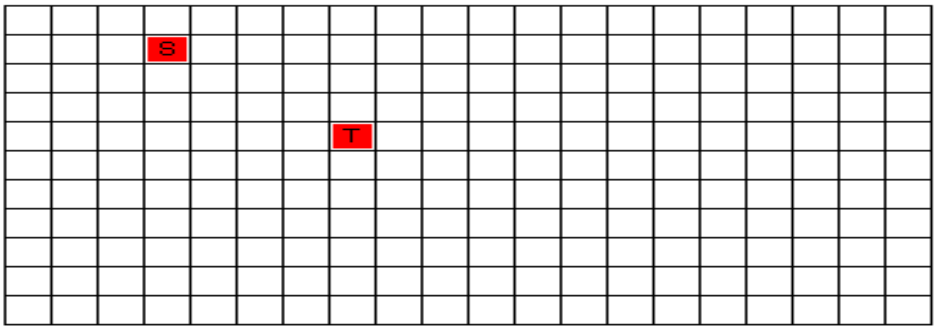
- Boshlanish nuqtasini tanlang, 0 bilan belgilash.
- i := 0
- Takrorlashni bajarish.
- i bilan belgilangan nuqtalarning barcha yorliqsiz qo'shnilarini i+1 bilan belgilang
- i := i+1
- Keyingi qadamda esa maqsadga erishildi yoki hech qanday ball belgilanmaydi.
- Orqaga qaytish.
- Maqsadli nuqtaga o'tish.
- haqiqiy tugundan pastroq belgiga ega bo'lgan keyingi tugunga o'tish
- ushbu tugunni yo'lga qo'shish
- UNTIL (boshlanish nuqtasiga yetdi)
- Tozalash
- Keyingi simlar uchun yo'lni to'sib qo'yish.
- Barcha belgilarni o'chirish.

Li algoritmini amalda boshlash:

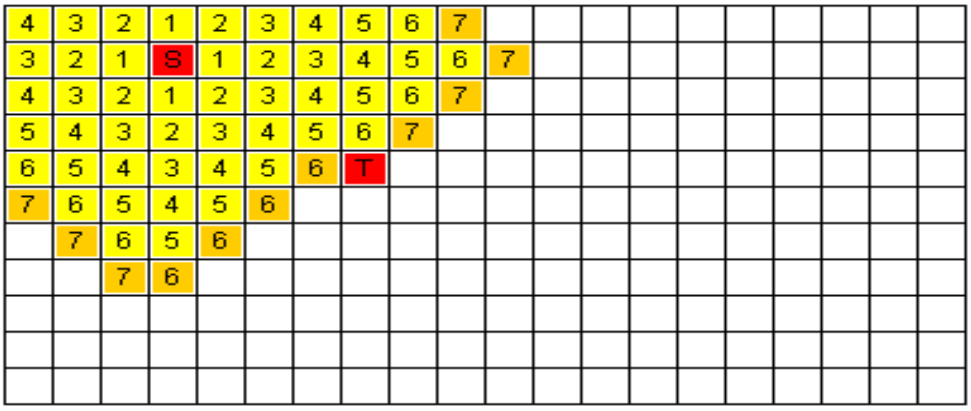
1. Marshrutlash qatlami panjara sifatida ifodalanadi
2. Har bir to'r nuqtasi qo'shni panjara nuqtalariga ulanishlarni o'z ichiga oladi.
3. Manba va maqsad o'rtasidagi eng qisqa yo'l ulanishini quyidagi yo'llar bilan qidiradi.
4. Kenglik-birinchi qidiruvni amalga oshirish.
5. Har bir tarmoq nuqtasini manbadan masofa bilan belgilash.
6. Trace-back fazasi yorliqlari kamayuvchi yo'llarni kuzatish orqali ulanishni tashkil qiladi.

1-rasmda manba va maqsad tugunlari tarmoqda tanlangan. Kengayish bosqichida 2-rasmda ko'rsatilganidek, to'lqinlar tarqala boshlaydi va "S" manbasidan maqsad "T" terminallarigacha

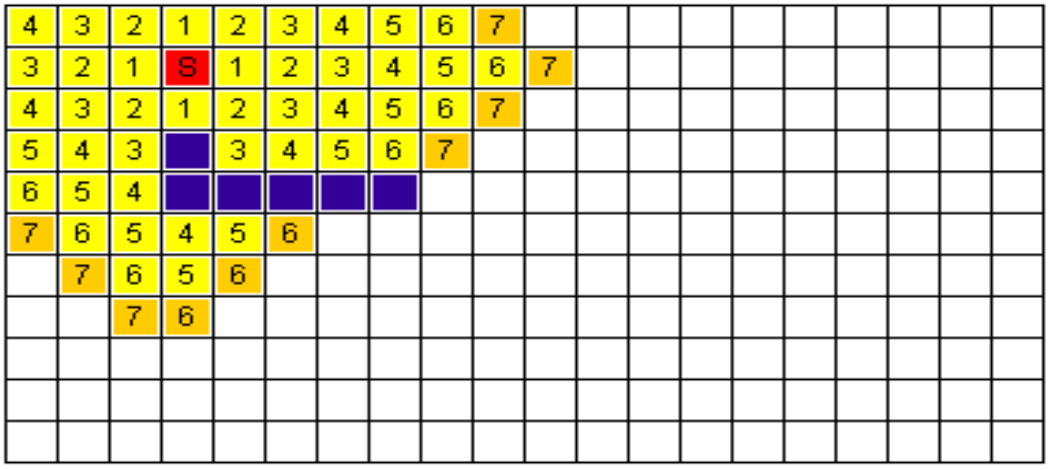
kengayadi. Maqsadga erishilganda, orqaga qaytish fazasi yoqiladi. 3-rasmda ko'rsatilganidek, algoritm maqsad nuqtasidan boshlanadi va tugunlar orasidagi eng qisqa yo'lni olish uchun tanlangan yo'lga qo'shish uchun eng kam yorliqli tugunlarni qidiradi. 4-rasmda orqaga iz tugallandi va ulanish hosil bo'ldi. Bu yo'l belgilangan va keyindagi simlar uchun bloklangan.



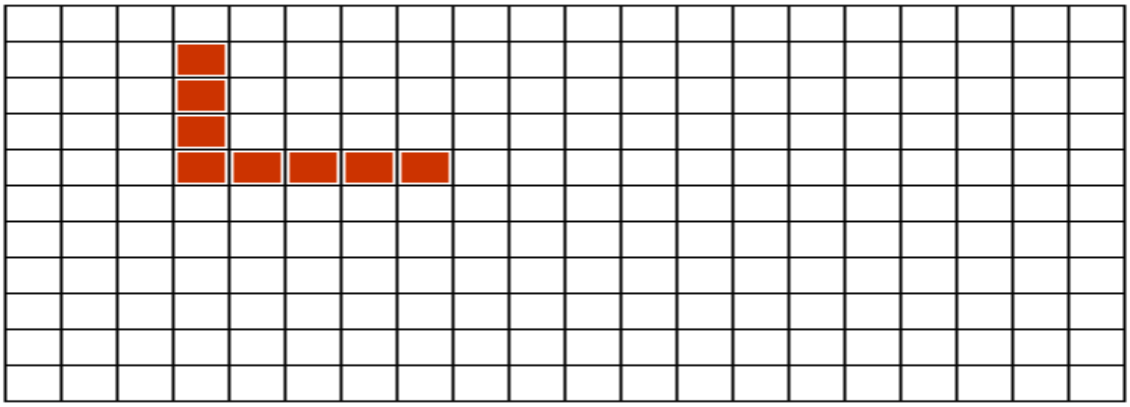
6-rasm: S va T, manba va tarmoq tugunlari tanlangan.



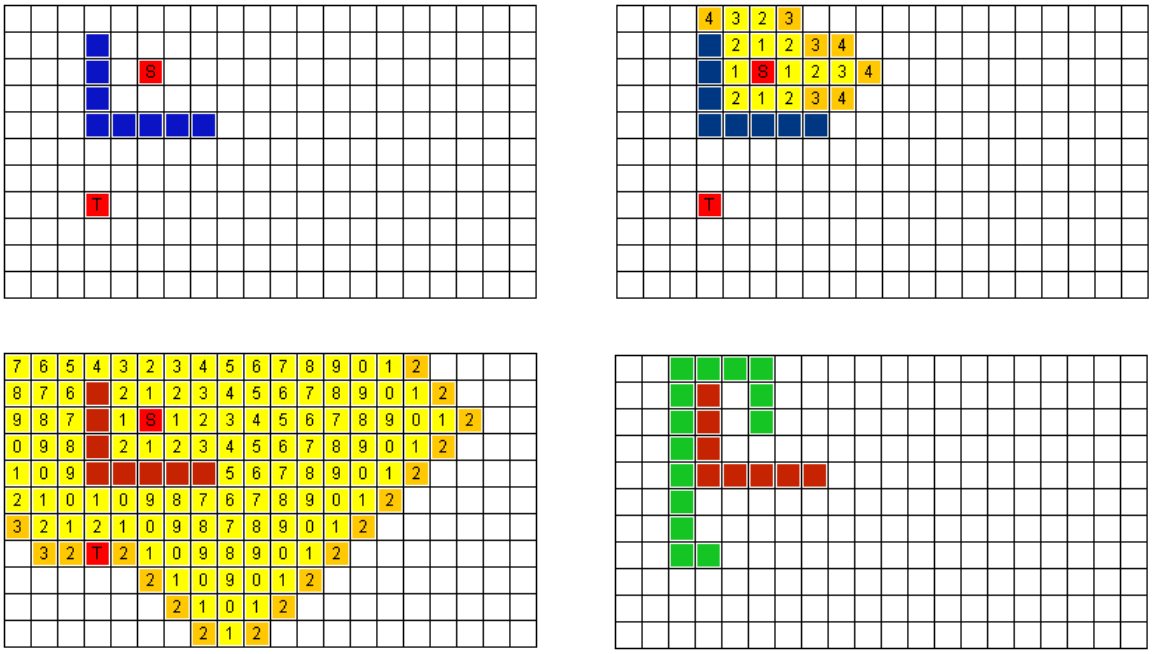
7-rasm: Kengayish jarayoni.



8-rasm: Ortga qaytish fazasi.

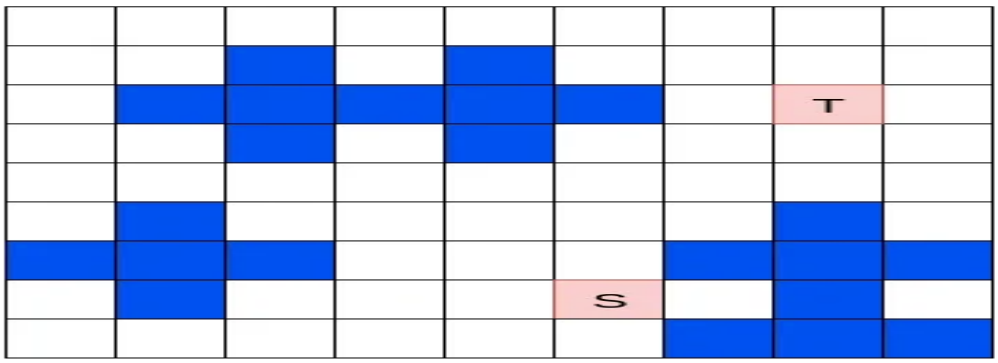


9-rasm: Keyindagi simlar uchun bloklash jarayoni.



10-rasm: Blokirovkaning har ikki tomoni uchun alohida ulanishni topishga harakat qilish algoritmi ko'rsatilgan. To'lqinlar maqsad tuguniga yetib borish uchun blokirovka atrofida tarqaladi.

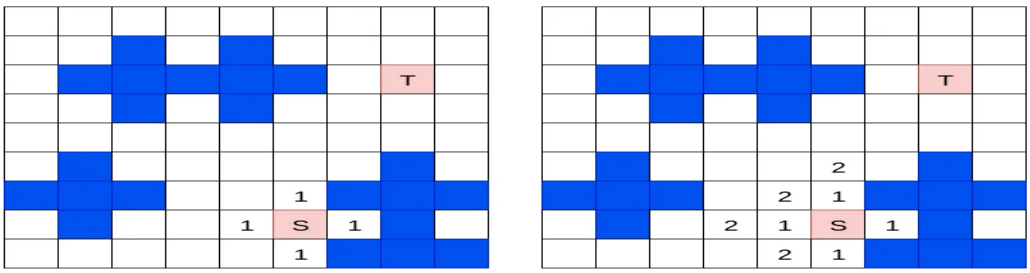
2-misol: Qizil kvadratlar S va T yuqoridagi to'rda boshlang'ich va maqsad nuqtalariga to'g'ri keladi. Moviy kvadratlar to'siqlarni anglatadi. Li algoritmi kompyuter yordamida dizayn tizimlarida simlarni bosilgan elektron platalarda, o'yin sanoatida (ayniqsa, real vaqtda strategiyalar uchun) va boshqa sohalarda qo'llaniladi.



11-rasm.

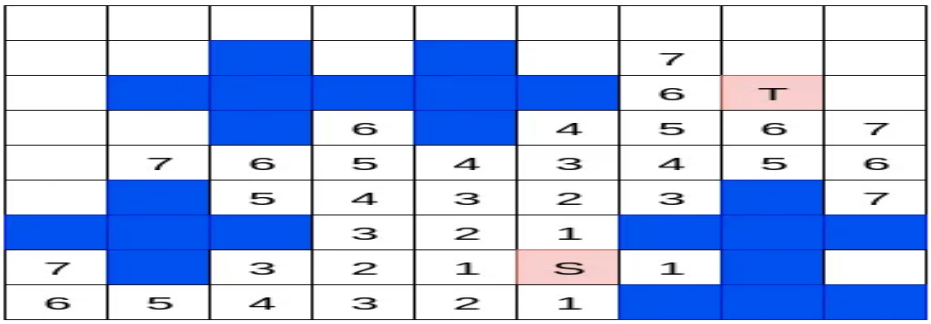
Li algoritmining birinchi qismi boshlang'ich va maqsad nuqtasi orasidagi eng qisqa yo'lning uzunligini topishdir. Uni topish bosqichma-bosqich jarayondir: biz boshidan boshlashimiz kerak, avval qo'shni nuqtalargacha bo'lgan masofalarni topamiz, so'ngra keyingi qo'shni nuqtalargacha bo'lgan masofani

hisoblaymiz va shunga o'xshash maqsadli nuqtaga oxirigacha erishamiz. Birinchi bosqichda barcha qo'shni nuqtalarning teglari S ga o'rnatiladi 1 va eng qisqa masofalar teng ekanligini ko'rsatadi. Shundan so'ng biz yorliqli nuqtalarning barcha qo'shnilarini ko'rib chiqamiz 1 va 2, boshlang'ich nuqtasidan eng qisqa masofalar teng ekanligini ko'rsatadi. Quyidagi rasmda ushbu ikki bosqichdan keyin to'r ko'rsatilgan:



12-rasm.

Ushbu protsedura maqsadli nuqtaga yetguncha davom etadi. Shundan so'ng, panjara quyidagicha ko'rinadi:

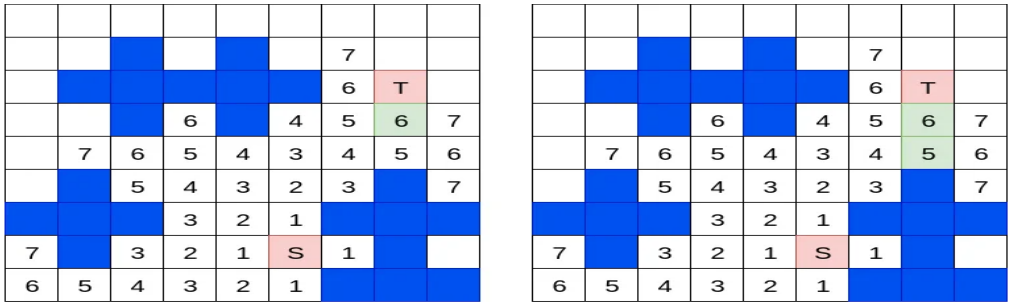


13-rasm.

Endi biz bilamizki, S va T orasidagi eng qisqa yo'l uzunligi 7 hisoblanadi. Xulosa qilib aytganda, Li algoritmining birinchi qismining bosqichlari:

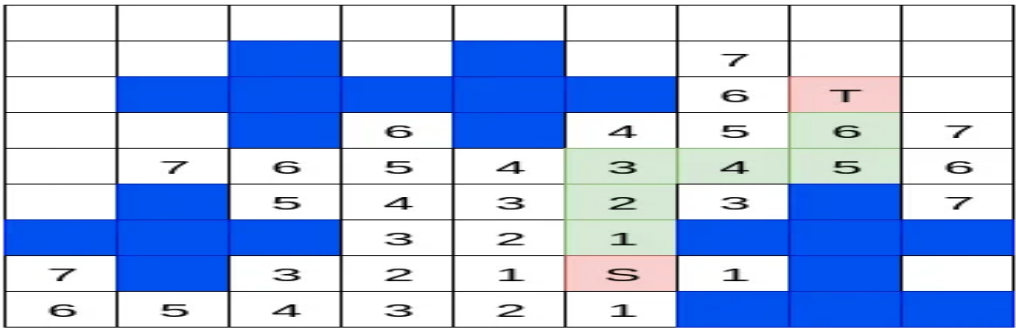
1. Boshlanish nuqtasini 0 bilan belgilang.
2. O'zgaruvchini i bilan 0 ishga tushiring.
3. Yorliqli nuqtalarning barcha yorliqsiz qo'shnilarini ko'rib chiqing. i va teglarini o'rnating i+1 kabi.
4. Maqsadga erishilgunga qadar bosqichlarni takrorlang.

Keyinchalik, boshlang'ich va maqsadli nuqtalar orasidagi eng qisqa yo'lni qayta qurishimiz kerak. Bu jarayon boshlang'ich nuqtasiga yetguncha davom etadi. Quyidagi raqamlar dastlabki ikki bosqichni ko'rsatadi; yashil nuqtalar orasidagi eng qisqa yo'lning boshlanishiga to'g'ri keladi S va i+1.



14-rasm.

Jarayon tugagandan so'ng, panjara quyidagicha ko'rinadi:



15-rasm.

Ba'zi hollarda, tanlash uchun bir nechta mumkin bo'lgan yo'nalishlar mavjud. Bu shuni anglatadiki, boshlang'ich va maqsadli nuqtalar o'rtasida bir nechta eng qisqa yo'llar bo'lishi mumkin. Shunday qilib, Li algoritmlarining ikkinchi qismining bosqichlari:

1. Maqsadli nuqtani joriy sifatida belgilang.
2. O'zgaruvchini ishga tushirish i maqsadli nuqta yorlig'i bilan.
3. Joriy nuqtaning barcha qo'shnilarini ko'rib chiqing i−1 kabi.

Pythonda Li algoritmini labirintlarni yo'naltirish muammosini hal qilish orqali o'rganamiz. Li algoritmi xususiyatlariga ega

- agar mavjud bo'lsa, har doim yo'l topish va
- har doim eng kam mumkin bo'lgan xarajat bilan yo'lni topish

Xulosa

Xulosa qilib aytganda, bu algoritm kompleks labirintlarda hatto boshqa algoritmlarni ishlatmagan holda ham ishlay oladi. Uning samaradorligi ko'p mashg'ulotli labirintlarda ham sinovdan o'tganligi uchun juda qulaylik bilan foydalaniladi. Algoritmda jarayonning yalpi vaqt o'lchami labirintning o'lchamiga va murakkablik darajasiga bog'liq bo'lishi mumkin.

Li algoritmi to'g'ri yo'l borligini va bu eng qisqa yo'l ekanligini kafolatlaydi . Ammo, bu algoritm juda ko'p vaqt va xotirani talab qiladi , bu kamchiliklarni bartaraf etish uchun Line Search Algoritm, Steiner Algoritms va boshqalar kabi boshqa ilg'or algoritmlar mavjud.

Foydalanilgan adabiyotlar:

1. Алфред В. Ахо., Джон Э. Хопкрофт, Джефри Д. Ульман. Структура данных и алгоритмы. //Учеб.пос., М.: Изд.дом: "Вильямс", 2000, 384 с.
2. Adam Drozdek. Data structures and algorithms in C++. Fourth edition. Cengage Learning, 2013.
3. Бакнелл Джулиан М. Фундаментальные алгоритмы и структуры данных в Delphi//СПб: ООО «ДиаСофтЮП», 2003. 560с.
4. Narzullaev U.X., Qarshiev A.B., Boynazarov I.M. Ma’lumotlar tuzilmasi va algoritmlar. //O’quv qo’llanma. Toshkent: Tafakkur nashriyoti, 2013 y. – 192 b.
5. Лойко В.И. Структуры и алгоритмы обработки данных. Учебное пособие для вузов. - Краснодар: КубГАУ. 2000. - 261 с., ил.