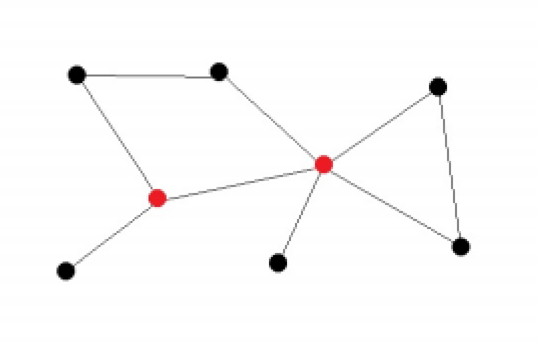
Решавање проблема минималног доминирајућег скупа

Немања Живановић 89/2016

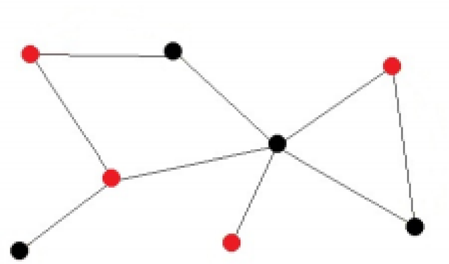
1. Опис проблема

За дати граф G = (V,E) одредити подскуп чворова V' ⊆ V тако да за свако u ∈ V – V' постоји v ∈ V' тако да (u,v) ∈ E. За скуп чворова за који ово важи се каже да је доминирајући скуп. Циљ је одредити најмањи такав подскуп V' односно најмањи доминирајући скуп.

Другим речима, треба одредити скуп чворова V' ⊆ V тако да сви чворови који остану ван тог скупа буду повезани граном бар са једним чвором који јесте у скупу и да тај скуп буде најмање могуће кардиналности.


[1]

Црвени чворови чине минимални доминирајући скуп



Црвени чворови чине доминирајући скуп, али није минималан

[1] http://www.sofsem.cz/sofsem12/files/presentations/Monday/FCSA/Couturier.pdf слајдови 5. и 6.

Authors	Polynomial space	Exponential space
Fomin et al.	$\mathcal{O}(1.9379^n)$	
Schiermeyer	$\mathcal{O}(1.8899^n)$	
Grandoni	$\mathcal{O}(1.9053^n)$	$\mathcal{O}(1.8021^n)$
Fomin et al.	$\mathcal{O}(1.5263^n)$	$\mathcal{O}(1.5137^n)$
van Rooij	$\mathcal{O}(1.5134^n)$	$\mathcal{O}(1.5086^n)$
van Rooij and Bodlaender	$\mathcal{O}(1.5134^n)$	$\mathcal{O}(1.5063^n)$
van Rooij et al.		$\mathcal{O}(1.5048^n)$

Најбржи познати алгоритми за решавање овог проблема

Један од таквих је и [1]. У њему се проблем минималног доминирајућег посматра као специјални случај проблема минималног покривача скупа.

Проблем минималног покривача скупа подразумева постојање скупа елемената {1,2,…,n} који се назива универзум, и скупа S од m подскупова датог универзума чија је унија једнака универзуму. Треба одредити минимални подскуп S’ ⊆ **S** чија је унија такође једнака универзуму и тај скуп S’ се назива минимални покривач.

Универзум ће бити чворови графа, а елементи **S** ће бити скупови N(v) ∪ {v} за сваки чвор v ∈ V, где N(v) садржи све суседе чвора v.

```
int MSC(𝒮) {
    if(|𝒮| = 0) return 0;
    if(∃S, R ∈ 𝒮: S ⊂ R) return MSC(𝒮\{S});
    if(∃s ∈ 𝒰_𝒮 ∃ a unique S ∈ 𝒮: s ∈ S) return 1 + MSC(del(𝒮, S));
    take S ∈ 𝒮 of maximum cardinality;
    return min{MSC(𝒮\{S}), 1 + MSC(del(𝒮, S))};
}
```

где је:

$$del(\mathscr{S}, R) = \{S' \neq \emptyset :\ S' = S \backslash R,\ S \in \mathscr{S}\}.$$

Односно овим се означава колекција добијена из **S** брисањем елемената из неког подскупа R ⊆ U из сваког s ∈ **S** и каснијим брисањем празног скупа.

Алгоритам се заснива на следећој леми:

(1) *If there is a set S in $\mathscr{S}$ which is (properly) included in another set R in $\mathscr{S}$ ($S \subset R$), then there is a minimum set cover which does not contain S. In particular:*

$$msc(\mathscr{S}) = msc\left(\mathscr{S} \backslash \{S\}\right).$$

(2) *If there is an element s of $\mathscr{U}$ which belongs to a unique $S \in \mathscr{S}$, then S belongs to every set cover. In particular:*

$$msc(\mathscr{S}) = 1 + msc\left(del(\mathscr{S}, S)\right).$$

(3) *For all the remaining $S \in \mathscr{S}$, the following holds:*

$$msc(\mathscr{S}) = \min\left\{msc\left(\mathscr{S} \backslash \{S\}\right), 1 + msc\left(del(\mathscr{S}, S)\right)\right\}.$$

Сложеност овог алгоритма је $O(1.9053^n)$, а додатно побољшање се може постићи динамичким програмирањем, где ће се чувати решења за неке израчунате **S** који онда могу да се понове. Тада се повећава просторна сложеност, али онда временска постаје $O(1.8021^n)$.

2. Опис решења

За решавања овог проблема коришћен је генетски алгоритам. За дати број n, који представља број чворова, граф је представљен низом од n целих позитивних бројева, који се међутим посматрају као низови битова, где је елемент на i-том месту i-ти чвор по реду, а у оквиру самих бројева бит на позицији j означава да из чвора i постоји грана према чвору j.

Пример:

1. чвор (1 0 1 1 0) број 22

2. чвор (0 1 0 0 1) број 9

3. чвор (1 0 1 0 0) број 20

4. чвор (1 0 0 1 0) број 18

5. чвор (0 1 0 0 1) број 9

Суседи се извлаче тако што се 1. чвор посматра као највиши бит:

1: [1,3,4]

2: [2,5]

3: [1,3]

4: [1,4]

5: [2,5]

За сваки чвор је укључен и сам тај чвор јер се рачуна да до њега увек може да се дође.

Јединке су такође кодиране као цели бројеви, где сваки бит одређује да ли је дати чвор, где је први чвор опет највиши бит, у доминирајућем скупу. Вредност сваке јединке је број јединица у њеном бинарном запису и циљ је да тај број буде што мањи.

На почетку се одређује почетна популација, али се одмах јавља проблем око тога да ли је дата јединка уопште доминирајући скуп. Међутим, због погодне репрезентације, могуће је једним проласком кроз граф простом битском конјункцијом утврдити да ли чвор i припада доминирајућем скупу или је бар повезан граном са чвором из скупа.

Рецимо, нека је чвор представљен као 1 0 1 1 0 и нека је јединка 0 1 1 0 0

Битском конјункцијом се добија 0 0 1 0 0, тј добија се број различит од нуле. Пошто је сваки чвор заправо представљен чворовима са којима је директно повезан, пресеком са јединком се добију тачно они чворови у доминирајућем скупу са којима је он повезан. Уколико се добије 0, довољно је само одрадити битску дисјункцију и текући чвор укључити у доминирајући скуп, а додатно ће онда бити укључени и чворови са којима је он директно повезан граном.

Након тога се популација сортира због елитизма који се примењује, а затим се врши турнирска селекција, где је победник јединка са најмањим бројем јединица.

Следећа генерација се прави тако што се штити најбољих неколико јединки, а остатак се попуњава јединкама које настају као последица једнопозиционог укрштања и мутације, који су такође подешени за рад са битовима. И овде је такође неопходно да се пази на то да новонастале јединке јесу доминирајући скупови.

Параметри алгоритма се бирају експерименталним путем и од њих понајвише зависи брзина и квалитет самог алгоритма.

3. Резултати алгоритма

За мање уносе (до 20) генетски алгоритам је спорији од обичног алгоритма грубе силе, из простог разлога што је неопходно да се јединке формирају и да прође одређени број итерација. Без обзира на смањење броја итерација, није могуће за те уносе престићи обичан алгоритам грубе силе, а претераним смањењем броја генерација или броја итерација се губи прецизност алгоритма, која је са постојећим параметрима 95-98% на око 1000 тестираних графова.

Алгоритам [1] је далеко бржи за уносе до 60-70 чворова, након чега убрзо почиње да троши доста времена. Генетски алгоритам и са постојећим параметрима избацује довољно добре резултате, па чак и ако не нађе баш минимални доминирајући скуп, налази скуп чија је кардиналност доста близу правог минималног скупа, а троши далеко мање времена за рачунање. Са увећањем параметара алгоритма и прецизност се побољшава, али науштрб ефикасности.

```
Број чворова у графу: 5
1: [ 3 5 ]
2: [ 3 4 ]
3: [ 1 2 5 ]
4: [ 2 ]
5: [ 1 3 ]
Време генетског алгоритма:
0.02544236183166504 сек
Решење пронађено генетским алгоритмом:
[ 4 5 ]
Алгоритам грубе силе:
0.00012731552124023438 сек
Решења:
[ 4 5 ]
[ 3 4 ]
[ 2 5 ]
[ 2 3 ]
[ 1 4 ]
[ 1 2 ]
Решење је тачно
MSC алгоритам:
0.00063323974609375 сек
Резултат MSC алгоритма: 2
```

```
Број чворова у графу: 12
1: [ 2 4 5 6 7 9 ]
2: [ 1 3 4 5 11 ]
3: [ 2 6 7 12 ]
4: [ 1 2 5 7 8 10 11 ]
5: [ 1 2 4 ]
6: [ 1 3 ]
7: [ 1 3 4 8 ]
8: [ 4 7 ]
9: [ 1 10 12 ]
10: [ 4 9 12 ]
11: [ 2 4 ]
12: [ 3 9 10 ]
Време генетског алгоритма:
2.008561372756958 сек
Решење пронађено генетским алгоритмом:
[ 3 4 12 ]
Алгоритам грубе силе:
0.028880834579467773 сек
Решења:
[ 4 6 12 ]
[ 4 6 10 ]
[ 4 6 9 ]
[ 3 4 12 ]
[ 3 4 10 ]
[ 3 4 9 ]
[ 1 4 12 ]
[ 1 3 4 ]
Решење је тачно
MSC алгоритам:
0.0028278827667236633 сек
Резултат MSC алгоритма: 3
```

```
Број чворова у графу: 22
1: [ 7 8 10 15 16 ]
2: [ 5 9 10 11 15 18 20 22 ]
3: [ 8 11 14 16 21 ]
4: [ 5 6 8 9 12 17 18 19 22 ]
5: [ 2 4 6 7 8 10 11 14 16 19 20 21 22 ]
6: [ 4 5 8 11 14 15 17 19 20 22 ]
7: [ 1 5 9 14 18 19 21 22 ]
8: [ 1 3 4 5 6 9 10 16 18 19 ]
9: [ 2 4 7 8 10 12 13 15 18 19 21 22 ]
10: [ 1 2 5 8 9 12 13 14 19 21 22 ]
11: [ 2 3 5 6 12 14 18 20 22 ]
12: [ 4 9 10 11 13 14 19 ]
13: [ 9 10 12 14 21 ]
14: [ 3 5 6 7 10 11 12 13 21 22 ]
15: [ 1 2 6 9 16 17 19 ]
16: [ 1 3 5 8 15 17 19 ]
17: [ 4 6 15 16 19 22 ]
18: [ 2 4 7 8 9 11 19 20 22 ]
19: [ 4 5 6 7 8 9 10 12 15 16 17 18 20 21 ]
20: [ 2 5 6 11 18 19 ]
21: [ 3 5 7 9 10 13 14 19 ]
22: [ 2 4 5 6 7 9 10 11 14 17 18 ]
Време генетског алгоритма:
12.27605128288269 сек
Решење пронађено генетским алгоритмом:
[ 5 9 16 ]
Алгоритам грубе силе:
68.60616278648376 сек
Решења:
[ 14 16 18 ]
[ 14 15 19 ]
[ 14 15 18 ]
[ 10 14 19 ]
[ 10 11 19 ]
[ 9 11 16 ]
[ 9 11 15 ]
[ 6 9 16 ]
[ 6 8 9 ]
[ 5 9 16 ]
[ 3 10 19 ]
Решење је тачно
MSC алгоритам:
0.06853795051574707 сек
Резултат MSC алгоритма: 3
```

```
Број чворова у графу: 65
Време генетског алгоритма:
16.10329580307007 сек
Решење пронађено генетским алгоритмом:
[ 8 22 33 45 65 ]
MSC алгоритам:
35.42807221412659 сек
Резултат MSC алгоритма: 4
```

```
Број чворова у графу: 5
Алгоритам је погодио у 98.70% случајева
```

Тестирање на 1000 случајно генерисаних графова од по 5 чворова

```
Број чворова у графу: 10
Алгоритам је погодио у 96.00% случајева
```

Тестирање на 100 случајно генерисаних графова од по 10 чворова

4. Закључак

Генетски алгоритам је довољно добар за решавање проблема минималног доминирајућег скупа, за разумно време ће успети да обради граф са великим бројем чворова и да врати задовољавајући резултат. Побољшање се може остварити додатним тестирањима и побољшавањем параметара алгоритма, који су тренутно постављени тако да алгоритам, колико је то могуће, буде и ефикасан и квалитетан.

5. Литература

[1] **Elsevier Science** Journal of Discrete Algorithms 2003

[2] **Elsevier Science** Discrete Applied Mathematic