

Depth-First search VS Breadth-First search

Sárközi Viktor, Oravecz Zsolt

2020. 05. 07

Tartalom

- 1 Breadth-first search
- 2 Depth-first search
- 3 Eredmények
- 4 Megvalósítás

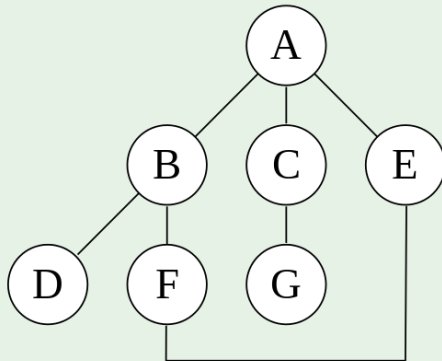
Breadth-first search

BFS

- A szélességi keresés (breadth-first search) egy olyan fabejárési algoritmus, amely a fában a gyökértől kiindulva minden csúcst bejárja.
- Az algoritmus lényege, hogy a fában igyekszik először a gyökérhez közelebbi csúcsokat átvizsgálni, és csak azután folytatja a keresést a mélyebb csúcsokon.
- Általában Sor (Queue) adatszerkezetet használnak a megvalósításához.

Breadth-first search

Működése



Útvonal: A, B, C, E, D, F, G.

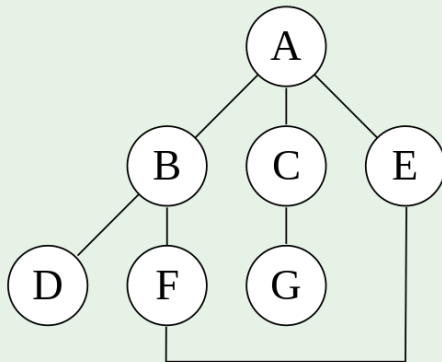
Depth-first search

DFS

- A mélységi keresés (depth-first search) egy olyan fabejárési algoritmus, amely a fában a gyökértől kiindulva minden csúcsot bejárja.
- Az algoritmus lényege, hogy a fában igyekszik először a gyökértől távolabbi (mélyebb) csúcsokat átvizsgálni, és csak azután lép vissza a szomszédos és a korábbi csúcsokhoz.
- Általában Verem (Stack) adatszerkezetet használnak a megvalósításához.

Depth-first search

Működése



Útvonal: A, B, D, F, E, C, G.

Eredmények

Főbb szempontok

- Végrehajtási idő mérése (Run time)
- Megoldás lépéseinek száma (Solving steps)
- Legrövidebb út megtalálása (Optional way)

Különböző labirintusok

- 50x50 labirintus (50x50 labyrinth)
- Saját labirintus (Own labyrinth)
- Random generált labirintus (Random labyrinth)

Eredmények

	Solving with Breadth-First-Search		
	<i>Solving steps</i>	<i>Optional way</i>	<i>Run time</i>
50x50 labyrinth	156	guaranteed	0:00:00.0016294
Own labyrinth	109	guaranteed	0:00:00.0013517
Random labyrinth	103	guaranteed	0:00:00.0012555

	Solving with Depth-First-Search		
	<i>Solving steps</i>	<i>Optional way</i>	<i>Run time</i>
50x50 labyrinth	194	not guaranteed	0:00:00.0015694
Own labyrinth	129	not guaranteed	0:00:00.009157
Random labyrinth	121	not guaranteed	0:00:00.0008352

Eredmények

Főbb szempontok

- Megoldás lépéseinek száma (Solving steps)
- Rekurzív hívások száma (Recursive call)
- Sor műveletek száma (Queue operation)

Különböző labirintusok

- 5 Random generált labirintus (Random labyrinth)

Eredmények

	Random labyrinth solving with BFS		
	<i>Solving steps</i>	<i>Recursive call</i>	<i>Queue operation</i>
maze1	103	not use	181
maze2	95	not use	219
maze3	79	not use	221
maze4	99	not use	149
maze5	87	not use	166
Average	92,6	-	187,2

	Random labyrinth solving with DFS		
	<i>Solving steps</i>	<i>Recursive call</i>	<i>Queue operation</i>
maze1	121	233	not use
maze2	137	177	not use
maze3	149	215	not use
maze4	117	129	not use
maze5	87	228	not use
Average	122,2	196,4	-

Megvalósítás

```
C:\Users\O.Zsolt\Desktop\Labirintus\LabirintusTeszt\LabirintusTeszt\bin\Debug\Labiri
```

```
Do you want to generate a maze(Press G) or do you enter one?(Press E)
```

```
Do you want to generate a maze(  
G  
Enter Start x: (4)  
4  
Enter Start y: (0)  
0  
Enter destination x: (33)  
33  
Enter destination y: (14)  
14
```

```
C:\Users\O.Zsolt\Desktop\Labirintus\LabirintusTeszt\Labi
```

```
Do you want to generate a maze(Press G) or  
E  
Please tyep a labirint: (for example: labi  
labirintus.txt
```

Generálás és manuális labirintus megadása.

Megvalósítás

```

StreamReader sr = new StreamReader(filename);
int counter = 0;
while ((line = sr.ReadLine()) != null)
{
    if (counter == 0)
    {
        for (int i = 0; i < line.Length; i++)
        {
            Read.Add(Convert.ToString(line[i]));
        }
        counter++;
    }
    else
    {
        for (int i = 0; i < line.Length; i++)
        {
            Read[i] = Read[i] + Convert.ToString(line[i]);
        }
    }
}

for (int y = 1; y < Read.Count - 2; y++)
{
    int a = Read[y].Length;
    for (int x = 0; x < Read[y].Length; x++)
    {
        string c = Convert.ToString(Read[y][x]);
        Palya2[y - 1, x] = Convert.ToString(Read[y][x]);
        if (Palya2[y - 1, x] == "S")
        {
            Program.start = new Point(y - 1, x);
            Palya2[y - 1, x] = " ";
        }
        if (Palya2[y - 1, x] == "C")
        {
            Program.destination = new Point(y - 1, x);
            Palya2[y - 1, x] = " ";
        }
        //string b = Palya2[y, x - 1];
    }
}

```

Labirintus beolvasása

Megvalósítás

```
/* ha meg nem találtuk meg a Kijaratot... ES ha tudunk jobbra menni...
if (!megtalalt && x < MERETX - 1 && Palya[y, x + 1] == Jarat)
{
    /* ha arra van a megfejtes */
    if (megfejt(Palya, x + 1, y, celx, cely)) megtalalt = true;
}

/* balra */
if (!megtalalt && x > 0 && Palya[y, x - 1] == Jarat)
{
    if (megfejt(Palya, x - 1, y, celx, cely)) megtalalt = true;
}

if (!megtalalt && y > 0 && Palya[y - 1, x] == Jarat)
{
    if (megfejt(Palya, x, y - 1, celx, cely)) megtalalt = true;
}
if (!megtalalt && y < MERETY - 1 && Palya[y + 1, x] == Jarat)
{
    if (megfejt(Palya, x, y + 1, celx, cely)) megtalalt = true;
}
```

Mélységi bejárás program részlet.

Megvalósítás

```
while (q.Count != 0)
{
    curr = q.Peek();
    Point pt = curr.pt;

    // Ha megtaláltuk a célt, akkor készen vagyunk
    if (pt.x == dest.x && pt.y == dest.y)
        return curr.dist;

    // Egyébként kivesszük a sorból az aktuális elemet
    q.Dequeue();

    for (int i = 0; i < 4; i++)
    {
        int row = pt.x + rowNum[i];
        int col = pt.y + colNum[i];

        // Ha a következő elem, érvényes és még nem vizsgált,
        //akkor beletesszük a sorba
        if (isValid(row, col) &&
            mat[row, col] == 1 &&
            !visited[row, col])
        {
            //az elemet vizsgált kulcsszóval jelöljük
            visited[row, col] = true;
            queueNode Adjcell = new queueNode(new Point(row, col),
                                                curr.dist + 1, curr);
```

Szélességi bejárás program részlet.

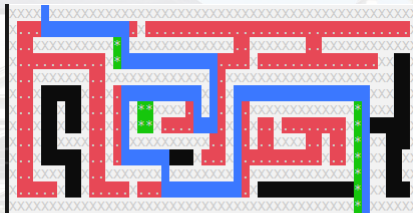
Megvalósítás



<Press Enter!>

DFS-Path is 93 steps.
 BFS-Optimal Path is 51 steps.
 Recursions number: 143
 Queue using: 153

- Dead End
- Path
- Optimal Path



<Press Enter!>

Path is 94 steps.
 Optimal Path is 90 steps.
 Resursions number: 262
 Queue using: 288

- Dead End
- Path
- Optimal Path

Generált és manuálisan megadott labirintus.



Köszönjük a figyelmet!