

Magic Grapher

Hledání magicky ohodnocených grafů počítačem

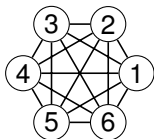
Augustin Žídek

augustin@zidek.eu

17. dubna 2012

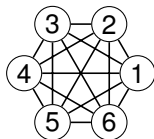
Motivace: Neúplné sportovní turnaje

- ▶ Úplný turnaj: každý hraje s každým

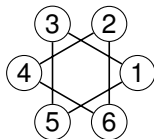


Motivace: Neúplné sportovní turnaje

- ▶ Úplný turnaj: každý hraje s každým



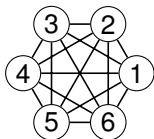
- ▶ Jak na spravedlivý turnaj, kde nehraje každý s každým



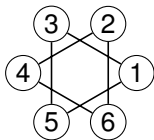
Nespravedlivý neúplný turnaj

Motivace: Neúplné sportovní turnaje

- ▶ Úplný turnaj: každý hraje s každým



- ▶ Jak na spravedlivý turnaj, kde nehraje každý s každým



Nespravedlivý neúplný turnaj

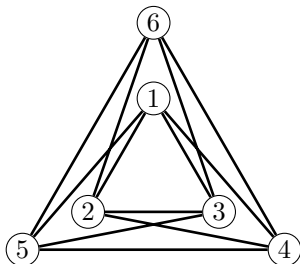
- ▶ Spravedlivý neúplný turnaj: Pravidelný neúplný graf, magicky ohodnocený

Magické ohodnocení grafů

- ▶ Součet vah sousedních vrcholů je stejný pro každý vrchol

Magické ohodnocení grafů

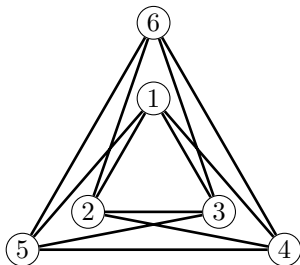
- ▶ Součet vah sousedních vrcholů je stejný pro každý vrchol
- ▶ Například pro $n = 6, r = 4$



4-pravidelný graf na 6 vrcholech

Magické ohodnocení grafů

- ▶ Součet vah sousedních vrcholů je stejný pro každý vrchol
- ▶ Například pro $n = 6, r = 4$



4-pravidelný graf na 6 vrcholech

- ▶ Magická konstanta m_k (váha každého vrcholu):

$$w(i) = m_k = \frac{(n+1)r}{2}$$

Jak se magické grafy hledají

Jak se magické grafy hledají

- ▶ Těžko

Jak se magické grafy hledají

- ▶ Těžko
- ▶ Vysoká algoritmická časová složitost – přibližně faktoriální

Jak se magické grafy hledají

- ▶ Těžko
- ▶ Vysoká algoritmická časová složitost – přibližně faktoriální
- ▶ Pro malá $n < 30$ to ještě jde

Jak se magické grafy hledají

- ▶ Těžko
- ▶ Vysoká algoritmická časová složitost – přibližně faktoriální
- ▶ Pro malá $n < 30$ to ještě jde
- ▶ $25! = 15511210043330985984000000 \doteq 1,5 \cdot 10^{25}$
 $30! \doteq 2,6 \cdot 10^{32}$
 $50! \doteq 3,0 \cdot 10^{64}$
- ▶ Počet částic na Zemi asi 10^{50}

Algoritmus

- ▶ `PermutationLister` nalezne všechny kombinace čísel se stejným součtem, vrátí je již omezené pro každý vrchol

Algoritmus

- ▶ `PermutationLister` nalezne všechny kombinace čísel se stejným součtem, vrátí je již omezené pro každý vrchol
- ▶ `MagicGraphTester` zkouší rekurzivně tyto možnosti pro jednotlivé vrcholy

Algoritmus

- ▶ `PermutationLister` nalezne všechny kombinace čísel se stejným součtem, vrátí je již omezené pro každý vrchol
- ▶ `MagicGraphTester` zkouší rekurzivně tyto možnosti pro jednotlivé vrcholy
- ▶ `MGTesterThread` umožňuje paralelizmus

Algoritmus

- ▶ `PermutationLister` nalezne všechny kombinace čísel se stejným součtem, vrátí je již omezené pro každý vrchol
- ▶ `MagicGraphTester` zkouší rekurzivně tyto možnosti pro jednotlivé vrcholy
- ▶ `MGTTesterThread` umožňuje paralelizmus
- ▶ `GraphTeXExport` export do \LaTeX u

Optimalizace

- ▶ Více vláken

Optimalizace

- ▶ Více vláken
- ▶ 6-pravidelný na 13 vrcholech za **35 s**, nyní **550 ms**

Optimalizace

- ▶ Více vláken
- ▶ 6-pravidelný na 13 vrcholech za **35 s**, nyní **550 ms**
- ▶ Pravděpodobnostní optimalizace
`if ((digits[i] < digitMax) && (nextDigiti != digits[i + 1]))`

Optimalizace

- ▶ Více vláken
- ▶ 6-pravidelný na 13 vrcholech za **35 s**, nyní **550 ms**
- ▶ Pravděpodobnostní optimalizace
`if ((digits[i] < digitMax) && (nextDigit[i] != digits[i + 1]))`
- ▶ Zvyšování paměťové náročnosti, snižování výpočetní

Optimalizace

- ▶ Více vláken
- ▶ 6-pravidelný na 13 vrcholech za **35 s**, nyní **550 ms**
- ▶ Pravděpodobnostní optimalizace
`if ((digits[i] < digitMax) && (nextDigit[i] != digits[i + 1]))`
- ▶ Zvyšování paměťové náročnosti, snižování výpočetní
- ▶ Booleovská pole místo polí intů $O(\log n) \rightarrow O(1)$

[1,2,4,6]	→	0	1	2	3	4	5	6
		false	true	true	false	true	false	true

Ukázka výhod booleovských polí

```
public List<boolean[]> getPermsForNode(final List<boolean[]> perms,
    final int[] inclNodes, final int[] exclNodes) {
    final List<boolean[]> correctPermutations = new ArrayList<>();

    permsLoop: for (final boolean[] i : perms) {
        for (final int included : inclNodes) {
            if (!i[included]) {
                continue permsLoop;
            }
        }

        for (final int excluded : exclNodes) {
            if (i[excluded]) {
                continue permsLoop;
            }
        }
        correctPermutations.add(i);
    }
    return correctPermutations;
}
```

GUI + ukázka

Nodes:

Edges:

Use antimagic

Symmetric graphs only

Show all solutions

Exit on first solution

Convert to LaTeX

Use multithread

Generate!

Terminate

Solutions: k:

Done! Took 703 ms.

Textual form:

```
2: 3, 4, 6, 7, 10, 12,
3: 2, 5, 7, 8, 9, 11,
4: 1, 2, 5, 9, 12, 13,
5: 1, 3, 4, 10, 11, 13,
6: 1, 2, 7, 8, 11, 13,
7: 2, 3, 6, 8, 11, 12,
8: 1, 3, 6, 7, 12, 13,
9: 1, 3, 4, 10, 11, 13,
10: 1, 2, 5, 9, 12, 13,
11: 3, 5, 6, 7, 9, 12,
12: 2, 4, 7, 8, 10, 11,
13: 4, 5, 6, 8, 9, 10,

1: 4, 5, 6, 8, 9, 10,
2: 3, 5, 6, 7, 9, 12,
3: 2, 4, 7, 8, 10, 11,
4: 1, 3, 5, 9, 11, 13,
5: 1, 2, 4, 10, 12, 13,
6: 1, 2, 7, 8, 11, 13,
7: 2, 3, 6, 8, 11, 12,
8: 1, 3, 6, 7, 12, 13,
9: 1, 2, 4, 10, 12, 13,
10: 1, 3, 5, 9, 11, 13,
11: 3, 4, 6, 7, 10, 12,
12: 2, 5, 7, 8, 9, 11,
13: 4, 5, 6, 8, 9, 10,
```

Instructions:
Amount of nodes must be greater than amount of edges.

When converting to LaTeX, LaTeX instalation must be present on the system. The application will create file called "nodes_edges.tex" in your home folder and LaTeX will process it.

The application will use as many threads as many processors available.

Augustin Židek, 2012

Hlavní okno programu

Diskuse, otázky, optimalizace?

`augustin@zidek.eu`

Děkuji za pozornost