Matematika labirinata

Knežević, Marko

SVEUČILIŠTE U ZAGREBU

PRIRODOSLOVNO–MATEMATIČKI FAKULTET

MATEMATIČKI ODSJEK

Marko Knežević

MATEMATIKA LABIRINATA

Diplomski rad

Voditelj rada:
doc. dr. sc. Franka Miriam Brückler

Zagreb, studeni, 2015.

Ovaj diplomski rad obranjen je dana _____ pred ispitnim povjerenstvom u sastavu:

1. _____, predsjednik
2. _____, član
3. _____, član

Povjerenstvo je rad ocijenilo ocjenom _____.

Potpisi članova povjerenstva:

1. _____
2. _____
3. _____

Mami i braći.
Though your dreams be tossed and blown...

Sadržaj

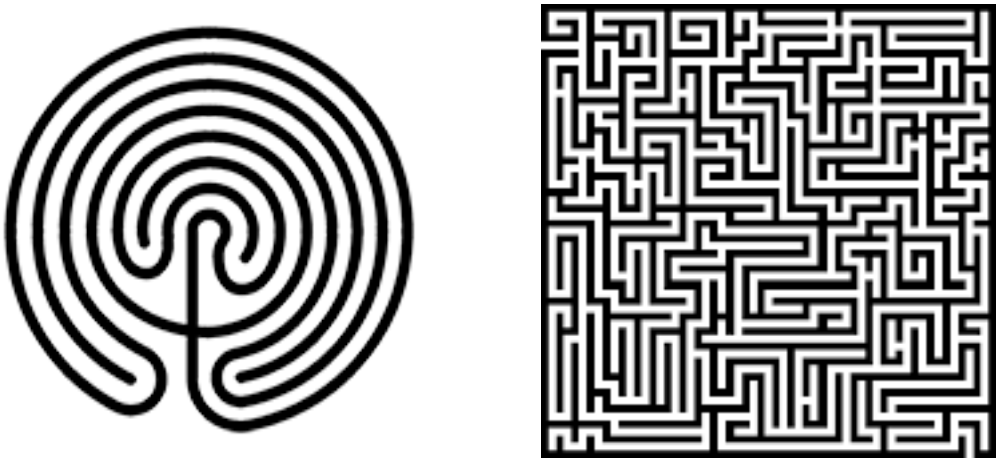
Uvod

Labirinti se prvi put spominju u grčkoj mitologiji. Prema legendi, labirint u Knosu je izgradio Dedalus po nalogu kralja Minosa, kako bi u njoj zadržali Minotaura, mitsko biće sa ljudskim tijelom i glavom od bika. Najpoznatiji labirint u Engleskoj je pak *Rosamond's Bower*. Legenda kaže da je izgrađen po nalogu kralja Henrika II kako bi sakrio svoju ljubavnicu *Rosamond the Fair* od svoje žene *Eleanor of Aquitaine*. Labirinti su dakle građeni kako bi zadržali nešto unutra i radi toga su morali bit vrlo složeni. Tijekom povijesti, njihova uloga se mijenjala. U mitologiji je labirint bila kompleksna građevina koja sprečava nekoga (ili nešto) da pobjegne. U srednjem vijeku su se počeli koristiti u duhovnim ritualima i za meditaciju. Kretanje po labirintu simboliziralo je dugačak put od početka do kraja, odnosno životno putovanje od rođenja pa sve do smrti. Krajem dvadesetog stoljeća labirinti su se počeli koristiti kao testovi inteligencije i sve češće u znanstvenim istraživanjima prostorne orijentacije i sposobnosti učenja. Zahvaljujući labirintima, znanstvenici su otkrili protein čiji je nedostatak povezan sa nastankom Alzheimerove bolesti, te su istražili djelovanje Valiuma i ostalih lijekova protiv anksioznosti.

U engleskom jeziku uobičajeno je koristiti izraze *labyrinth* i *maze* kao sinonime, no razlika postoji. *Labyrinth* je struktura koja se sastoji od jednog ulaza (ujedno i izlaza) te jednog hodnika. Drugim riječima, ako nakon ulaska dovoljno dugo hodamo, vratit ćemo se opet na isto mjesto. Krećemo se jedinstvenim putem od ulaza do izlaza. Primjer za *labyrinth* je prikazan na slici 0.1a.

Maze je složena struktura koja se može sastojati od više ulaza, više izlaza i više hodnika, a put od ulaza do izlaza nije uvijek jedinstven. Primjer za *maze* se nalazi na slici 0.1b. Iako su po tehničkim karakteristikama i stupnju složenosti *labyrinth* i *maze* vrlo različiti u hrvatskom jeziku koristimo samo jedan naziv: labirint.

Labirinti imaju široku primjenu i korišteni su u razne svrhe osim već navedenih, a dio su i popularne kulture kao zabavni zadaci u novinama i drugim publikacijama. U ovom radu ćemo se usredotočiti na matematičku pozadinu labirinta, te opisati razne algoritme za generiranje i rješavanje labirinata, kao i primjere kako nam labirinti mogu pomoći u nastavi



(a) *Labyrinth* (b) *Maze*

Slika 0.1: Usporedba *Labyrinth* i *Maze*

i popularizaciji matematike.

Poglavlje 1

Teorija grafova

Državni zavod za statistiku matematiku kao znanstveno polje dijeli na 10 znanstvenih grana: algebra, geometrija i topologija, kombinatorna i diskretna matematika, matematička analiza, matematička logika i računalstvo, numerička matematika, primjenjena matematika i matematičko modeliranje, teorija vjerojatnosti i statistika, financijska i poslovna matematika, te na ostale matematičke discipline.

Za ovaj diplomski rad su ključne samo dvije, geometrija i topologija te kombinatorna i diskretna matematika. Zapravo, glavna poveznica između labirinata i matematike je teorija grafova, matematička disciplina na granici između diskretne matematike i topologije.

Teorija grafova i topologija su nastale u 18. stoljeću kada je švicarski matematičar Leonhard Euler formulirao i riješio problem *Sedam königsberških mostova*. Njegov članak *Solutio problematis ad geometriam situs pertinentis* u časopisu *Commentarii academiae scientiarum Petropolitanae* objavljen 1747. godine smatra se prvim radom u teoriji grafova. Topologija se razvijala u raznim smjerovima, a matematičar i kemičar Arthur Cayley je krajem 19. stoljeća utemeljio pojam stabla i razvio prvu sustavnu metodu u teoriji grafova. Od ovog trenutka se teorija grafova i topologija razvijaju kao dvije discipline.

Topologiju u žargonu možemo nazvati geometrijom gumenih objekata. Topolozi proučavaju svojstva koja ostaju nepromijenjena kada neki oblik deformiramo na neprekidan način, odnosno kada ga rastegnemo, izvrnemo ili promijenimo na neki drugi način bez kidanja. Takve transformacije nazivamo topološkim ekvivalencijama. Preciznije, radi se o homeomorfizmima: neprekidnim preslikavanjima s neprekidnim inverzima. Primjerice, kocka je topološki ekvivalentna kugli. Topološka svojstva poput povezanosti i nepostojanje rupa su ostala nepromijenjena. Kuglu možemo preoblikovati u valjak te dobiti još jedno topološku ekvivalenciju. Međutim, dozvoljeno je privremeno „zarezati" objekt, ali pod uvjetom da se dijelovi ponovo spoje tako da susjedne točke duž reza ponovo budu susjedne.

Takvim neprekidnim transformacijama možemo u potpunosti ignorirati okolni prostor i usredotočiti na unutrašnja svojstva objekata. Kod labirinta, neka od unutrašnjih svojstva bi bila križanja i hodnici, odnosno njihov broj. Nije nam bitno je li hodnik dugačak 5 metara, 50 metara ili 500 metara. Raspored hodnika unutar labirinta također možemo zanemariti ukoliko se oni nalaze između ista dva križanja. Ako su dva križanja direktno povezana sa dva hodnika, recimo crvene i plave boje, svejedno nam je koji od njih je lijevi, a koji desni.

Kako bi smo lakše i preciznije objasnili algoritme za generiranje i rješavanje labirinata, slijedi pregled osnovnih pojmova i rezultata teorije grafova koje ćemo koristiti u ostatku rada.

Teorija grafova se bavi matematičkom strukturom zvanom *graf* koja opisuje povezanost sustava; tipično pomoću grafova modeliramo transportne ili komunikacijske sustave, električne ili internetske mreže, ali i molekule i drugo.

Definicija 1.0.1. *Graf je uređen par $G = (V, E)$ gdje je $\emptyset \neq V = V(G)$ skup vrhova (eng. vertex), a $E = E(G)$ skup bridova (eng. edge), a svaki brid $e \in E$ spaja dva vrha $u, v \in V$ koje nazivamo krajevima od e.*

Definicija 1.0.2. *Stupanj vrha v u grafu G je broj bridova grafa G incidentnih s v, pri čemu se svaka petlja računa kao dva brida*

Definicija 1.0.3. *Jednostavan graf je graf koji nema petlji ni dva brida koja spajaju isti par vrhova.*

Definicija 1.0.4. *Šetnja u grafu G je niz $W : v_0e_1v_1e_2 \cdot \ldots \cdot e_hv_h$ čiji članovi su naizmjenično vrhovi v_i i bridovi e_i, tako da su krajevi od e_i vrhovi v_{i-1} i v_i $(1 \leq i \leq h)$.*

U **jednostavnom grafu** šetnja je potpuno određena samo nizom vrhova $v_0, v_1, \ldots, v_h$. Kažemo da je v_0 početak, a v_h kraj šetnje W ili da je W šetnja od v_0 do v_h ili (v_o, v_h) šetnja. Šetnja W je **zatvorena** ako je $v_o = v_h$. Ako su bridovi $e_1, e_2, \ldots, e_h$ šetnje W međusobno različiti, onda se W zove **staza**. Ako su, osim bridova, i svi vrhovi međusobno različiti, onda se zove **put**. Zatvorena staza pozitivne duljine čiji su vrhovi (osim krajeva) međusobno različiti zove se **ciklus**.

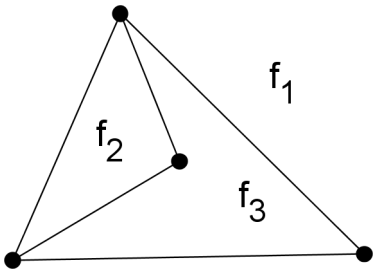
Za graf kažemo da je **povezan** ako postoji put između svaka dva vrha.

Definicija 1.0.5. *Stablo je povezan graf bez ciklusa. Razapinjuće stablo grafa G je razapinjući podgraf (sadrži sve vrhove) koje je stablo.*

Kažemo da je graf **planaran** ako se može nacrtati u ravnini tako da mu se bridovi sijeku samo u vrhovima, odnosno ako postoji funkcija f koja svakom vrhu v grafa pridružuje

točku u ravnini $\mathbb{R}^2$, a svakom bridu e grafa jednostavnu krivulju $f(e) \subset \mathbb{R}^2$ tako da se $f(e_1)$ i $f(e_2)$ sijeku u točki T ako i samo ako je $T = f(v)$, za neki vrh incidentan s e_1 u e_2 u G.

Planarni graf G dijeli $\mathbb{R}^2 \setminus G$ na područja. Zatvorenja tih područja zovu se **strane**. $F(G)$ je skup svih strana (eng. *faces*) planarnog grafa G. Svaki planarni graf ima točno jednu neomeđenu stranu, zovemo je vanjska strana.



Slika 1.1: Planarni graf

Teorem 1.0.6. *Neka je G povezan ravninski graf. Tada je Eulerova karakteristika od G*

$$\chi(G) = v(G) - e(G) + f(G) = 2.$$

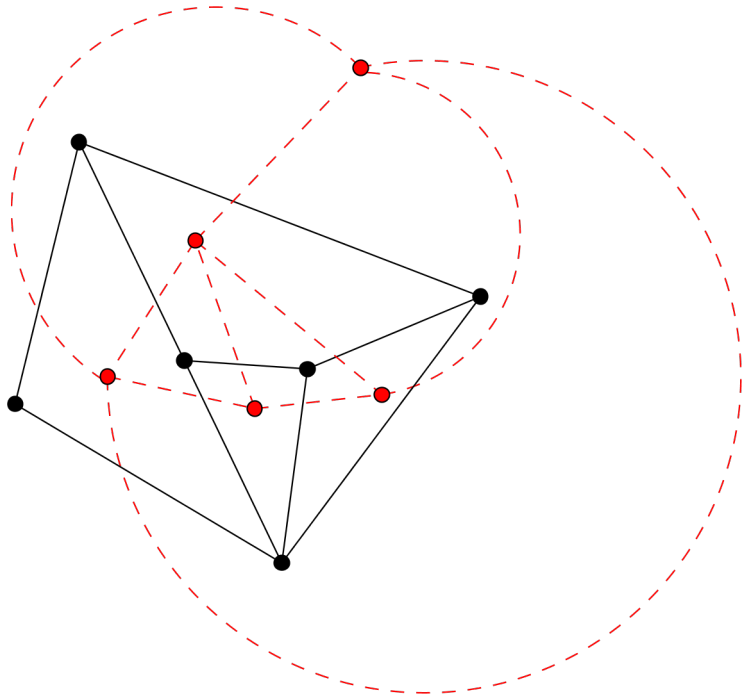
Dokaz. Navodimo skraćeni dokaz, za potpuni dokaz čitatelja upućujemo na [8]. Indukcijom po broju bridova $e(G)$. Ako $e(G) = 0$, onda je broj vrhova $v(G) = 1$ i broj strana $f(G) = 1$, pa formula vrijedi. Neka je $e(G) \geq 1$. Ako je G stablo, onda se može pokazati da je $v(G) = e(G) + 1$ i $f(G) = 1$, pa formula vrijedi. Ako G nije stablo, onda ima ciklus. Neka je $e \in E(G)$ brid nekog ciklusa i promatrajmo $G - e$ (graf dobiven brisanjem brida e iz grafa G). Povezani ravninski graf $G - e$ ima $v(G)$ vrhova, $e(G) - 1$ bridova i $f(G) - 1$ strana. Po pretpostavci indukcije je $v(g) - (e(G) - 1) + (f(G) - 1) = 2$, pa slijedi formula. □

Definicija 1.0.7. *Dualni graf G^* planarnog grafa G nastaje tako da svakoj strani f grafa G pridružimo vrh $f^* \in V(G^*)$, a svakom bridu $e \in E(G^*)$ brid $e^* \in E(G^*)$ pri čemu vrijedi pravilo: Dva vrha f^* i g^* spojena su bridom e^* u G^* ako i samo ako su strane f i g separirane bridom e u G.*

Na slici 1.2 crnom bojom je prikazan ravninski graf G, a crvenom bojom njegov dualni graf G^*.

Teorem 1.0.8. *Dualni graf povezanog planarnog grafa je povezan.*

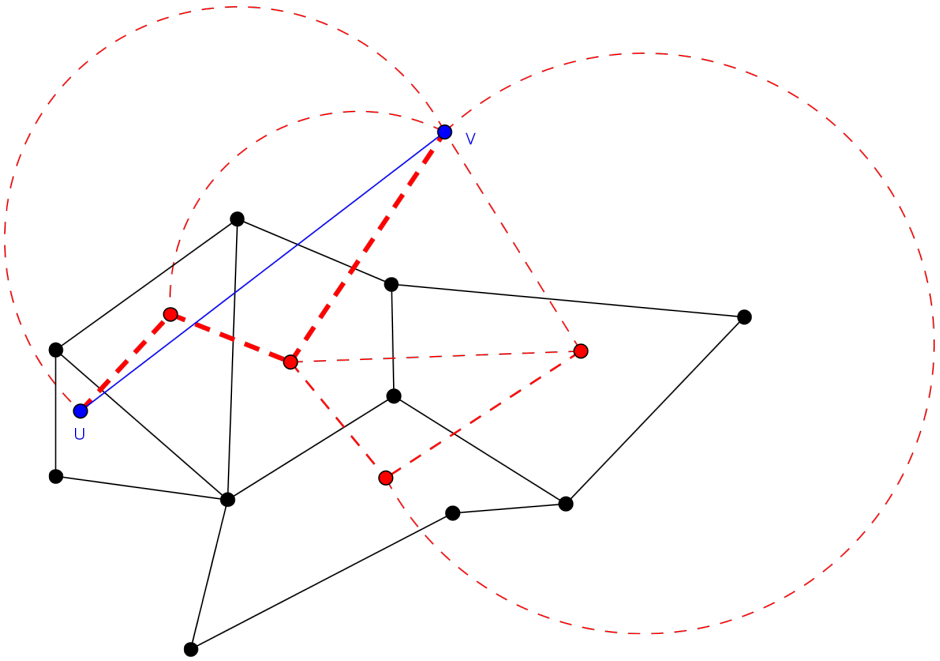
Dokaz. Neka je G bilo koje (omeđeno) ulaganje zadanog planarnog grafa u ravninu. Neka je G^* dualni graf povezanog planarnog grafa G i V skup svih vrhova grafa G. Na slici



Slika 1.2: Graf s pripadnim dualnim grafom

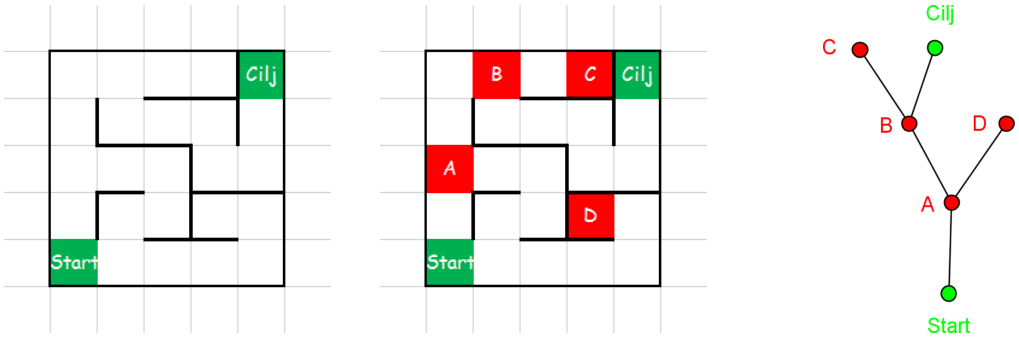
1.3 crnom bojom je prikazan ravninski graf G, a crvenom bojom njegov dualni graf G^*. Uočimo da je dovoljno pokazati da postoji put od proizvoljne unutrašnje strane do vanjske strane. Neka je $U \in f$ proizvoljna točka unutar proizvoljne unutrašnje strane f. Očito postoji beskonačno mnogo pravaca kroz točku U, ali konačno mnogo pravaca koji prolaze kroz U i vrhove grafa. Stoga postoji beskonačno mnogo pravaca kroz U koji ne prolaze kroz vrhove grafa G. Iz toga i činjenice da je f unutrašnja strana grafa slijedi da postoji pravac kroz U koji siječe neke bridove grafa G. Nadalje, kako je pravac neomeđen, a G omeđen, na tom pravcu postoji bar jedna točka V koja je u vanjskoj strani grafa. Na slici 1.3 takav pravac je označen plavom bojom. Iz definicije 1.0.7 sad slijedi da postoji put od proizvoljne unutrašnje strane f do vanjske strane. Na slici 1.3 takav put je označen podebljanim crvenim isprekidanim linijama. □

Teorija grafova je vrlo korisna grana matematike što se tiče labirinta. Osim što ju možemo koristiti prilikom generiranja labirinata, ona nam omogućuje zorni prikaz. Vrlo kompliciranе labirinte možemo jasno prikazati te rješenje učiniti očitim. S druge strane, jednostavne labirinte možemo otežati dodajući vrhove i bridove. Svaki labirint možemo reprezentirati grafom na način da križanja i slijepe završetke u labirintu poistovjetimo sa vrhovima u grafu, a hodnike (odnosno putove) unutar labirinta sa bridovima grafa. Primjer



Slika 1.3: Povezani dualni graf

izrade grafa koji reprezentira labirint se nalazi na slici 1.4. Takav graf zvat ćemo **grafom labirinta**. Uočimo da u takvom grafu moraju postojati bar dva istaknuta vrha stupnja 1 (ulaz i izlaz iz labirinta), no graf labirinta, kako ćemo vidjeti u primjerima, ne mora biti jednostavan.



(a) Labirint (b) Ključne točke (c) Graf

Slika 1.4: Reprezentacija labirinta grafom

Poglavlje 2

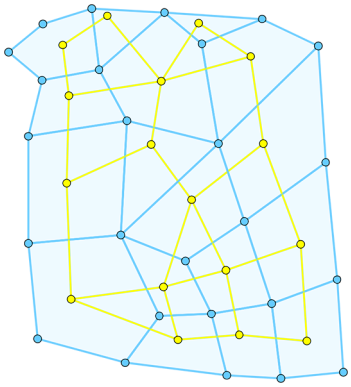
Algoritmi za generiranje labirinata

Postoji mnogo različitih algoritama za generiranje labirinata i svaki od njih ima svoje karakteristike. Neke od karakteristika su udio slijepih završetaka, vrijeme i memorija potrebna za generiranje, ali i načini generiranja. Labirinte možemo generirati dodajući zidove na praznom predlošku, mičući zidove na punom predlošku ili kombinacijom te dvije metode.

U daljnjem radu opisat ćemo četiri algoritma koja se relativno lako mogu iskoristiti u stvarnom životu, bilo da rukom izrađujemo labirint ili ga generiramo pomoću računala. Algoritmom dualnog grafa najlakše možemo vizualizirati cjelokupni proces generiranja labirinata. Algoritam za pretraživanje s prvenstvom po dubini (*depth-first search*) je jedan od najčešćih algoritama, dok su Kruskalov i Primov dva najpoznatija algoritma za traženje minimalnog razapinjućeg stabla u težinskom grafu. Joseph Bernard Kruskal je objavio svoj algoritam 1956. godine. Kako to često u matematici biva, pojedini teoremi i pravila, a u ovom slučaju algoritmi, se zovu po stručnjaku koji je poznatiji i eksponiraniji, a ne prema onom koji je prvi došao te spoznaje. Primov algoritam je jedan od tih slučaja. Taj algoritam je razvio češki matematičar Vojtěch Jarník 1930. godine. Američki računalni stručnjak i matematičar Robert C. Prim razvija isti algoritam 1957. godine, dok ga nizozemski računalni stručnjak Edsger W. Dijkstra opisuje 1957. godine. Primov algoritam se u literaturi spominje i pod nazivima Jarníkov algoritam, Prim-Jarníkov algoritam ili DJP algoritam.

2.1 Algoritam dualnog grafa

Labirint možemo generirati koristeći predložak sa ćelijama. Ćelije mogu biti pravilne (papir sa kvadratićima) ili nepravilne (poput ovoga na slici 2.1). Rubovi ćelija su ujedno i



Slika 2.1: Predložak s ćelijama

zidovi našeg labirinta i čine bridove plavo obojanog grafa. Svaku ćeliju možemo promatrati kao vrh grafa. Spojnice susjednih ćelija (vrhova) su bridovi žuto obojenog grafa.

Iz opisa je vidljivo da ovdje u paru promatramo planaran graf i njemu dualan graf, do na ignoriranje vanjske strane: Predložak sa ćelijama prikazan na slici 2.1 je nastao tako što smo plavom grafu G našli njemu dualan graf G^* (definicija 1.0.7) zanemarujući vanjsku stranu plavog grafa (jer nas zanima samo ono što se nalazi unutar labirinta).

Neka je E skup bridova ravninskog smještenja planarnog grafa G, F skup njegovih strana bez vanjske strane te V^* skup vrhova pripadnog dualnog grafa bez vrha koji odgovara vanjskoj strani. Tada algoritam možemo opisati pseudokodom:

```
P = ∅
Odaberi proizvoljan vrh t ∈ V* i proglasi ga ulazom u labirint.
P = P ∪ {t}
V* = V* \ {t}

while (V* ≠ ∅)
{
    if(u skupu V* postoji vrh v' koji je susjedan vrhu t)
    {
        Odaberi proizvoljni vrh v' ∈ V* koji je susjedan vrhu t
        Izbaci brid e koji separira strane u kojima se nalaze
        vrhovi v' i t iz skupa E
    }
    else
    {
        t = proizvoljan vrh iz skupa P koji ima susjedan vrh iz skupa V*
        Odaberi proizvoljni vrh v' ∈ V* koji je susjedan vrhu t
        Izbaci brid e koji separira strane u kojima se nalaze
```

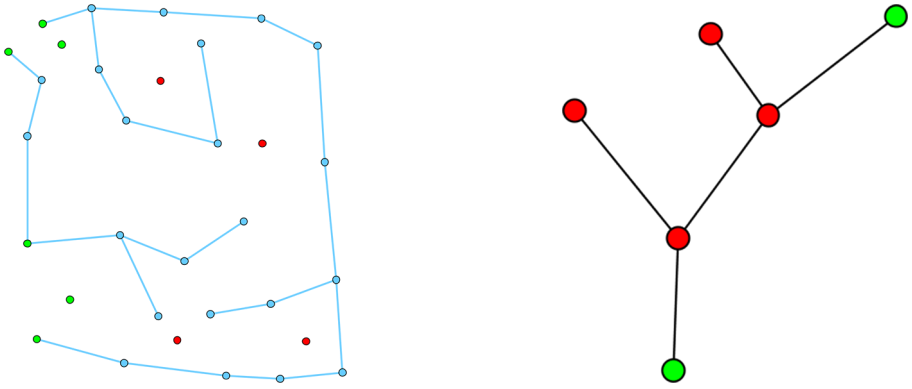
```
            vrhovi v' i t iz skupa E
        }
        P = P ∪ {v'}
        V* = V* \ {v'}
        t = v'
}

skup E je skup unutrasnjih zidova labirinta.
```

Algoritam možemo opisati i riječima. U situaciji kao na slici 2.1, plavi graf predstavlja moguće zidove labirinta, a žuti dualni graf predstavlja moguće putove unutar tog labirinta.

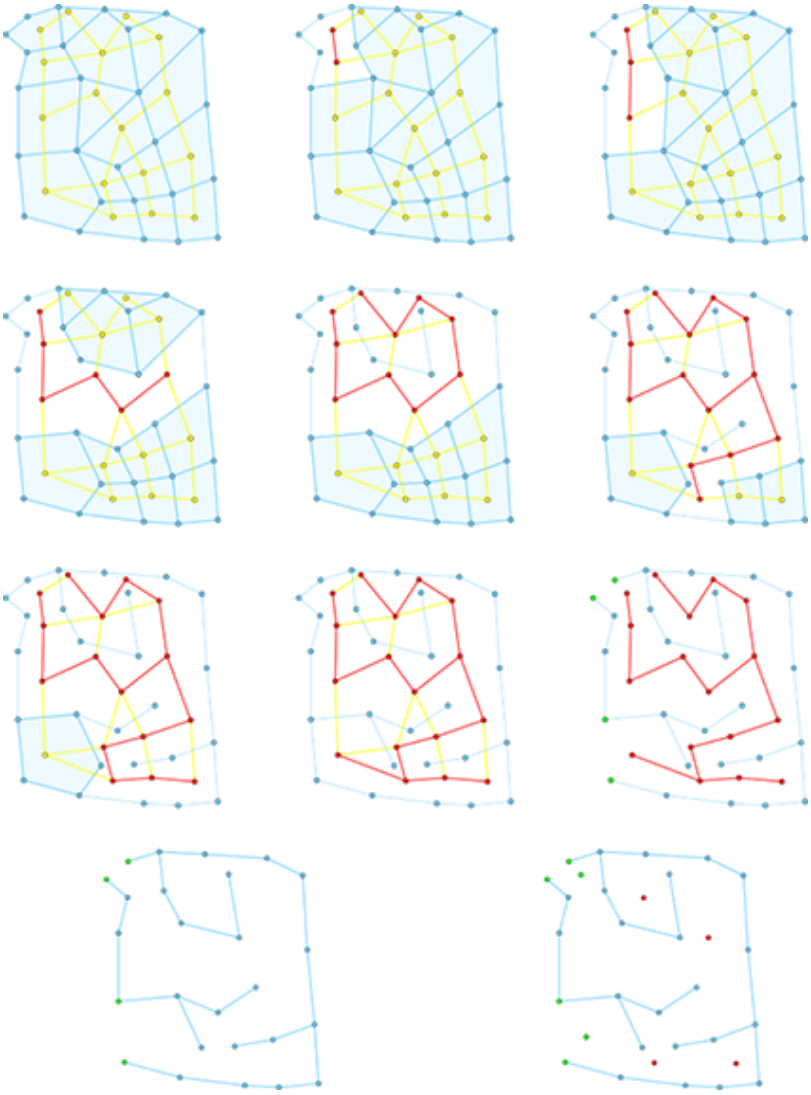
Po volji odabran žuti vrh nam predstavlja start, odnosno ulaz u labirint. Krećemo se po bridovima žutog grafa. Svaki put kada presiječemo plavi graf, obrišemo plavi brid (napravili smo prolaz u zidu) i nastavimo dalje sve dok ili ne prođemo kroz sve vrhove žutog grafa (u tom slučaju zadnji vrh nam je izlaz iz labirinta) ili se ne zaustavimo u nekom vrhova i taj vrh proglasimo izlazom. Ključni koraci prilikom nastajanja labirinta su prikazani na slici 2.2.

Na slici 2.3 je usporedno prikazan zadnji korak sa slike 2.2 te graf koji reprezentira labirint koji smo upravo generirali. Primijetimo plavo obojane zidove labirinta te ukupno 6 točaka na slici 2.3a. Dvije zelene točke nam predstavljaju ulaz, odnosno izlaz iz labirinta, dok nam crvene točke predstavljanju križanja unutar našeg labirinta. Sad lako uočavamo kako je nastao graf sa slike 2.3b. Vrhovi u grafu nam predstavljaju križanja, a bridovi nam predstavljaju hodnike.



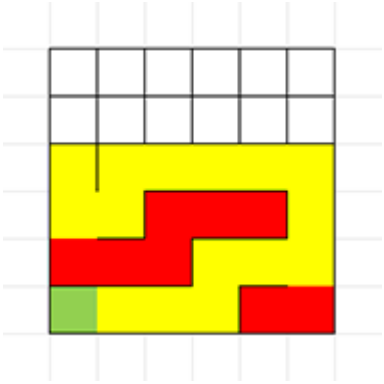
(a) Labirint (b) Graf

Slika 2.3: Reprezentacija labirinta grafom



Slika 2.2: Koraci algoritma dualnog grafa

2.2 Pretraživanje s prvenstvom po dubini (*depth-first search*)



Slika 2.4: Predložak za ilustraciju algoritma

Ovaj algoritam je jedan od najjednostavnijih i najčešćih algoritama za generiranje labirinta pomoću računala. Generira jednostavnije labirinte sa malo križanja, ali je memorijski vrlo zahtjevan. Sastoji se od samo dva koraka, ali je potrebno pamtiti koja polja koja smo već posjetili i cijeli labirint prolazimo dva puta. Postupak u kojemu se vraćamo na već posjećeno polje nazivamo *backtracking*, odnosno unatražno pretraživanje. Ovim postupkom osiguravamo da smo sva polja u labirintu maksimalno istražili, odnosno da ne postoji neko polje u rešetki koje nije dio labirinta. Slično kao i kod prethodnog algoritma, krećemo od potpuno ispunjene rešetke, no ovdje imamo labirint čiji je svaki hodnik omeđen sa četiri zida.

Pravila za kretanje su sljedeća:

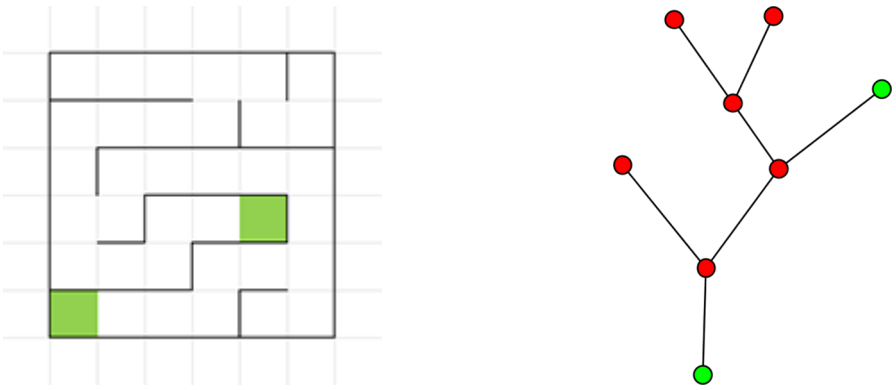
- Ako je moguće, odabiremo susjedno polje u kojem još nismo bili i brišemo zid između;
- Ako ne možemo preći u susjedno polje u kojem još nismo bili, vratimo se na prethodno polje (eng. *backtracking*).

Ovim jednostavnim pravilima smo osigurali da prođemo kroz sva polja točno dva puta. Prvo odabrano polje je izlaz iz labirinta, a ulaz po volji odaberemo.

Pojedini koraci su ilustrirani na slikama 2.5 i 2.6. Zeleno obojane ćelije predstavljaju ulaz i izlaz iz labirinta. Ćelije koje smo jednom posjetili označavamo žutom bojom, dok sa crvenom bojom označavamo ćelije koje smo posjetili dva puta.

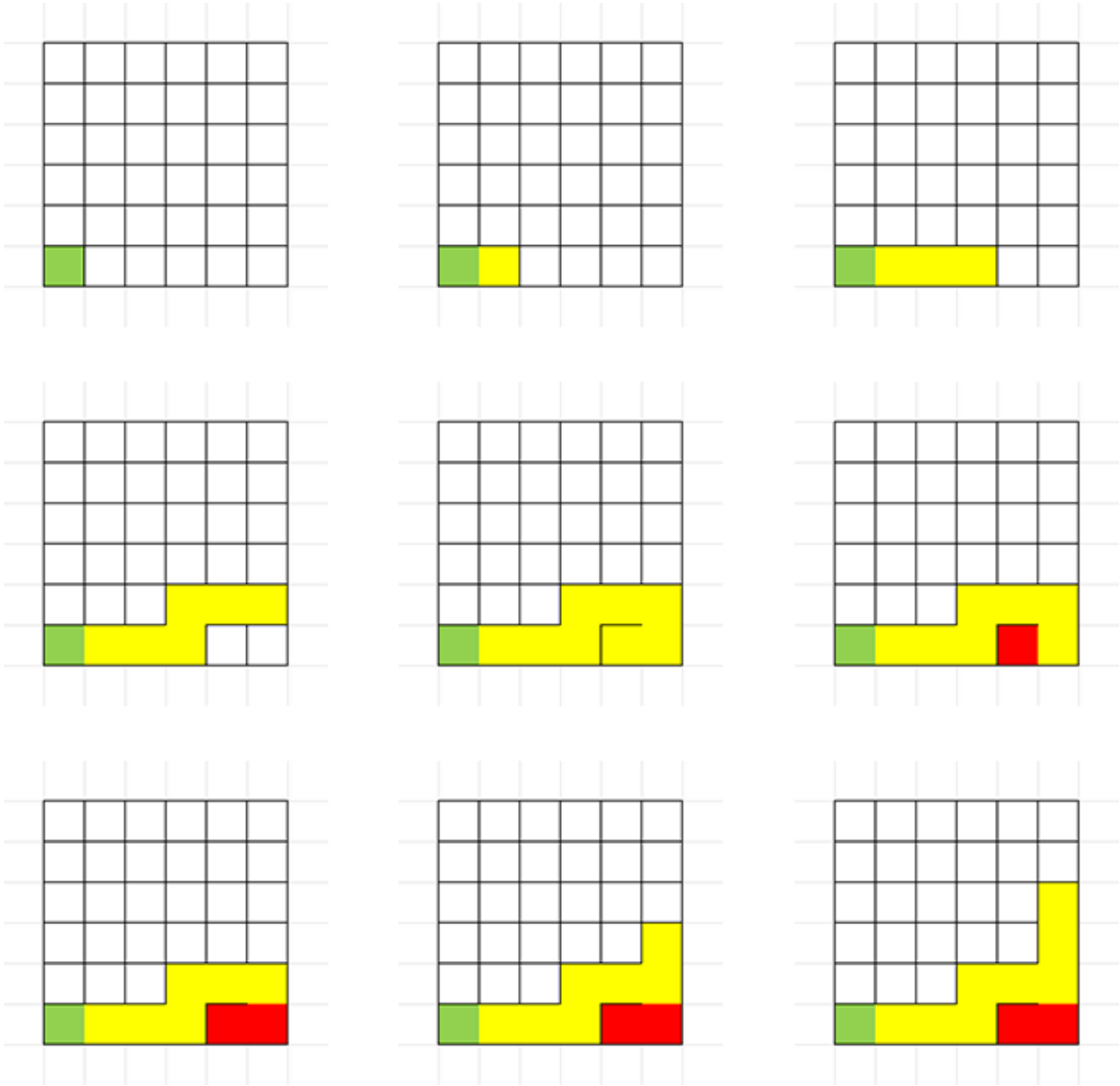
Zamislimo da imamo tablicu sa 36 polja (rešetka tipa 6×6). Svako polje ima 4 zida. Odaberemo jedno rubno polje i to polje proglasimo izlazom. U ovom trenutku se nalazimo na prvom koraku na slici 2.5. Proizvoljno odaberemo jedno susjedno polje. To nam je prvi ulazak u to polje pa maknemo zid između ta dva polja. Zatim opet odaberemo neko susjedno polje. Ako smo već posjetili to polje, odaberemo neko drugo susjedno polje u kojem nismo bili. Prijeđemo u to drugo polje i maknemo zid između. Algoritam se ponavlja sve dok ne dođemo na takvo polje koje nema susjednih polja u kojima nismo bili. Nalazimo se u situaciji ilustriranoj u šestom koraku na slici 2.5. U ovom trenutku provodimo *backtracking* i označavamo ćelije crvenom bojom. U te ćelije se nikada ne vraćamo. Zatim se vratimo jedno polje nazad te ponovo pokušamo provesti prethodno opisani algoritam. Ako ga ne možemo provesti, opet se vratimo jedno polje unazad. *Backtracking* provodimo ili dok ne dođemo na polje u kojem možemo provesti naš algoritam (deveti korak na slici 2.5) ili dok se ne vratimo na početno polje (šesnaesti korak na slici 2.5). Sad nam još samo preostaje odrediti početnu poziciju. Nju po volji odaberemo od preostalih polja.

Slično kao i kod prethodnog algoritma i ovdje ćemo reprezentirati labirint pomoću grafa (slika 2.7). Podsjetimo se, križanja u labirintu su vrhovi, a putovi su bridovi. Primijetimo kako je iz grafa lako iščitati neke karakteristike labirinta poput broja hodnika, križanja, slijepih završetaka, ... Te podatke možemo iskoristiti pri procjeni složenosti slučajno generiranog labirinta.

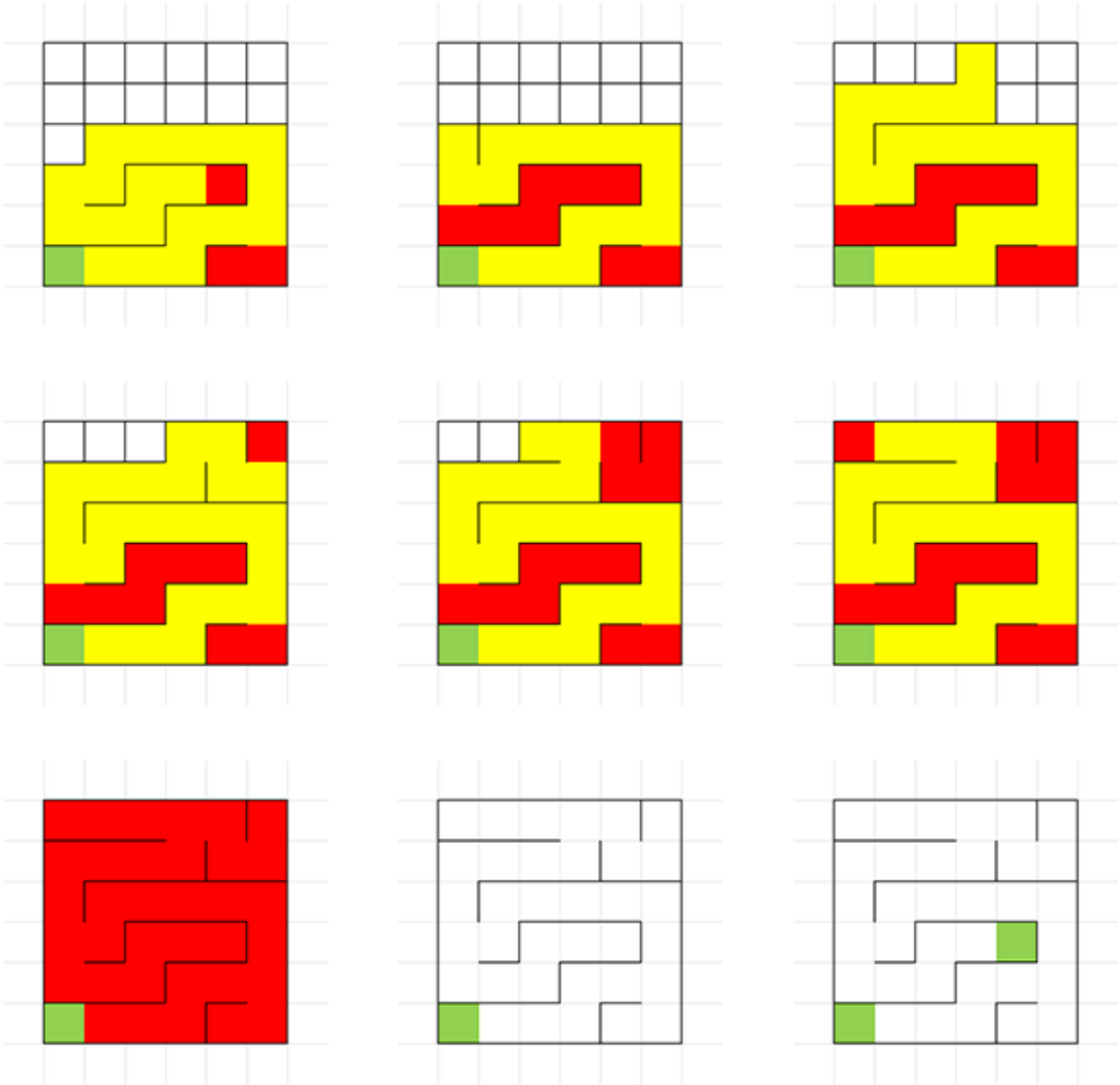


(a) Labirint (b) Graf

Slika 2.7: Reprezentacija labirinta grafom

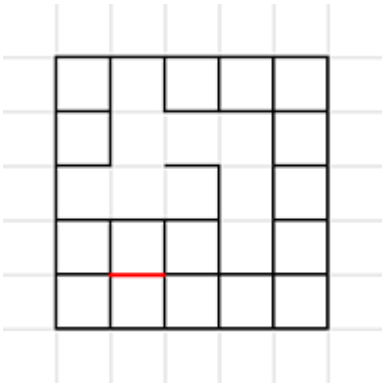


Slika 2.5: Koraci 1 - 9



Slika 2.6: Koraci 10 - 18

2.3 Primjena Kruskalovog algoritma na generiranje grafova



Slika 2.8: Predložak za ilustraciju algoritma

Kruskalov algoritam je jedan od dva klasična algoritma za pronalaženje *minimalnog* razapinjućeg stabla u težinskom grafu[1]. U ovom poglavlju ćemo koristiti Kruskalov algoritam izostavljajući uvjet minimalnosti. Pritom kao u prethodnom poglavlju krećemo od pravokutne tablice ćelija. Zanemarujemo veličinu pojedinih ćelija (hodnika), odnosno duljine bridova u grafu. Algoritam generira labirinte tako što briše zidove ako zadovoljavaju određeni kriterij, tj. brišemo brid ako on separira dvije strane.

Kruskalov algoritam provodimo na pravokutnoj tablici pa ćemo brid koji separira dvije strane definirati na sljedeći način.

Definicija 2.3.1. *Neka je* $Z = \{z_1, z_2, z_3, \ldots, z_i\}$ *skup svih unutrašnjih zidova i* $C = \{c_1, c_2, c_3, \ldots, c_j\}$ *skup svih ćelija. Svaki zid* $z_x \in Z$ *se nalazi između dvije ćelije* $c_{lijeva}(z)$ *i* $c_{desna}(z)$. $c_{lijeva}(z), c_{desna}(z) \in C$. *Kažemo da su ćelije* c_{lijeva} *i* c_{desna} *spojene ako možemo iz jedne doći u drugu.*

[1]Težinski grafovi razlikuju se od bezteţinskih po tome što svaki brid ima pridruženu težinu prolaska. Preciznije, težinski graf je uređeni par $(G; \omega)$, gdje je $G = (V; E)$ graf, a $\omega : E \to R + 0$ funkcija koju nazivamo težinskom funkcijom. Minimalno razapinjuće stablo u težinskom grafu je razapinjuće stablo s najmanjim mogućim zbrojem težina bridova.

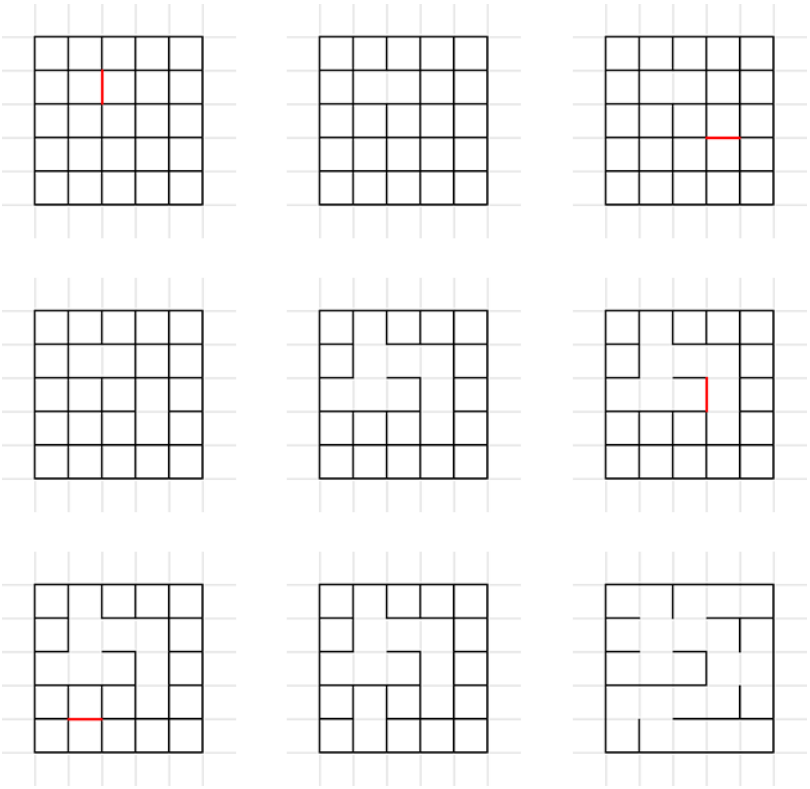
Sada možemo algoritam opisati sljedećim pseudokodom:

```
while |L| > 1
{
    Odaberi proizvoljan z ∈ Z;
    if (c_lijeva(z) nije spojena sa c_desna(z))
        {
            izbrisi zid z iz skupa Z;
            izbrisi celiju c_desna(z) iz skupa C;
        }
}
```

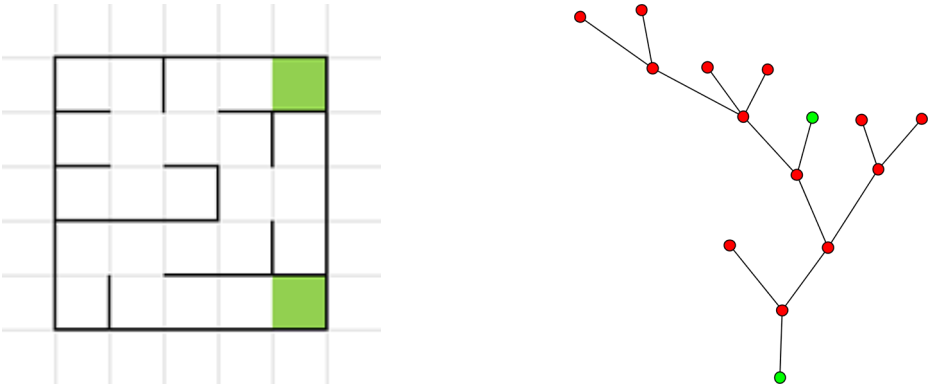
Provedimo algoritam na rešetki 5×5. Ključni koraci su ilustrirani na slici 2.9.

Odaberemo proizvoljan zid unutar labirinta. Taj zid se nalazi između dvije ćelije i ako one *nisu spojene*, onda izbrišemo taj zid. Trenutna situacija je ilustrirana u drugom koraku na slici 2.9. Ponavljamo istu radnju dok ne dođemo do slučaja ilustriranom u šestom koraku na slici 2.9, odnosno dok ne odaberemo zid koji dijeli dvije *spojene* ćelije. U tom slučaju jednostavno izaberemo neki drugi zid. Algoritam provodimo sve dok ne dođemo u situaciju ilustriranom u devetom koraku na slici 2.9. Svaki zid koji odaberemo dijeli dvije spojene ćelije. Tu je kraj algoritma i sad nam preostaje samo odabrati dvije ćelije i proglasiti ih ulazom, odnosno izlazom iz labirinta.

Ukoliko za ulaz i izlaz odaberemo ćelije označene zelenom bojom na slici 2.10a, labirint je reprezentriran grafom na slici 2.10b.



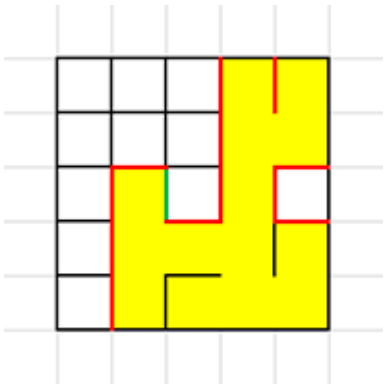
Slika 2.9: Koraci Kruskalovog algoritma



(a) Labirint (b) Graf

Slika 2.10: Reprezentacija labirinta grafom

2.4 Primjena Primovog algoritma na generiranje grafova



Slika 2.11: Predložak za ilustraciju algoritma

Drugi klasični algoritam za pronalaženje *minimalnog* razapinjućeg stabla u težinskom grafu je Primov algoritam. U ovom poglavlju ćemo također izostaviti uvjet minimalnosti te koristiti definiciju 2.3.1.

Neka je $Z = \emptyset$ skup svih unutrašnjih zidova i $C = \emptyset$ skup svih ćelija koje tvore labirint. Odaberemo proizvoljnu ćeliju i proglasimo je dijelom labirinta, a zidove te ćelije proglasimo unutrašnjim zidovima.

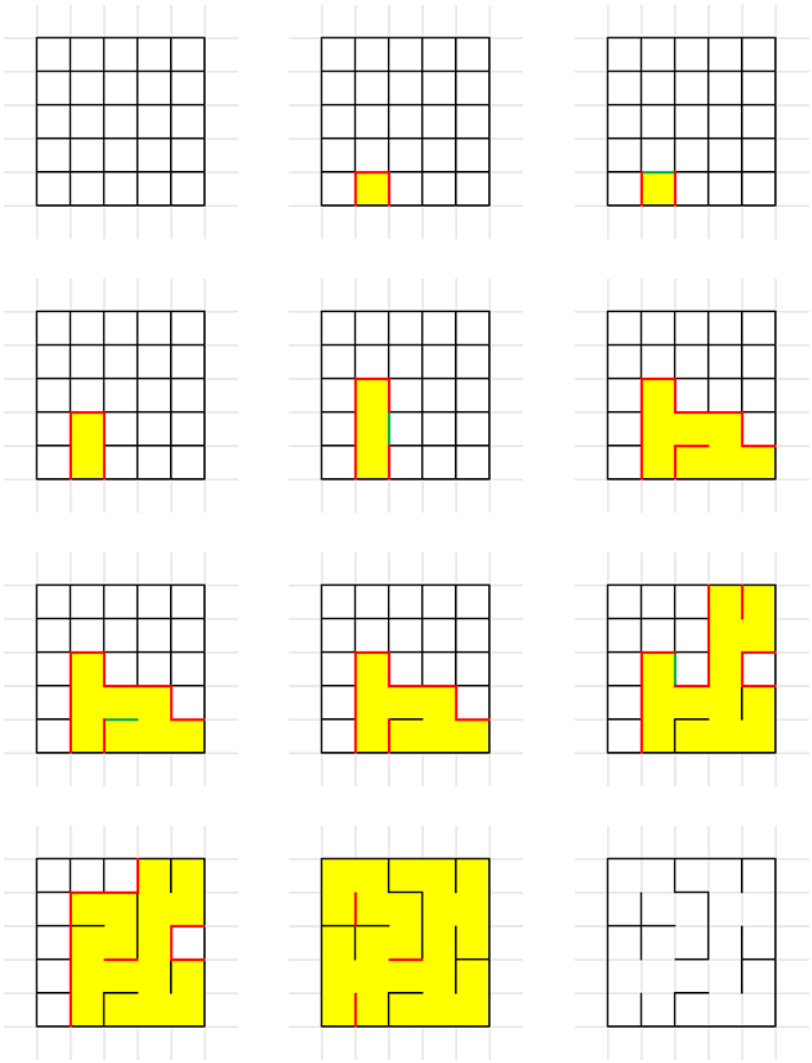
Uz prethodne početne uvjete, algoritam je prikazan sljedećim pseudokodom:

```
while (Z ≠ ∅)
{
    Odaberi proizvoljan z ∈ Z;
    if (c_lijeva(z) nije spojena sa c_desna(z))
    {
        dodaj susjednu celiju u skup C;
        dodaj zidove susjedne celije u skup Z;
    }
    izbrisi zid z iz skupa Z;
}
```

Provedimo algoritam na rešetki 5×5. Ključni koraci su ilustrirani na slici 2.12.

Odaberemo proizvoljnu ćeliju na rešetki te ju proglasimo dijelom labirinta, a njezine zidove proglasimo unutrašnjim zidovima labirinta. Ćeliju označimo žutom bojom, a zidove crvenom bojom. Zatim odaberemo neki unutrašnji zid i provjerimo što se nalazi sa druge strane. Nalazimo se na trećem koraku na slici 2.12, a zelenom bojom smo označili zid koji smo odabrali. Ako je s druge strane ćelija koja ne pripada labirintu pripojimo je

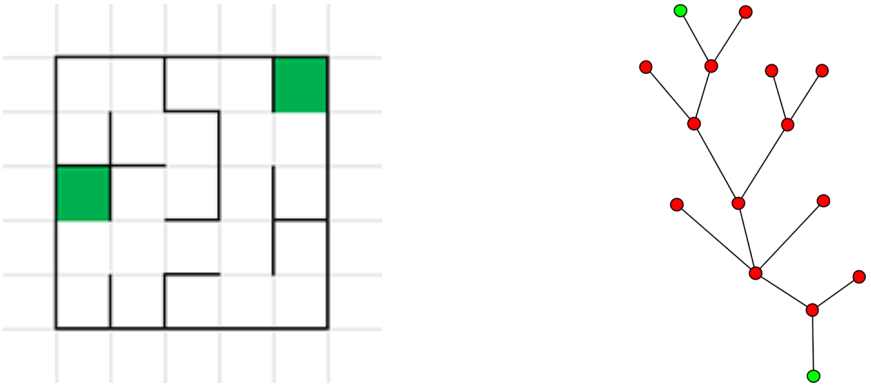
zajedno sa njezinim zidovima, a odabrani zid rušimo i izbacujemo ga iz skupa unutrašnjih zidova. Nalazimo se na četvrtom koraku na slici 2.12 te ponavljamo postupak sve dok se ne nađemo u situaciji prikazanoj na sedmom koraku na slici 2.12. Proizvoljno smo odabrali zid (označen zelenom bojom), ali sa druge strane se nalazi ćelija koja već pripada labirintu. U tom slučaju izbacimo promatrani zid iz skupa unutrašnjih zidova, ali ga ne rušimo. On je i dalje unutrašnji zid, ali ga više nikad ne možemo odabrati jer se više ne nalazi u skupu Z. Taj zid je označen crnom bojom. Algoritam provodimo dok sve ćelije ne budu uključene u labirint, odnosno dok ne „potrošimo" sve zidove iz skupa Z.



Slika 2.12: Koraci Primovog algoritma

Primijetimo, razlika između Primovog i Kruskalovog algoritma je u odabiru zidova

koji ćemo izbrisati. Kod Kruskalovog algoritma možemo izbrisati bilo koji zid, dok kod Primovog algoritma biramo zid iz puno manjeg skupa. Konkretno kod naših primjera, u prvom koraku Kruskalovog algoritma $|Z| = 60$, a kod Primovog $|Z| \leq 4$. Ako bismo promatrali usporednu animaciju oba algoritma, uočili bi da Kruskalov algoritam pripaja ćelije koje mogu, ali i ne moraju biti susjedne. Ako bismo prekinuli izvođenje algoritma, mogli bismo dobiti nepovezani graf koji reprezentira labirint. To nije slučaj kod Primovog algoritma. On pripaja isključivo susjedne ćelije i u svakom trenutku graf koji reprezentira labirint je povezan. Iz tog razloga labirinti generirani Kruskalovim algoritmom u pravilu imaju više slijepih završetka nego labirinti generirani Primovim algoritmom.



(a) Labirint (b) Graf

Slika 2.13: Reprezentacija labirinta grafom

Poglavlje 3

Algoritmi za rješavanje labirinata

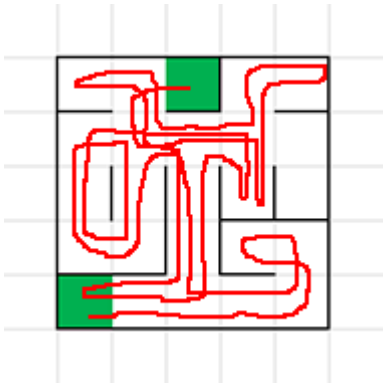
Istraživanje raznih metoda za rješavanje labirinata je vrlo popularno područje. Slično kao i algoritmi za generiranje labirinata, algoritmi za rješavanje labirinata se razlikuju po svojim karakteristikama. Neki od njih mogu naći samo jedno rješenje, a neki mogu i sva. Pojedini algoritmi mogu riješiti sve vrste labirinata dok neki samo uobičajene vrste. Također se razlikuju u brzini rješavanja i količinom potrebne memorije.

Opisat ću po pet mom mišljenju najzanimljivijih algoritama. Najtrivijalniji od njih je *random mouse* algoritam, odnosno metoda slučajnog odabira. Algoritam „slijedi zid" je uvjerljivo najpoznatiji algoritam, ali ne može riješiti sve vrste labirinata. John Pledge je sa samo 12 godina osmislio po njemu nazvan algoritam koji je brz, memorijski nezahtjevan, a ipak može riješiti većinu labirinata. Tréamauxov algoritam je zanimljiv jer ga možemo primijeniti i dok rješavamo labirint na papiru i u slučaju kada se fizički nalazimo u labirintu. Ukoliko rješavamo labirinte iz enigmatskih časopisa, vrlo često nam je najprikladniji algoritam pod nazivom algoritam „slijepih ulica".

3.1 Metoda slučajnog odabira

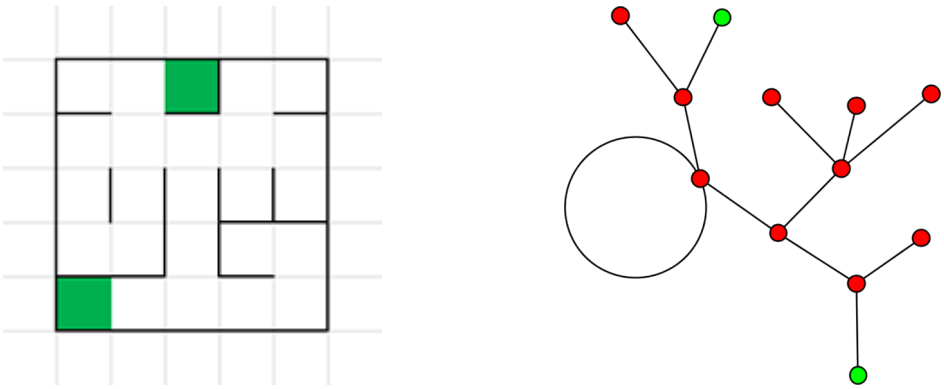
Engleski naziv ovog algoritma je *random mouse* i nije teško zaključiti zašto se tako zove. Doslovno se radi o besciljnom lutanju po labirintu nadajući se da ćemo kad tad naići na izlaz. Ovo je zasigurno najjednostavnija metoda i nakon dovoljno mnogo koraka će dati željeni rezultat, odnosno izlaz iz labirinta. Postoji samo jedno pravilo koje glasi „Na svakom križanju proizvoljno odaberi put kojim ćeš nastaviti". Put od ulaza do izlaza sigurno postoji (inače labirint nema smisla) pa postoji i teoretska mogućnost za izlazak iz labirinta. Dakle i netrenirani miš može izaći iz labirinta. Kad tad.

Na slici 3.2b uočavamo novu strukturu u grafu. Radi se o petlji i ona nam može biti vrlo



Slika 3.1: Ilustracija metode slučajnog odabira

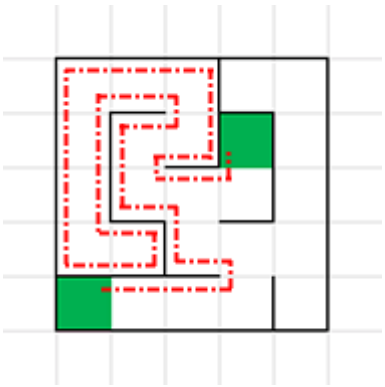
nezgodna prilikom rješavanja labirinta ako izaberemo neprikladan algoritam. Petlja nema nikakvog utjecaja na ovaj algoritam, no u sljedećim poglavljima ćemo vidjeti algoritme kojima petlja može predstavljati problem.



(a) Labirint (b) Graf

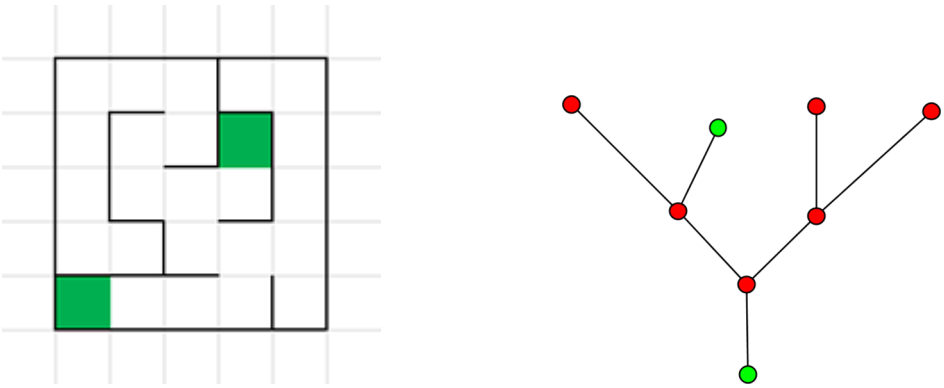
Slika 3.2: Reprezentacija labirinta grafom

3.2 Algoritam „slijedi zid” (*wall follower*)



Slika 3.3: Predložak s ćelijama

Ako prosječnog čovjeka pitamo da nam kaže neki algoritam za rješavanje labirinta, *Slijedi zid* je prvi (ako ne i jedini) algoritam kojeg se većina sjeti. Algoritam se svodi se na to da dotaknemo jedan zid (lijevi ili desni) i pratimo ga sve do izlaza. Odnosno na svakom križanju skrećemo uvijek u istu stranu (lijevu ili desnu). Poznat je još kao i pravilo lijeve (desne) ruke, ovisno o tome koju ruku stavimo na zid. Na slici 3.3 labirint je riješen pravilom lijeve ruke, no može se riješiti koristeći pravilo desne ruke.

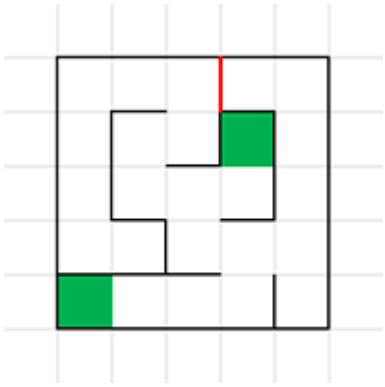


(a) Labirint (b) Graf

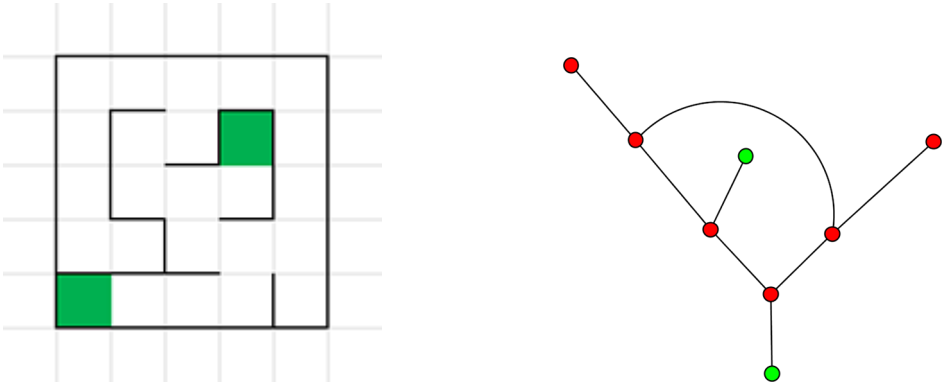
Slika 3.4: Reprezentacija labirinta grafom

Većinu labirinata koje susretnemo možemo riješiti koristeći ovaj algoritam, no uz vrlo male preinake dobivamo labirint koji ne možemo riješiti ovom metodom. Što bi se dogodilo ako maknemo crveno označeni zid na slici 3.5? Uspoređujući slike 3.4a i 3.6a na prvi pogled nismo napravili značajnu preinaku. No uspoređujući reprezentacije grafom (slika

3.4b i slika 3.6b) uočavamo bitnu razliku. Graf na slici 3.6b više nije jednostavno povezan. Stvorili smo jedan „otok" na koji moramo doći, a ne možemo. Dakle, ovaj algoritam će dati rješenje ako su unutrašnji zidovi kod izlaza spojeni sa vanjskim zidom labirinta. Inače, vratiti će nas na početak.



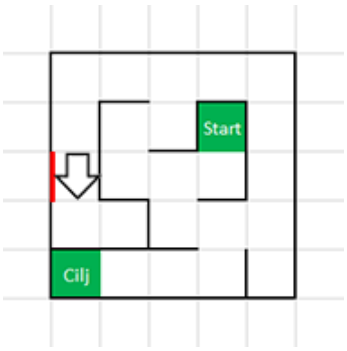
Slika 3.5: Preinaka labirinta



(a) Labirint (b) Graf

Slika 3.6: Reprezentacija labirinta grafom

3.3 Pledgeov algoritam



Slika 3.7: Predložak s ćelijama

Pledgeov algoritam možemo shvatiti kao nadogradnju algoritma „slijedi zid". Krećemo se labirintom na isti način, ali pamtimo smjer kretanja i dozvoljeno je maknuti ruku sa zida ako su određeni uvjeti ispunjeni. Smjer kretanja pratimo pomoću brojača. Za svako skretanje u desno, brojač povećamo za jedan, a za svako skretanje u lijevo brojač smanjimo za jedan. Smijemo maknuti ruku sa zida samo kad je brojač jednak nuli.

Sve dok se Pledgeov algoritam ne izvršava, brojač je na nuli, ne držimo ruku na zidu i pomičemo se po jedno polje unaprijed. Kada dođemo do prepreke, odnosno ispred nas je zid, stavimo desnu ruku na zid, okrenemo se lijevo, smanjimo brojač za 1 i pokrenemo algoritam.

Pledgeov algoritam je prikazan sljedećim pseudokodom:

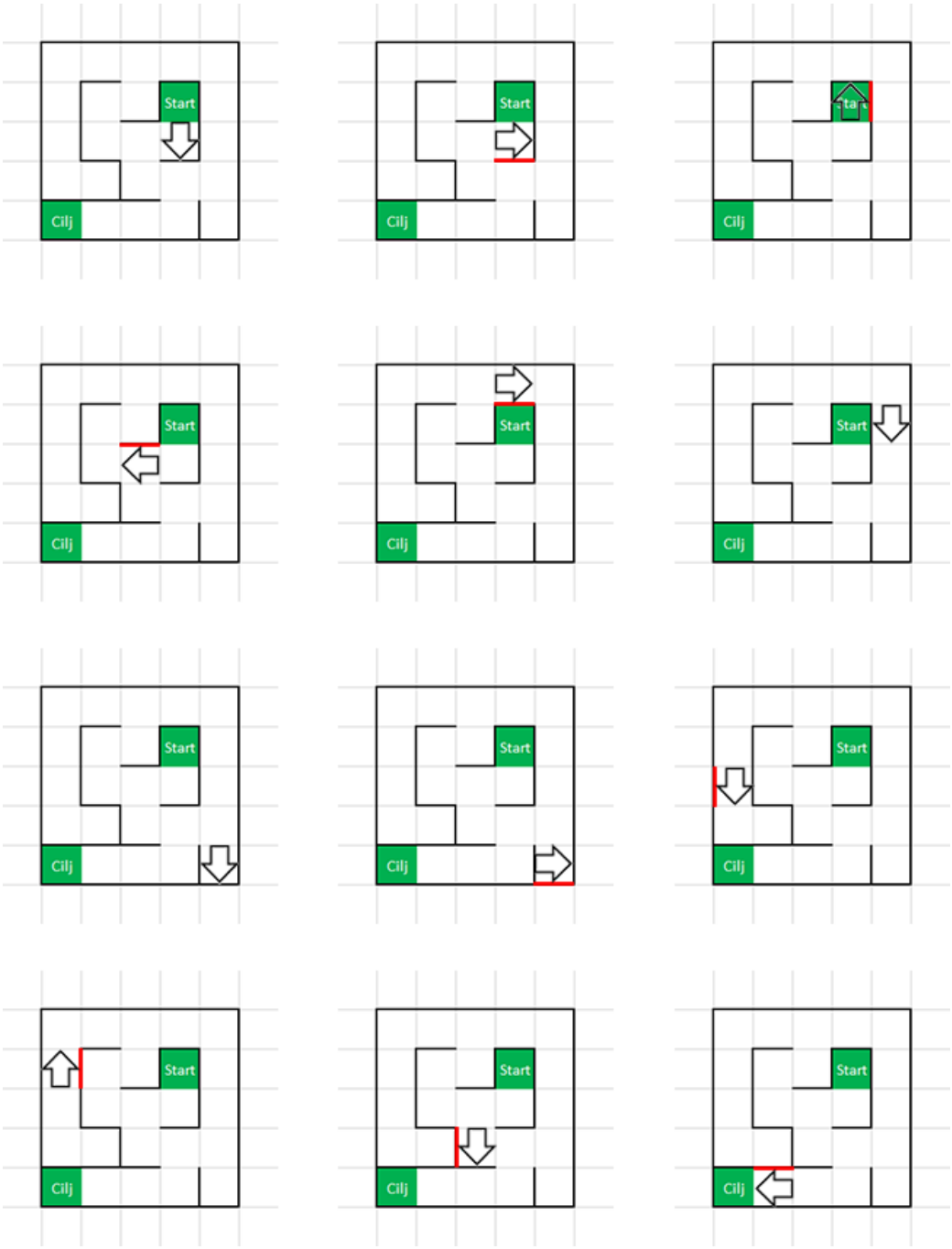
```
while(brojac != 0)
{
    if(nema zida sa desne strane)
    {
        okreni se desno i napravi korak naprijed;
        brojac++;
    }
    else if(postoji zid sa desne strane, a nema ga ispred)
    {
        napravi korak naprijed;
    }
    else if(postoji zid sa desne strane i postoji ispred)
    {
        okreni se lijevo;
        brojac--;
    }
}
```

Pokušajmo izaći iz labirinta koristeći Pledgeov algoritam. Ključni koraci su ilustrirani na slici 3.8. Strelica nam pokazuje u kojem smjeru smo okrenuti. Ako je zid sa desne strane strelice crvene boje to znači da ga držimo.

Algoritam se pokreće u prvom koraku na slici 3.8 jer je ispred nas zid. Brojač je na -1. Sada se ponašamo kao da koristimo *slijedi zid* algoritam, ali pamtimo dodatni parametar. Za svako skretanje ulijevo brojač smanjimo za 1, a za svako skretanje udesno brojač povećamo za jedan. Na petom koraku na slici 3.8 brojač nam je na -1 i pomaknemo se korak naprijed. Sa desne strane se ne nalazi zid, pa se okrenemo desno, napravimo korak naprijed i povećamo brojač za 1. U šestom koraku na slici 3.8 brojač je na 0, izlazimo iz Pledgeovog algoritma, mičemo ruku sa zida i pomičemo se po jedno polje unaprijed sve dok ili ne dođemo do prepreke (u tom slučaju ponovo pokrenemo Pledgeov algoritam) ili ne dođemo do izlaza. U osmom koraku na slici 3.8 opet pokrećemo algoritam i on se izvršava sve dok ne dođemo do izlaza.

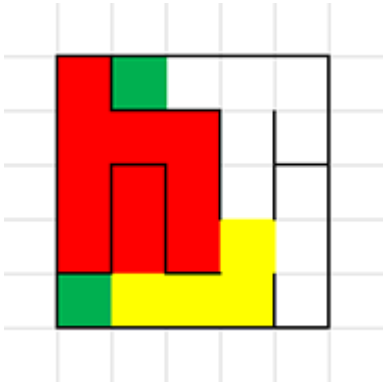
Slično kao i kod prethodnog algoritma, postoje dvije verzije Pledgeovog algoritma, desna i lijeva, ovisno sa kojom rukom diramo zid. Kod desnog Pledgeovog algoritma prvo skretanje je ulijevo. Kada dođemo do zida i stavimo desnu ruku na njega, fizički nam je prirodnije da krenemo lijevo. Ovim algoritmom uvijek možemo doći od točke S do točke C ako nam je točka S unutar labirinta, a C izvan labirinta, odnosno, ako smo zatočeni unutar labirinta i želimo izaći van. Pledgeov algoritam osigurava izlazak iz labirinta ukoliko je izlaz spojen sa vanjskim zidom, a ulaz se nalazi negdje unutar labirinta. Ukoliko zamijenimo start i cilj na slici 3.7, ovim algoritmom nećemo uspjeti doći do cilja.

Lijevi Pledgeov algoritam je analogan desnom. Umjesto prvog skretanja u lijevo, skrenemo u desno jer ako stavimo lijevu ruku na zid, prirodno je da skrenemo desno.



Slika 3.8: Koraci Pledgeovog algoritma

3.4 Tréamauxov algoritam



Slika 3.9: Predložak s ćelijama

Édouard Lucas [4] je pripisao ovu metodu M. Trémauxu, a H. E. Dudeney [2] daje sljedeći opis:

> Slijedi metoda za rješavanje bilo kojeg labirinta, zahvaljujući M. Trémauxu, ali je neophodno pažljivo označavati na neki način ulaze i izlaze na račvanjima. . . . „Novi" brid, odnosno vrh je onaj kojeg nismo još posjetili; „stari" brid, odnosno vrh je onaj kojim smo već prošli.
>
> - Nijednim bridom ne smijemo proći više od dva puta.
> - Kad dođemo do novog vrha, nastavljamo bilo kojim „novim" bridom.
> - Ako se nalazimo na novom bridu i dođemo do starog vrha, vratimo se istim putem.
> - Ako se nalazimo na starom bridu i dođemo do starog vrha, nastavljamo novim bridom, ako postoji. Inače nastavljamo starim.

Trémauxov algoritam je još jedan od jednostavnijih i učinkovitijih metoda rješavanja labirinta, a svodi se na označavanje putova kojim smo već prošli. Put može biti neoznačen, označen jednom ili označen dvaput. Dok se krećemo labirintom označavamo putove kojima se krećemo i slijedimo jednostavna pravila. Na križanju odabiremo put koji smo manje puta označili (nijednom ili jednom) i nikada ne odabiremo put koji smo označili dva puta.

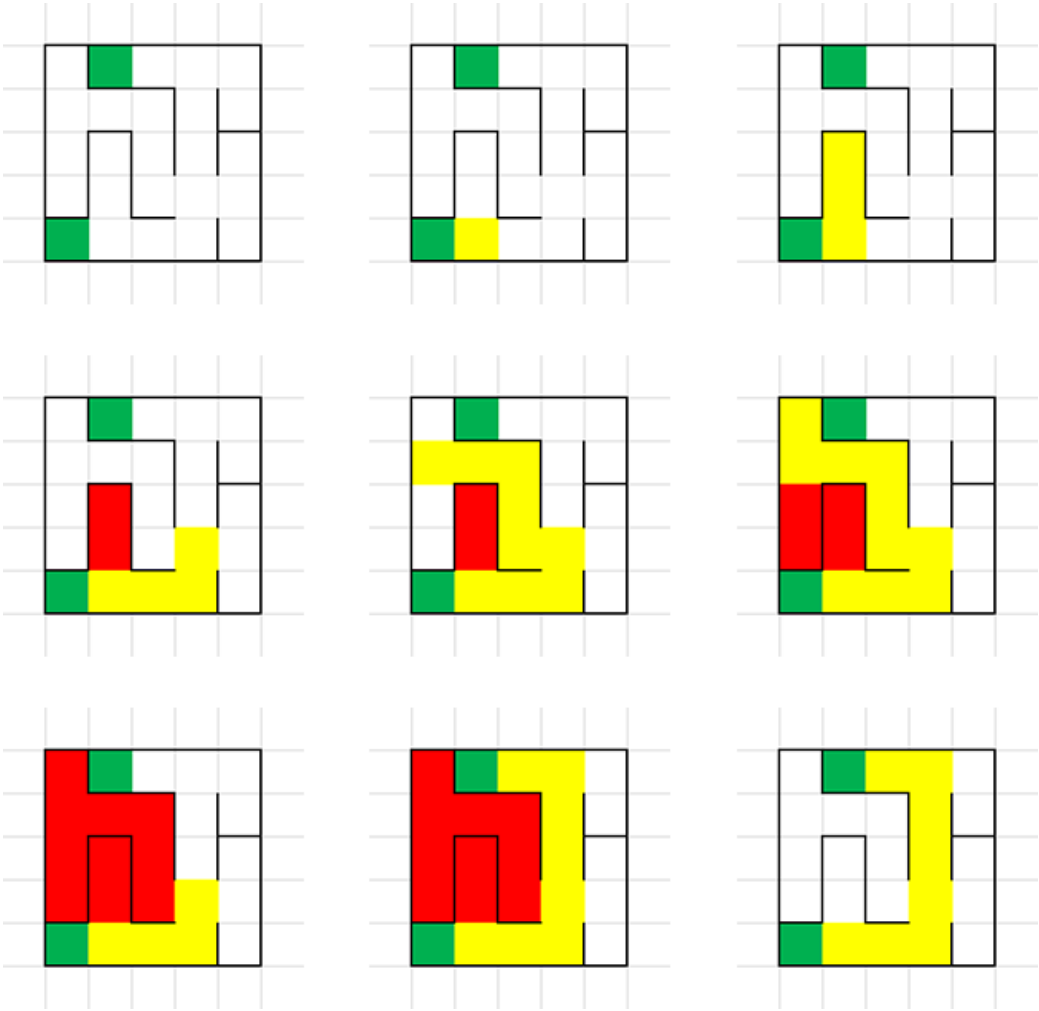
Logika je vrlo jednostavna. Ako je put neoznačen, možda vodi prema izlazu pa ima smisla krenuti tim putom dalje. Ako je put označen dvaput, znači da smo ga prošli u oba

smjera. Taj put je potpuno istražen i nije nas doveo do izlaza. Ako je put označen jednom, to bi značilo da na drugom kraju postoji križanje koje nismo do kraja istražili. Možda baš to križanje vodi prema izlazu pa ima smisla odabrati taj put.

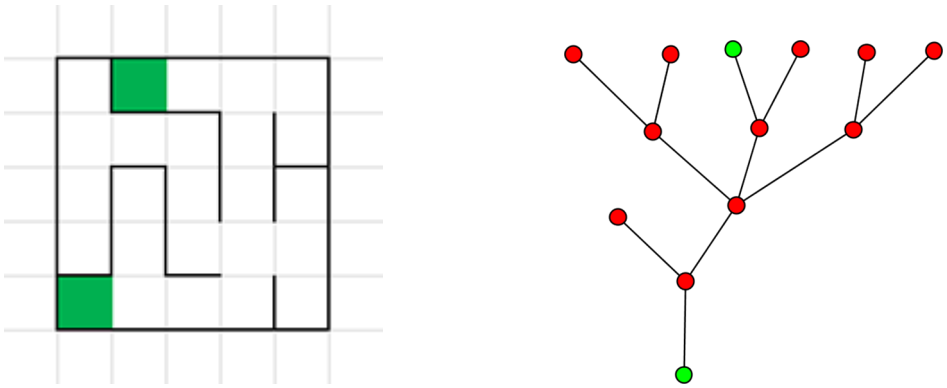
Vjerojatnije je da se izlaz nalazi na putu kojeg smo manje istražili pa zato dajemo prednost neistraženom putu. Ako imamo tri neoznačena puta, za nastavak kretanja proizvoljno odabiremo jedan od njih. Ako imamo dva neoznačena puta i jedan jednom označeni put, za nastavak kretanja proizvoljno biramo jedan od dva neoznačena puta.

Primjer kretanja po labirintu koristeći Trémauxov algoritam je ilustriran na slici 3.10. U drugom koraku smo došli do križanja i možemo nastaviti u dva smjera. Oba smjera su neoznačena pa nam je svejedno kojim ćemo nastaviti. Krenemo nekim smjerom te ga označimo žutom bojom jer nam je to prvi prolazak kroz taj hodnik. Na žalost, došli smo do slijepe ulice pa se vraćamo do prvog križanja označujući put crvenom bojom. To nam je drugi put da označavamo ovaj hodnik i više ga nikad nećemo koristiti. Nastavimo kretanje drugim hodnikom te dođemo do sljedećeg križanja. Nalazimo se u situaciji ilustriranom u četvrtom koraku na slici 3.10. Možemo nastaviti kroz tri hodnika. Sva tri su neoznačena pa proizvoljno odaberemo jedan on njih te ge istražimo, naravno označujući žutom i crvenom bojom. Svi putovi u odabranom hodniku su slijepi pa se vratimo nazad na križanje. Nalazimo se u situaciji ilustriranoj u sedmom koraku na slici 3.10. Na križanju možemo odabrati tri hodnika, ali jedan od njih je označen žutom bojom. To nam je znak da smo tamo već bili pa proizvoljno izaberemo jedan od dva hodnika u kojima nismo bili. Algoritam provodimo sve dok ne dođemo izlaza i tada žuto označeni hodnici nam predstavljaju put to ulaza.

Nakon konačno mnogo koraka, postoje dva slučaja. Ili smo izašli iz labirinta (hodnici koji su označeni točno jednom predstavljaju put od starta do cilja) ili smo svim hodnicima prošli dva puta i vratili smo se na početak. Odnosno, ne postoji izlaz iz labirinta. Labirint i reprezentacija pomoću grafa se nalaze na slici 3.11



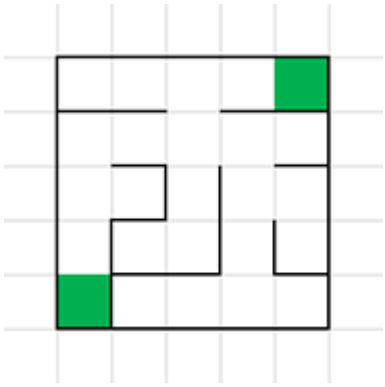
Slika 3.10: Koraci Trémauxovovog algoritma



(a) Labirint (b) Graf

Slika 3.11: Reprezentacija labirinta grafom

3.5 Algoritam „slijepih ulica" (*dead end filling*)



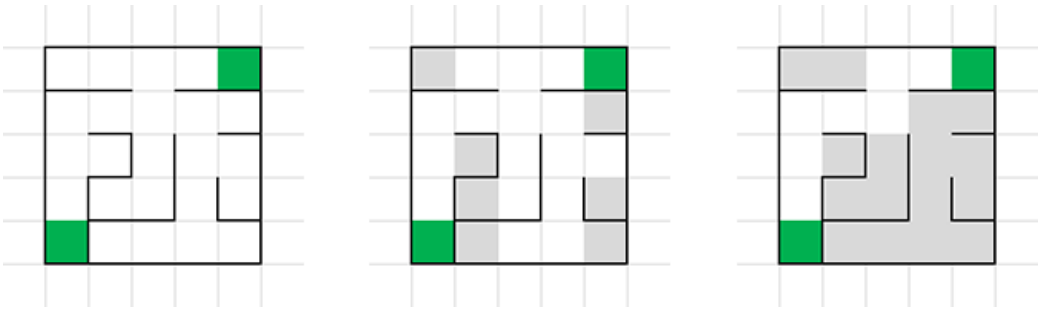
Slika 3.12: Predložak s ćelijama

Kao što mu i ime sugerira, ovaj algoritam se svodi na označavanju slijepih ulica, odnosno putova koje sigurno ne vode prema izlazu. Svim putovima koje nismo zacrnili možemo doći od ulaza do izlaza. Ako se fizički nalazimo u labirintu nemamo nikakve koristi od ovog algoritma, ali je vrlo prikladan za rješavanje labirinta na papiru.

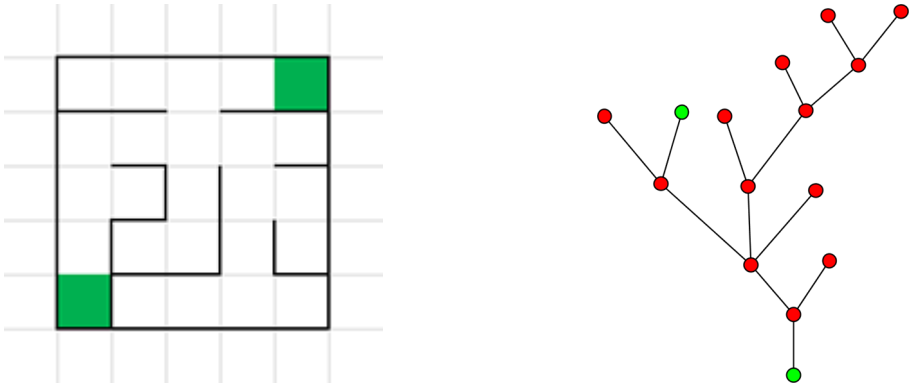
Algoritam ima samo dva koraka.

- Označi sve slijepe završetke.
- Zacrni sve hodnike krećući se od slijepih završetaka pa do prvog križanja.

Primjer rješavanja labirinta metodom slijepe ulice je prikazan na slici 3.13, a reprezentacija pomoću grafa se nalazi na slici 3.14b



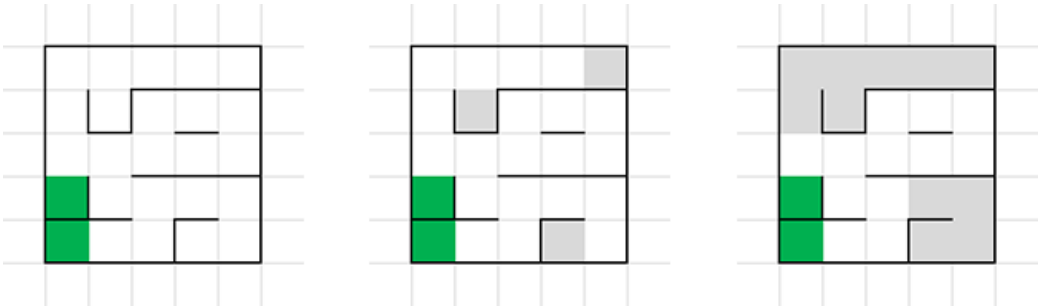
Slika 3.13: Koraci algoritma „slijepih ulica"



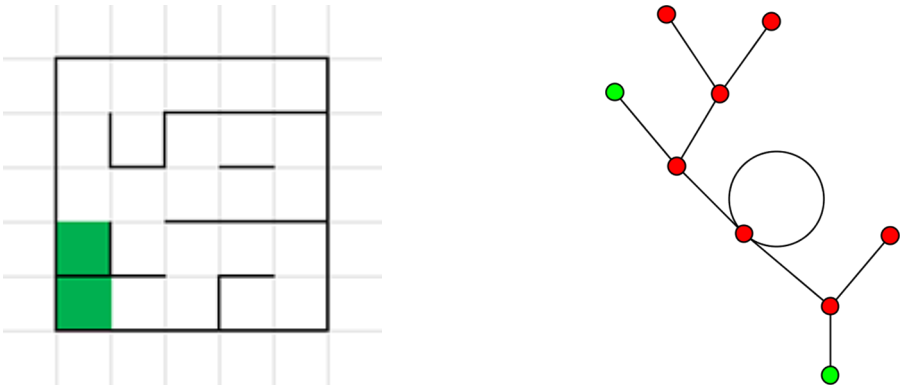
(a) Labirint (b) Graf

Slika 3.14: Reprezentacija labirinta grafom

Naravno, ukoliko nema slijepih završetaka ovaj algoritam je besmislen, a ako se u labirintu nalaze i petlje imamo višestruka rješenja. Primjer labirinta sa petljom je riješen na slici 3.15, a reprezentacija pomoću grafa se nalazi na slici 3.16b.



Slika 3.15: Koraci algoritma „slijepih ulica”



(a) Labirint (b) Graf

Slika 3.16: Reprezentacija labirinta grafom

Poglavlje 4

Labirinti u nastavi matematike

U ovom poglavlju su dani primjeri labirinata koje možemo koristiti u nastavi matematike i općenito u popularizaciji matematike. Većinu labirinata opisanih u ovom poglavlju je osmislio Steve Humble, poznatiji pod pseudonimom *Dr. Maths*. Također, dani su prijedlozi na koji način pojedine labirinte možemo prilagoditi uzrastu i intelektualnim sposobnostima učenika.

4.1 Primjer iz nastave

Sljedeća aktivnost je provedena u petom razredu osnovne škole u sklopu nastavne cjeline *prirodni brojevi* i nastavnoj jedinici je *ususatvljivanje gradiva*.

Aktivnost – Labirint

Cilj aktivnosti: Učenici će individualnim radom učinkovito koristiti računske operacije, uočavati veze između pojedinih računskih operacija te poštivati prednost računskih operacija. Učenici će zadatak s riječima uspješno analizirati, zapisati matematičkim simbolima te točno riješiti.

Nastavni oblik: Diferencirana nastava u obliku individualnog rada.

Nastavna metoda: Metoda rada s tekstom, problemska metoda.

Potreban materijal: Nastavni listić

Tijek aktivnosti: Učenici redom rješavaju zadatke, a odgovore „zacrnjuju“ u danoj tablici (riješena tablica se nalazi na slici 4.1) krećući se od starta prema cilju. Ukoliko dođu od starta do cilja, točno su riješili sve zadatke. Ukoliko ne mogu nastaviti „niz“, taj zadatak su krivo riješili.

3	1	4	1	5	9	2	6	5	3	**0**	**CILJ**
5	8	9	7	9	3	2	3	8	4	**8**	6
2	6	**0**	**1**	**6**	**9**	**0**	4	3	3	**0**	8
3	2	**0**	7	9	5	**3**	0	2	8	**1**	8
4	1	**7**	9	7	1	**8**	6	9	3	**0**	9
9	3	**1**	7	5	1	**9**	0	5	8	**7**	2
0	9	**2**	7	4	9	**6**	**3**	**2**	**3**	**9**	4
9	**2**	**3**	4	5	9	2	3	0	7	8	1
START	6	4	0	6	2	8	6	2	8	0	3

Slika 4.1: Rješenje labirinta

Zadatci:

1. Izračunaj na najbrži način:

   a) $747 + 129 + 21 + 26$

   b) $5 \cdot 217 \cdot 20 =$

   c) $126 - 13 + 4 \cdot 13$

   Rješenje:

   a) $747 + 129 + 21 + 26 = (129 + 21) + (747 + 26) = 150 + 773 = 923$

   b) $5 \cdot 217 \cdot 20 = 217 \cdot 5 \cdot 20 = 217 \cdot 100 = 21700$

   c) $126 - 13 + 4 \cdot 13 = 13(126 + 4) = 13 \cdot 130 = 1690$

2. Izračunaj razliku prethodnika broja 5712 i sljedbenika broja 1814.

   Rješenje:

   $$5711 - 1815 = 3896$$

3. Količniku brojeva 1504 i 32 dodaj umnožak broja 56 i njegovog sljedbenika.

   Rješenje:

   $$(1504 : 32) + (56 \cdot 57) = 47 + 3192 = 3239$$

4. Izračunaj: $60 + 1140 : (11 \cdot 11 - 7)$

   Rješenje:

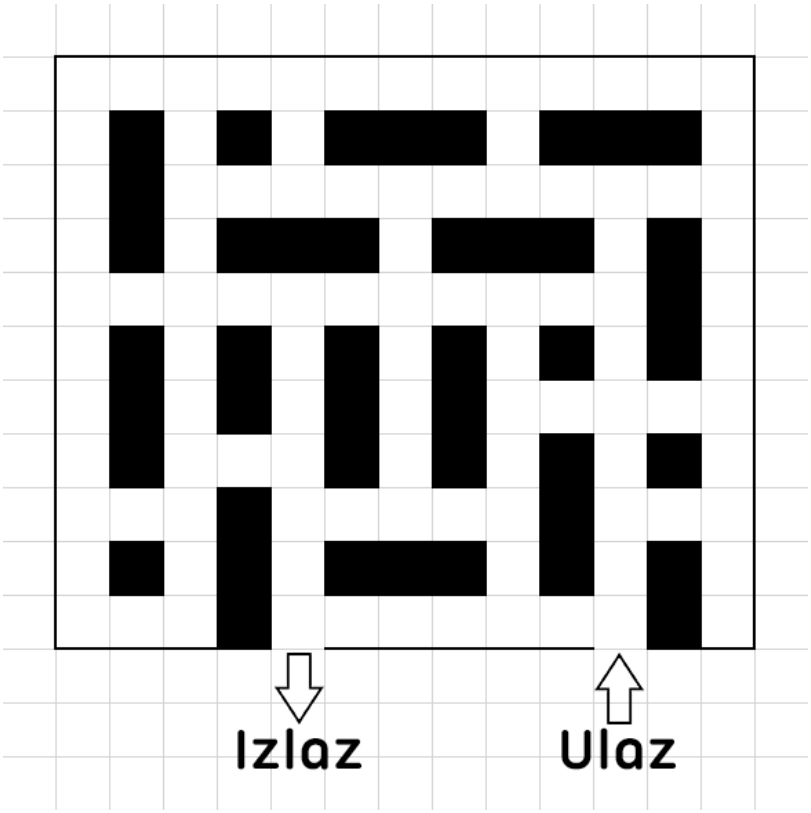
   $$60 + 1140 : (11 \cdot 11 - 7) = 60 + 1140 : (121 - 7) = 60 + 1140 : 114 = 60 + 10 = 70$$

5. U školskoj knjižnici ima tri puta više knjiga za lektiru od matematičkih priručnika, a matematičkih priručnika ima za sto i pet manje od ostalih knjiga. Koliko ima knjiga u knjižnici ako tristo knjiga ne pripadaju ni lektiri ni matematičkim priručnicima?

   Rješenje:

   - Ostale knjige: 300
   - Matematički priručnici: 300 – 105
   - Lektira: $3 \cdot (300 - 105)$

4.2 Zabranjeno lijevo (*no left maze*)

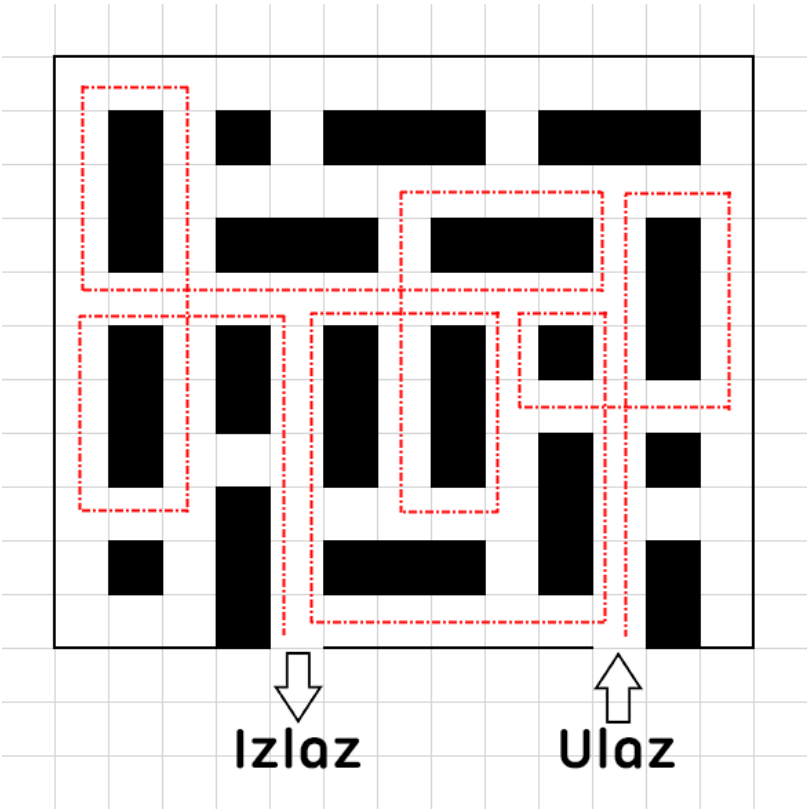


Slika 4.2: Primjer labirinta

Labirint prikazan na slici 4.2 ima samo jedno, naizgled trivijalno pravilo. Nije dozvoljeno skretanje u lijevo. Na križanju ili zadržimo smjer kretanja (produžimo ravno) ili skrenemo desno. Labirintom se možemo kretati u svim smjerovima (sjever, istok, zapad, jug) no primijetimo da se možemo okrenuti prema zapadu samo ako skrenemo tri puta u desno.

Ovakav zadatak je primjeren učenicima osnovnih i srednjih škola. Ovdje nema smisla primjenjivati algoritme jer na križanjima imamo uvjete za izbor hodnika kojima možemo nastaviti. Učenici će najčešće metodom slučajnog odabira izabrati jedan od dva ponuđena smjera kretanja (ravno ili desno) te olovkom ucrtati put od starta do cilja. Riješenje je prikazano na slici 4.3

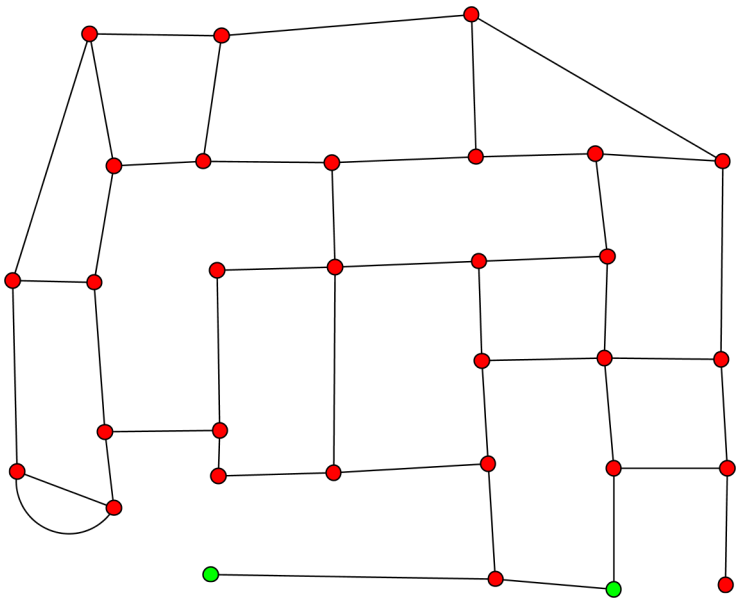
Neki učenici će možda primijetiti da je ekvivalentno krenuti od cilja prema startu, ali da u tom slučaju nije dozvoljeno skretanje u desno, te na taj način spojiti početnu i krajnju točku. Ti učenici imaju izraženiju moć zapažanja i logičkog zaključivanja ili su se već



Slika 4.3: Rješenje labirinta

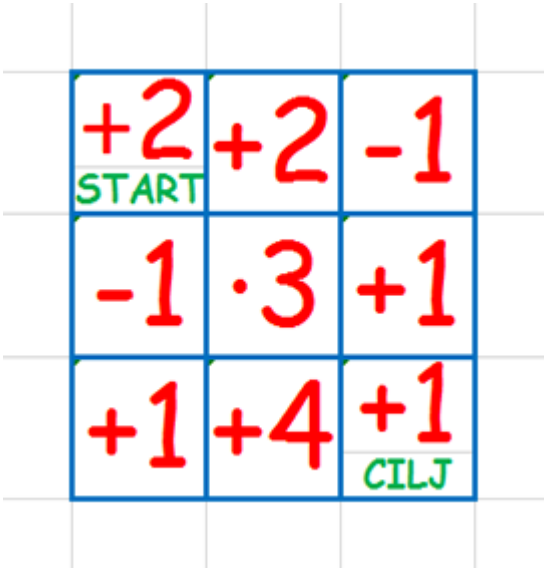
susreli sa enigmatskim zadatcima ovog tipa. Naime, često je lakše spojiti krajnju točku sa početnom nego obratno.

Reprezentacija labirinta grafom se nalazi na slici 4.4.



Slika 4.4: Reprezentacija labirinta grafom

4.3 Računski labirint



Slika 4.5: Primjer računskog labirinta

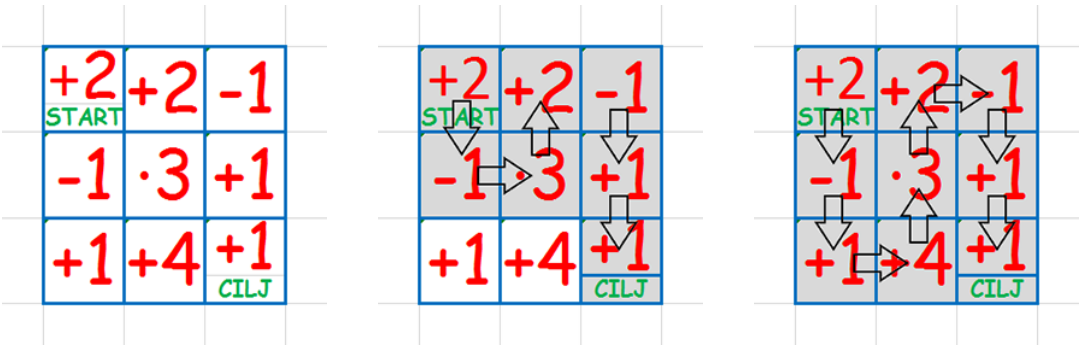
Ovakvi labirinti su primjereni učenicima nižih razreda osnovnih škola. Imamo rešetku 3×3 i u svakom kvadratiću je upisana računska operacija i broj. Svakim kretanjem izvršavamo zadanu računsku operaciju. Potrebno je naći put od starta do cilja poštujući pravila kretanja i na cilju imati traženi rezultat.

Pravila za kretanje su sljedeća:

- Dozvoljeno je kretanje u svim smjerovima, osim dijagonalnim;
- Svaki kvadratić smijemo posjetiti najviše jednom.

U primjeru sa slike 4.5 traženi krajnji rezultat je broj 6. Za dodatne bodove potrebno je pronaći put kojim dobivamo najveći rezultat. Sva rješenja su prikazana na slici 4.6

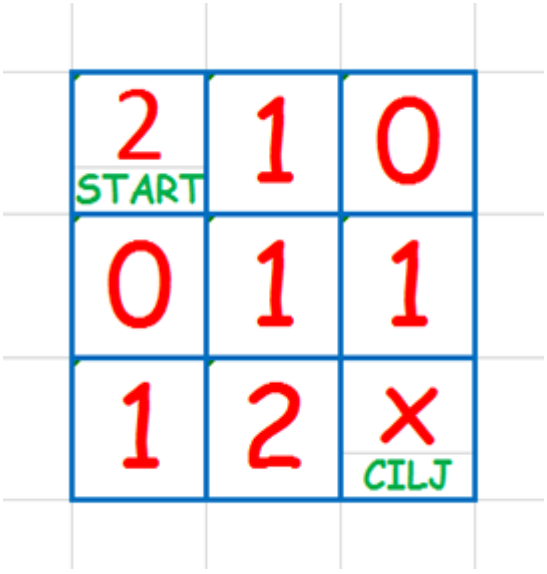
Nije preporučljivo proširivati rešetku na 4×4 ili više jer bi učenici imali previše kombinacija za odabrati traženi put. Potrošili bi previše vremena i ova aktivnost ne bi ispunila svoj cilj. Ukoliko želimo zadatak otežati, umjesto brojeva u polja možemo upisati neke složenije računske operacije. Na ovaj način zadatak možemo prilagoditi učenicima srednjih škola.



(a) Labirint (b) Rješenje (c) Dodatno rješenje

Slika 4.6: Zadatak i rješenja

4.4 Brojčani labirint



Slika 4.7: Primjer brojčanog labirinta

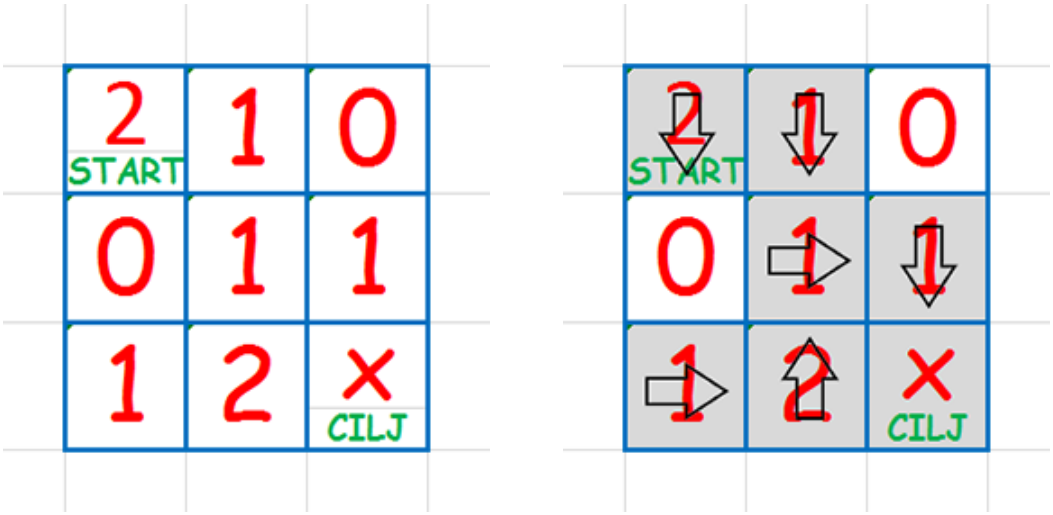
Ovaj labirint je primjeren učenicima nižih razreda osnove škole. Krećemo se u proizvoljnom smjeru za onoliko mjesta koliko piše na polju u kojem se trenutno nalazimo. Potrebno je naći put od starta do cilja, odnosno do polja sa oznakom $\times$. Pravila za kretanje su ista kao i kod prethodnog labirinta:

- Dozvoljeno je kretanje u svim smjerovima, osim dijagonalnim.
- Svaki kvadratić smijemo posjetiti najviše jednom.

Rješenje zadatka sa slike 4.7 je prikazano na slici 4.8. Zadatak sa slike 4.7 možemo prilagoditi višim razredima osnovnih škola tako da brojeve u poljima zamijenimo sa oznakama zadataka, primjerice A, B, C, D, E, F, G i H, kao što je prikazano na slici 4.9b.

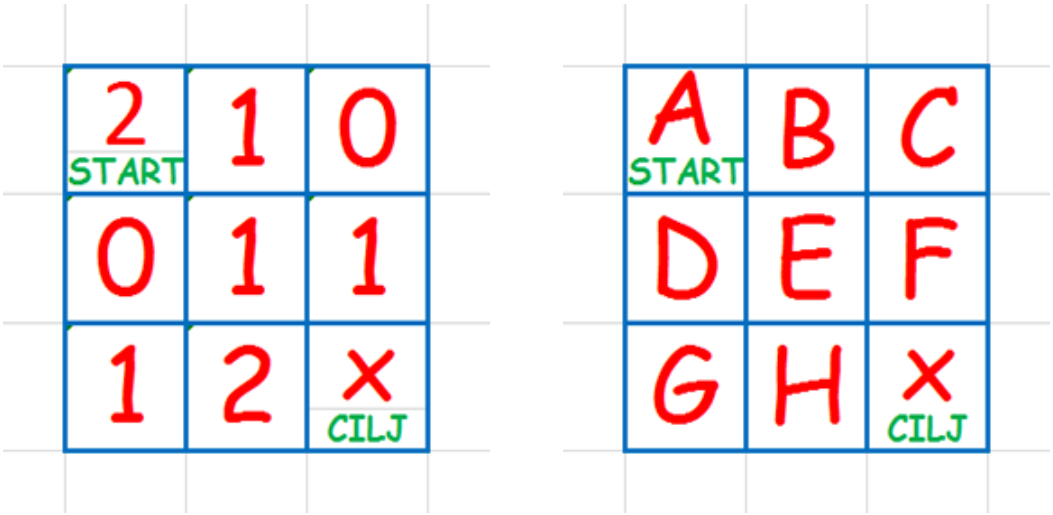
Svaki od sedam zadataka je zadatak jednostavnog višestrukog izbora sa ponuđena četiri odgovora, a redni broj točnog odgovora je broj koji nam označuje za koliko mjesta se trebamo pomaknuti. U ovom slučaju, točan odgovor zadatka A bi bio pod rednim brojem 2. Točan odgovor zadatka B bi bio pod rednim brojem 1, itd.

Ukoliko želimo proširiti rešetku na recimo 5×5, valja biti oprezan jer često možemo na više načina doći od starta do cilja. To nije nužno loše jer treba poticati učenike da na što



(a) Labirint (b) Rješenje

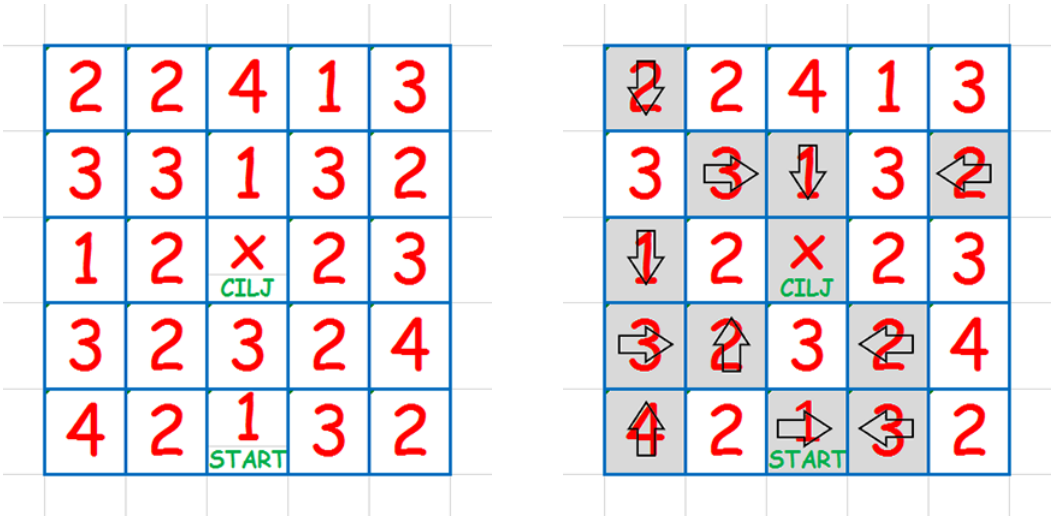
Slika 4.8: Labirint i rješenje



(a) Labirint (b) Poopćeni zadatak

Slika 4.9: Poopćavanje zadatka

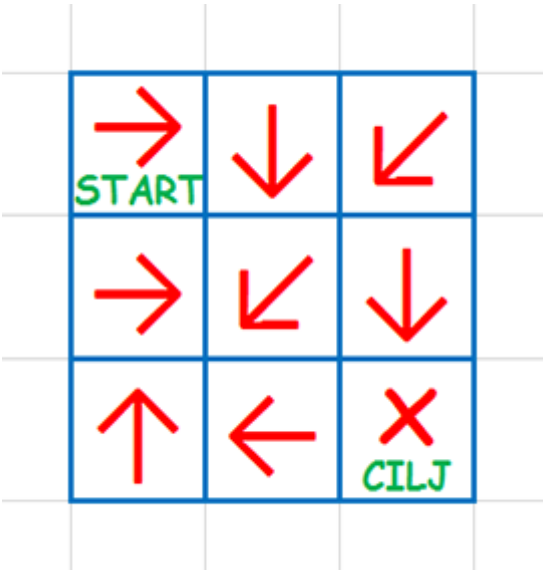
više načina dođu do konačnog rješenja, ali u svakom slučaju moramo biti oprezni. Primjer takvog zadatka, te jedno od dvadeset osam mogućih rješenja se nalazi na slici 4.10



(a) Labirint (b) Jedno rješenje

Slika 4.10: Prošireni zadatak

4.5 Labirint sa strelicama

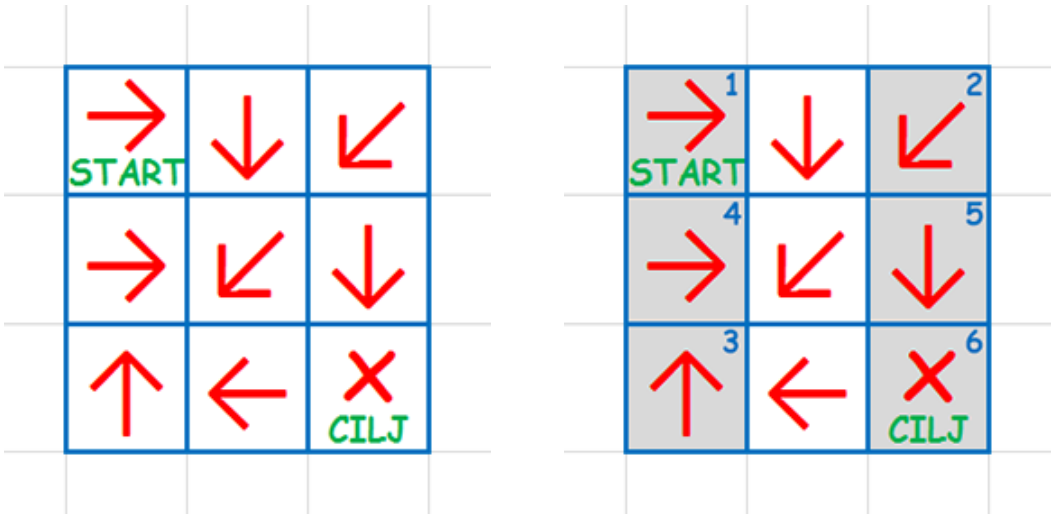


Slika 4.11: Primjer labirinta

Labirinti sa strelicama su primjereni učenicima nižih razreda osnovnih škola. U ovom labirintu se krećemo za proizvoljno mnogo koraka, ali nam je smjer kretanja određen sa strelicom koja se nalazi u polju na kojem se trenutno nalazimo. Potrebno je naći put od starta do cilja, odnosno polja označenom sa $\times$.

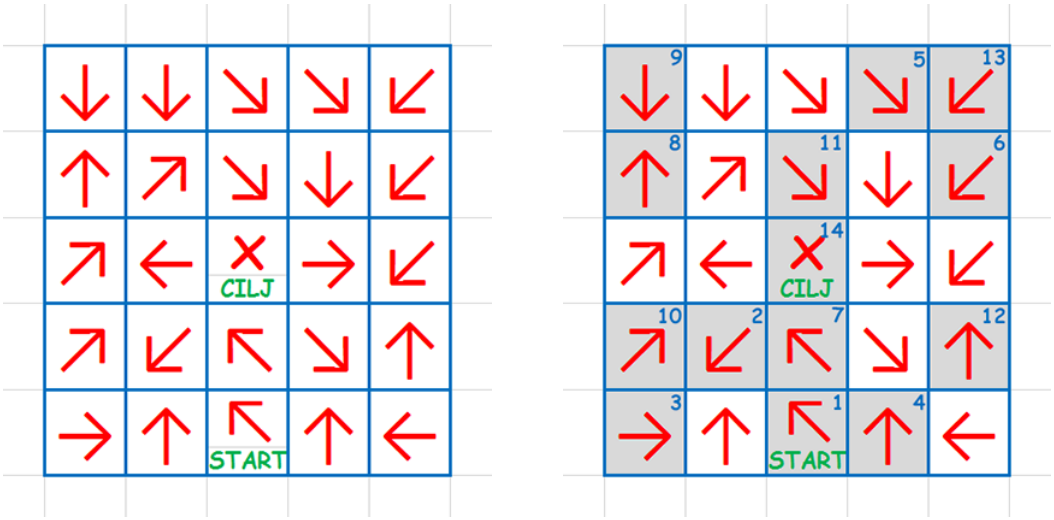
Rješenje zadatka sa slike 4.11 je prikazano na slici 4.12. Labirint sa strelicama možemo prilagoditi na isti način kao i brojčani labirint. Postupak je opisan na stranici 45, a vrijede ista upozorenja. Proširivanjem na veću rešetku često dobijemo višestruka rješenja.

Jedno proširenje sa jednim od rješenja je prikazano na slici 4.13.



(a) Labirint (b) Rješenje

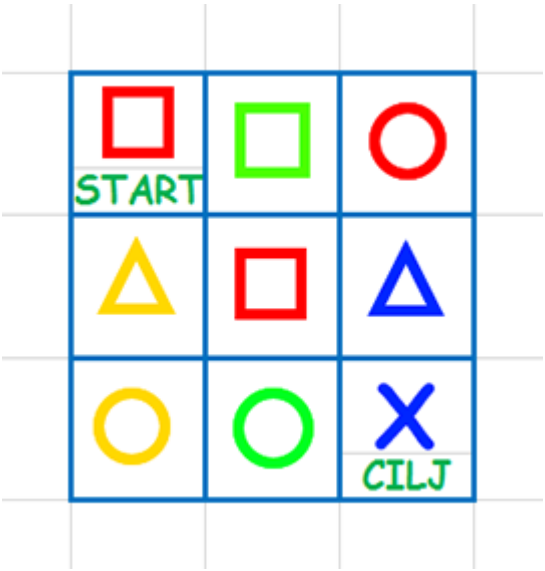
Slika 4.12: Labirint i rješenje



(a) Labirint (b) Rješenje

Slika 4.13: Labirint i rješenje

4.6 Labirint s geometrijskim likovima

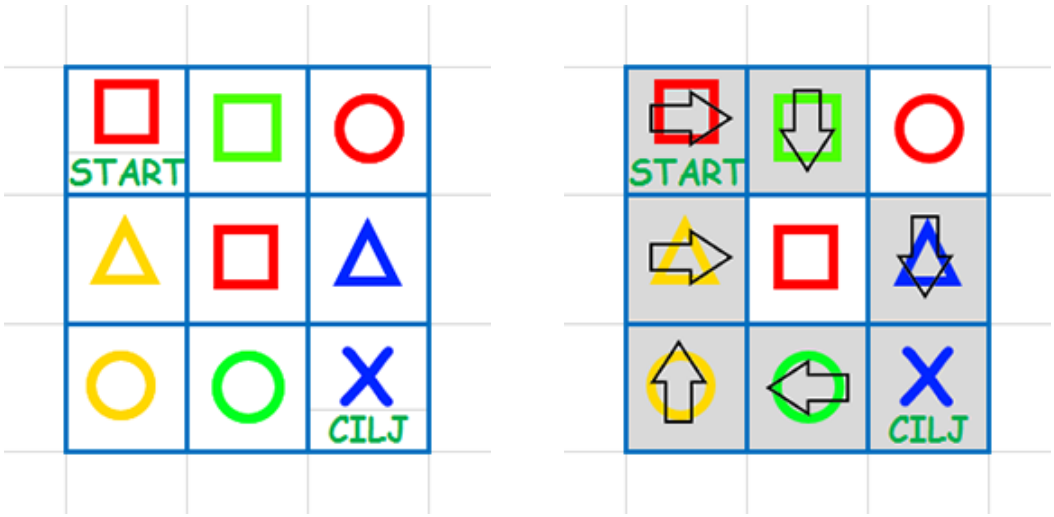


Slika 4.14: Primjer labirinta s geometrijskim likovima

Ovaj labirint je primjeren učenicima nižih razreda osnovnih škola, odnosno pri uvođenju geometrijskih likova. Potrebno je pronaći put od starta do cilja, a smijemo se kretati horizontalno ili vertikalno između dva polja na kojima se nalazi isti geometrijski lik ili lik iste boje. Dijagonalna kretanja nisu dozvoljena.

Rješenje labirinta sa slike 4.14 je prikazano na slici 4.15.

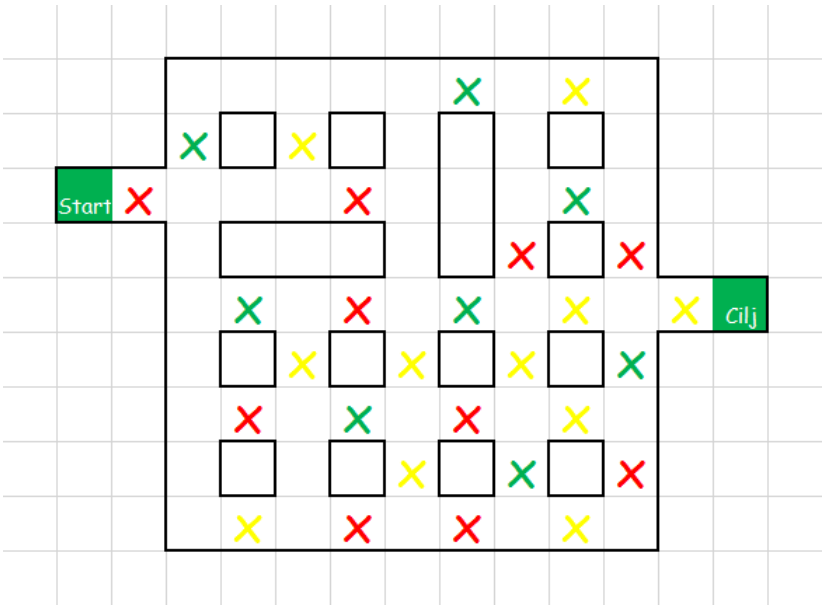
Analogne zadatke ovog tipa lako možemo napraviti proširivanjem rešetke. Višestruka rješenja koja time dobijemo su poželjna jer se ovakvi labirinti brzo rješavaju. Također, umjesto jedinstvene oznake za trokut možemo koristiti i pravokutan trokut, tupokutan trokut, raznostraničan trokut, ...



(a) Labirint (b) Rješenje

Slika 4.15: Labirint i rješenje

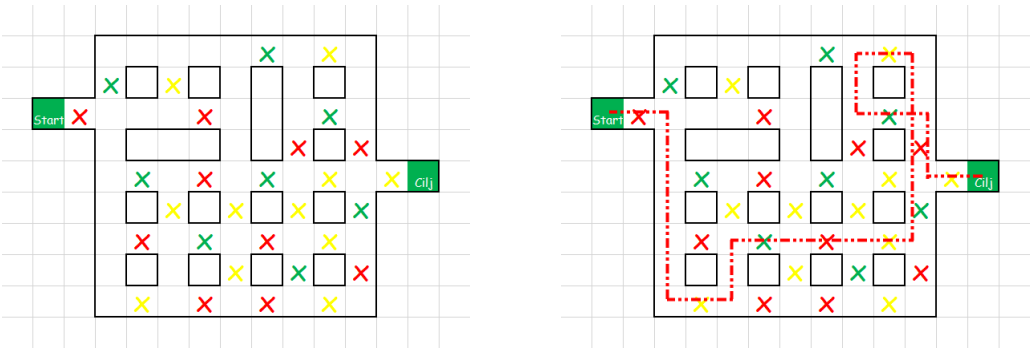
4.7 Alternirajući labirint



Slika 4.16: Primjer alternirajućeg labirinta

Kretanje po ovim labirintu je određen redoslijedom oznaka (odnosno njihovih boja) kojima se smijemo kretati. Nakon što prođemo crvenu oznaku, moramo nastaviti kretanje preko žute oznake, zatim zelene oznake, pa opet crvene. Redoslijed oznaka je dakle, crvena-žuta-zelena.

Rješenje labirinta sa slike 4.16 je prikazano na slici 4.17.

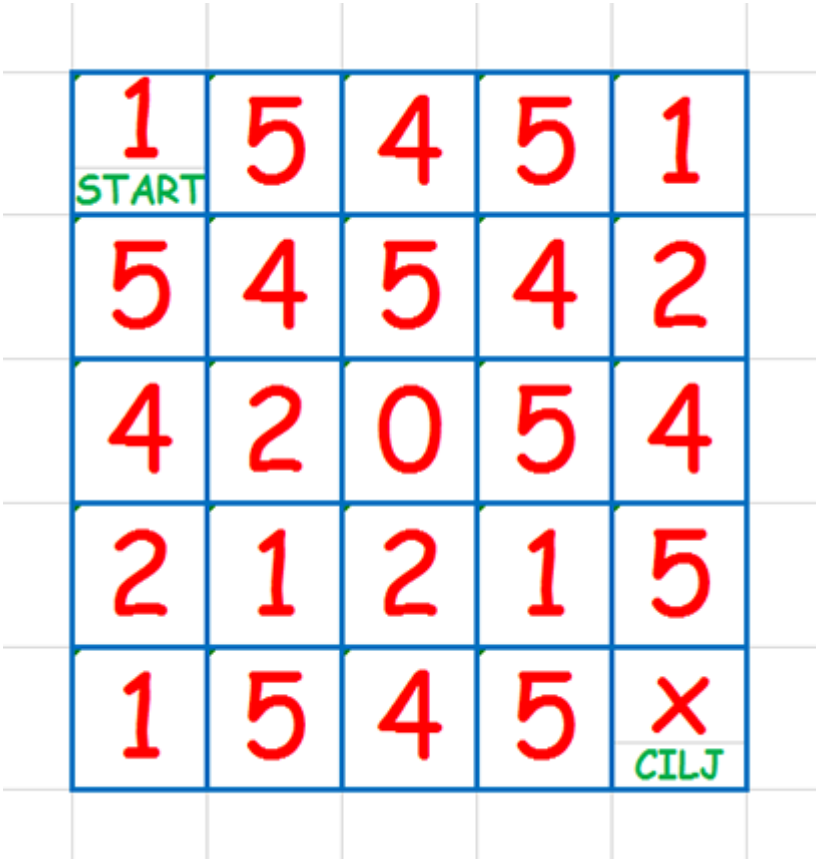


(a) Labirint (b) Rješenje

Slika 4.17: Labirint i rješenje

Ovaj labirint možda nije toliko prikladan u nastavi matematike, ali zasigurno pridonosi popularizaciji matematike. Može se iskoristiti na danima škole, a iznimno je efektan ukoliko se izradi u prirodnoj veličini, odnosno dovoljno velik da se učenici i roditelji mogu kretati kroz njega. Prepreke mogu predstavljati klupe i stolci, a oznake na podu se lako izrade od kolaž papira u boji.

4.8 Skakačev labirint

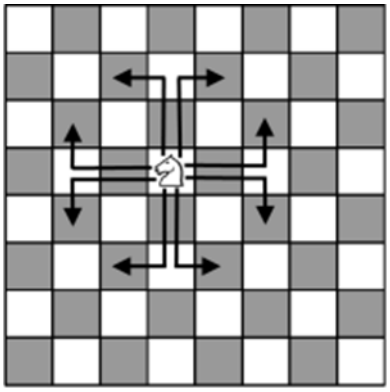


Slika 4.18: Primjer skakačevog labirinta

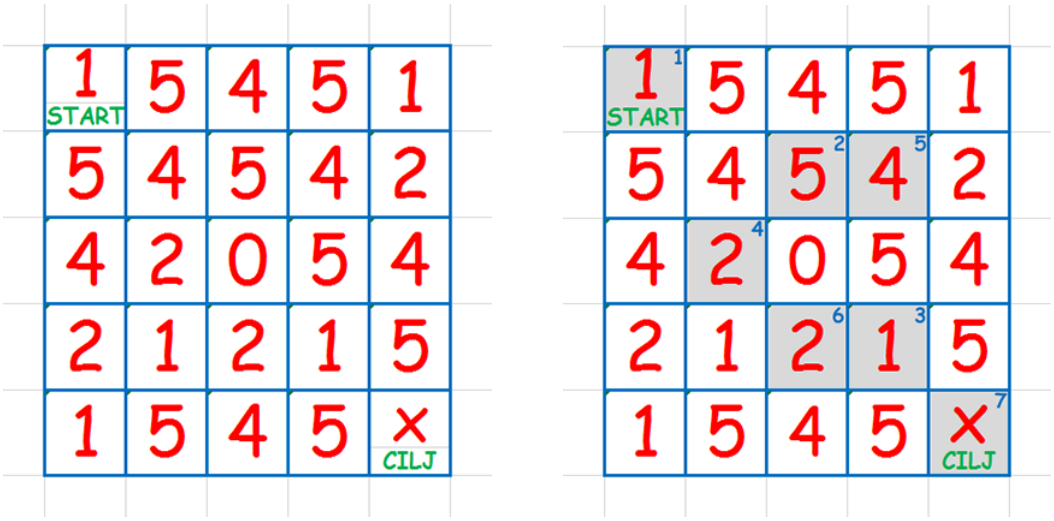
Dozvoljeno kretanje je određeno kretanjem skakača u šahu prikazanom na slici 4.19. Potrebno je doći od starta do cilja sumirajući brojeve na poljima kojima se krećemo. Dodatni uvjet je da suma na kraju mora biti 15.

Rješenje labirinta sa slike 4.18 je prikazano na slici 4.20.

Labirint je prikladan za osnovnoškolski uzrast, ali je potrebno učenicima objasniti na koji način se kreče skakač u šahu. Labirint sa skakačem možemo prilagoditi uzrastima i sposobnostima učenika na sličan način kao i brojčani labirint. Postupak je opisan na stranici 45. Ukoliko bi se labirint proširio na taj način, na rešetci 5×5 bi imali 24 polja sa zadatcima. Zbog vremenske učinkovitosti preporučljivo je ovu aktivnost izvesti u obliku rada u parovima, odnosno grupnom radu.



Slika 4.19: Kretanje skakača



(a) Labirint (b) Rješenje

Slika 4.20: Labirint i rješenje

Labirint sa skakačem je prikladan u popularizaciji matematike, odnosno na danima škole. Šahovsko polje se napravi u prirodnoj veličini, a učenik ili roditelj predstavljaju skakača koji se kreće od starta prema cilju.

Poglavlje 5

Zaključak

Iako se na prvi pogled nije činilo, labirinti imaju veliku i čvrstu matematičku podlogu. To nam je omogućilo da osmislimo brojne algoritme za rješavanje labirinata, ali i za generiranje novih, te na taj način labirinte povežemo sa još jednom važnim znanstvenim područjem, informatikom. Spregom labirinata, matematike i informatike dolazimo do idealne kombinacije za edukaciju. Matematički sadržaj na zanimljiv način možemo uvoditi tijekom cijelog osnovnoškolsko i srednjoškolskog obrazovanja. Mnogi labirinti opisani u knjizi R. Abbotta *SuperMazes : Mind Twisters for Puzzle Buffs, Game Nuts, and Other Smart People* se mogu iskoristiti u nastavi matematike i njenoj popularizaciji. Postoji iznimno mnogo algoritama koje se koriste u generiranju i rješavanju labirinta pomoću računala. Svaki od njih ima različitu metodu pristupa rješavanju problematike i ne postoji optimalan algoritam. Pregled dvanaest najučestalijih algoritama je obradio Jamis Buck u svojoj knjizi *Mazes for Programmers: Code Your Own Twisty Little Passages*.

Bibliografija

[1] C. J. Sangwin C. J. Budd, *Maths aMazes*, +plus magazine, http://plus.maths.org/content/maths-amazes/.

[2] H. Dudeney, *Amusements in Mathematics*, Dover Publications, 1959.

[3] M. Gardner, *Eleusis and the Soma Cube*, Mathematical Association of America, 2008.

[4] E. Lucas, *Recreations Mathematiques*, Paris **I** (1882.), 47–51.

[5] O. Ore, *An Excursion into Labyrinths*, The Mathematics Teacher **52** (1959.), 367–370.

[6] T. Phillips, *The Octosphericon and the Cretan Maze*, http://www.ams.org/samplings/feature-column/fcarc-octo-cretan.

[7] ———, *Through Mazes to Mathematics*, http://www.math.stonybrook.edu/~tony/mazes.

[8] D. Veljan, *Kombinatorna i diskretna matematika*, Algoritam, 2001.

Sažetak

Labirinti imaju dugu povijest, a uz logičko razmišljanje potrebno za njihovo osmišljavanje i rješavanje postoje mnogi drugi matematički aspekti. U ovom diplomskom radu dajemo pregled tih aspekata tako da bude jasan čitateljima raznih uzrasta i matematičkog predznanja.

U uvodnom dijelu dajemo kratak povijesni pregled, te uvodimo osnovne pojmove teorije grafova potrebne za razumijevanje algoritama navedenih u radu.

Glavni dio rada čine detaljno opisani algoritmi za generiranje i rješavanje labirinata. Algoritmi su prikazani pseudokodom pa čitatelj može provesti korake na papiru i na taj način demonstrirati generiranje ili rješavanje labirinta. Ukoliko imamo dovoljno računalnog predznanja, pseudokodove možemo implementirati u neki od programskih jezika te na taj način generirati ili rješavati labirinte pomoću računala.

Rad je dopunjen primjerima upotrebe teme u popularizaciji matematike. Primjeri se mogu iskoristiti i u nastavi matematike, a jedan od primjera je osmišljen i proveden u sklopu kolegija Metodička praksa iz matematike u osnovnoj školi.

Summary

Mazes have a long history and there are a lot of other mathematical aspects beside the logical reasoning needed for generating and solving them. In this thesis a clear and comprehensive overview of these aspects is presented, understandable to a wide range of readers of various ages and mathematical backgrounds.

A brief history of mazes is presented in the opening part, as are basic notions of graph theory needed for understanding of presented algorithms.

The main part of the thesis are maze generation algorithms and maze solving algorithms, which are described in detail. Algorithms are described in pseudocode and the reader can follow the steps to demonstrate generating or solving a maze with a pen and paper. Readers with a sufficient level of computer knowledge could implement the pseudocodes in a computer language of their choice and use them to generate or solve mazes.

The thesis is supplemented with examples of how to use mazes in popularization of mathematics. The examples can be used in teaching as well, and one of the examples was particularly designed and conducted in the course Methodical practice of mathematics in elementary school.