*06.06.2015*

**Paweł Troka, 132334**

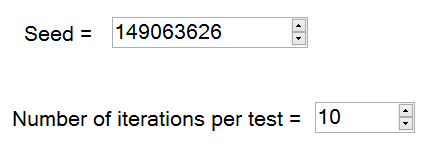
# Projekt

Max-Cut (2-przybliżony)

Algorytmy optymalizacji dyskretnej

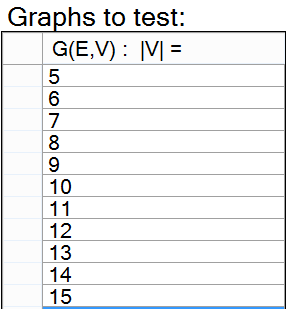
# Wspólne parametry dla wszystkich testów

Wszystkie przeprowadzone testy używają tego samego generatora losowe z tym samym ziarnem ustawionym na wartość 149063626. W każdym teście, aby uzyskać lepszą statystykę każdy pomiar jest powtarzany 10 razy a ostateczny wynik pomiaru to średnia z tych dziesięciu pomiarów.



# Testy algorytmu dokładnego

W testach algorytmu dokładnego wykorzystano 10 wygenerowanych losowo grafów o stosunkowo niedużych rozmiarach od pięciu wierzchołków do 15 wierzchołków. Spowodowane jest to oczywiście bardzo niską wydajnością algorytmu dokładnego.



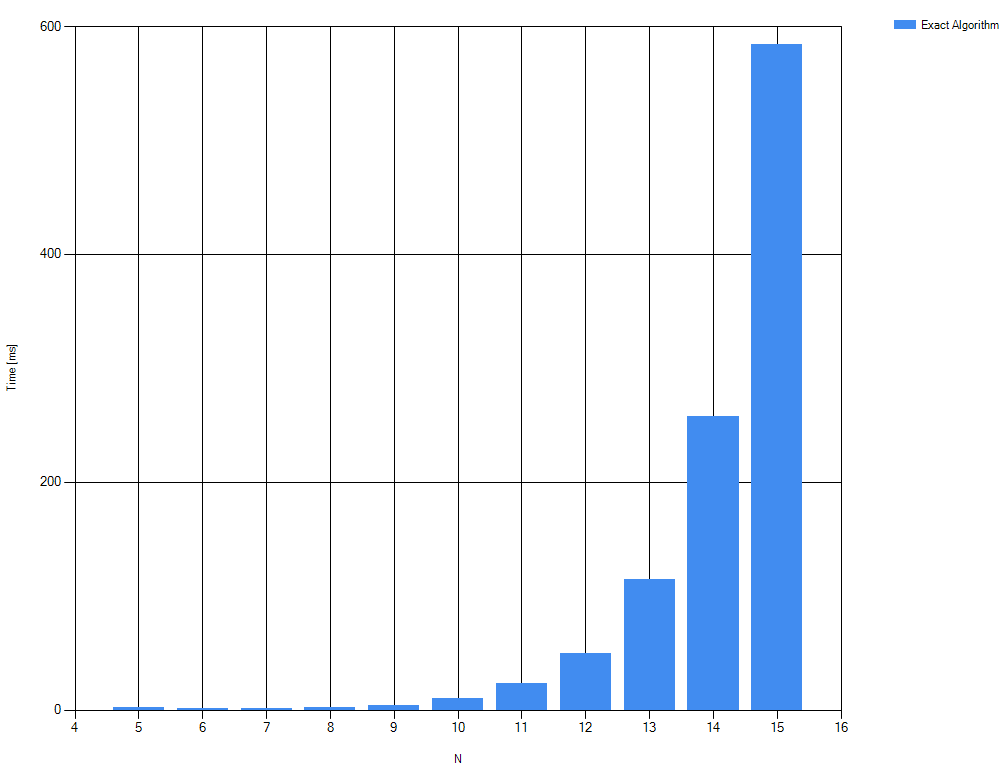
Ustawienia te wygenerowały następujące grafy

|  |
| --- |
| G(|V|=5, |E|=7) |
| G(|V|=6, |E|=7) |
| G(|V|=7, |E|=12) |
| G(|V|=8, |E|=16) |
| G(|V|=9, |E|=16) |
| G(|V|=10, |E|=23) |
| G(|V|=11, |E|=29) |
| G(|V|=12, |E|=33) |
| G(|V|=13, |E|=43) |
| G(|V|=14, |E|=44) |
| G(|V|=15, |E|=52) |

Mimo niskiej wydajności algorytmu dokładnego, ze względu na małe rozmiary wygenerowanych grafów na wyniki nie trzeba było zbyt długo czekać:

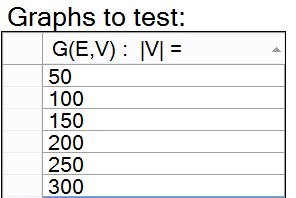
|  |  |  |
| --- | --- | --- |
| Graph | Exact max-cut size | Exact max-cut set |
| G(|V|=5, |E|=7) | 6 | {0, 1, 3} |
| G(|V|=6, |E|=7) | 6 | {1, 2, 3} |
| G(|V|=7, |E|=12) | 9 | {1, 2, 3, 5} |
| G(|V|=8, |E|=16) | 12 | {2, 3, 4, 6, 7} |
| G(|V|=9, |E|=16) | 13 | {1, 2, 6, 7, 8} |
| G(|V|=10, |E|=23) | 17 | {0, 2, 4, 5, 8, 9} |
| G(|V|=11, |E|=29) | 21 | {1, 4, 5, 6, 9, 10} |
| G(|V|=12, |E|=33) | 25 | {1, 3, 5, 7, 8, 11} |
| G(|V|=13, |E|=43) | 30 | {1, 7, 8, 9, 10, 11, 12} |
| G(|V|=14, |E|=44) | 31 | {0, 2, 3, 4, 6, 10, 12, 13} |
| G(|V|=15, |E|=52) | 38 | {1, 3, 5, 7, 8, 9, 10, 12} |

Wyniki czasowe względem liczby wierzchołków grafu można przedstawić na wykresie:



# Testy algorytmu przybliżonego

Ze względu na to, że algorytm 2-przybliżony jest znacznie szybszy niż algorytm dokładny można tutaj sobie pozwolić na znacznie większe grafy. Przetestowano losowo wygenerowane grafy o rozmiarach od 50 do 300 wierzchołków, przy czym następny wygenerowany graf był o 50 wierzchołków większy od poprzedniego.



Generator losowy na podstawie tej oraz poprzednich opcji wygenerował następujące grafy:

|  |
| --- |
| G(|V|=50, |E|=657) |
| G(|V|=100, |E|=2487) |
| G(|V|=150, |E|=5688) |
| G(|V|=200, |E|=10010) |
| G(|V|=250, |E|=15647) |
| G(|V|=300, |E|=22456) |

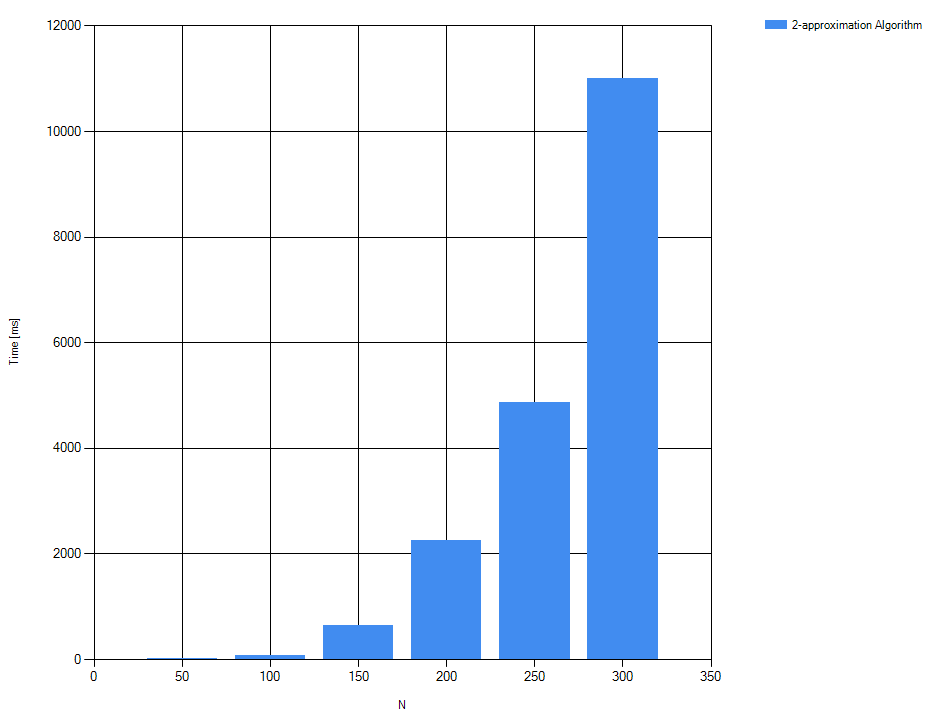
Po odczekaniu dosyć długiego czasu (rzędu kilka do kilkanaście minut) udało się uzyskać następujące rozwiązania:

|  |  |  |
| --- | --- | --- |
| Graph | 2-approximation max-cut size | 2-approximation max-cut set |
| G(|V|=50, |E|=657) | 389 | {0, 2, 3, 9, 10, 11, 13, 14, 15, 16, 17, 19, 20, 21, 22, 30, 32, 34, 29, 48, 5, 31, 1, 38} |
| G(|V|=100, |E|=2487) | 1399 | {1, 7, 8, 9, 12, 13, 15, 16, 17, 19, 20, 22, 23, 24, 25, 26, 27, 29, 30, 31, 32, 33, 35, 36, 38, 39, 40, 41, 42, 43, 44, 49, 52, 53, 60, 66, 82, 86, 97, 45, 50, 91, 74, 85, 93, 61, 55, 46, 5} |
| G(|V|=150, |E|=5688) | 3160 | {18, 22, 24, 27, 30, 31, 32, 34, 35, 41, 43, 45, 46, 47, 49, 50, 54, 56, 57, 58, 59, 60, 62, 66, 68, 69, 70, 72, 73, 78, 79, 82, 88, 91, 95, 104, 111, 112, 117, 125, 144, 113, 119, 87, 129, 149, 102, 4, 133, 141, 0, 17, 115, 98, 121, 122, 96, 145, 138, 127, 12, 84, 23, 105, 83, 134, 74, 108, 11, 77, 101, 5, 36, 142} |
| G(|V|=200, |E|=10010) | 5539 | {6, 7, 11, 15, 18, 26, 29, 32, 34, 38, 39, 40, 42, 43, 44, 50, 51, 55, 56, 57, 59, 60, 63, 64, 65, 66, 68, 69, 70, 71, 74, 76, 77, 79, 80, 86, 87, 90, 91, 92, 94, 95, 96, 98, 99, 101, 102, 103, 108, 109, 113, 119, 123, 126, 151, 179, 166, 173, 129, 3, 160, 172, 144, 127, 186, 146, 159, 174, 164, 168, 106, 148, 167, 8, 22, 157, 169, 37, 118, 188, 196, 4, 28, 2, 33, 1, 138, 189, 47, 9, 36, 134, 191, 93, 176, 153, 175, 97, 35, 177, 194} |
| G(|V|=250, |E|=15647) | 8499 | {17, 38, 41, 43, 44, 46, 48, 50, 51, 52, 53, 56, 60, 61, 65, 66, 67, 69, 70, 71, 72, 73, 75, 76, 81, 82, 84, 85, 86, 87, 88, 89, 92, 94, 95, 96, 97, 98, 100, 104, 105, 106, 108, 109, 110, 111, 113, 114, 116, 118, 120, 121, 122, 123, 124, 127, 130, 131, 146, 148, 149, 150, 165, 170, 176, 179, 181, 188, 193, 231, 195, 220, 221, 132, 187, 6, 3, 135, 151, 156, 157, 198, 205, 1, 139, 226, 216, 190, 196, 23, 219, 233, 245, 154, 177, 2, 25, 163, 202, 134, 10, 26, 241, 203, 4, 207, 215, 138, 144, 8, 55, 54, 167, 90, 128, 47, 158, 0, 42, 161, 184, 242, 36, 229, 159, 20} |
| G(|V|=300, |E|=22456) | 12136 | {8, 9, 32, 34, 36, 44, 48, 51, 55, 56, 58, 59, 62, 67, 68, 70, 71, 74, 75, 76, 77, 81, 82, 83, 88, 89, 92, 93, 96, 97, 98, 99, 101, 103, 104, 105, 106, 107, 108, 109, 110, 112, 113, 116, 117, 118, 120, 122, 124, 125, 126, 129, 131, 133, 135, 137, 138, 140, 141, 142, 143, 145, 147, 150, 151, 156, 163, 165, 167, 174, 177, 179, 193, 194, 202, 236, 239, 255, 280, 0, 181, 243, 262, 216, 175, 206, 3, 16, 197, 245, 212, 222, 11, 279, 210, 166, 31, 251, 287, 21, 161, 289, 294, 214, 2, 266, 244, 253, 39, 258, 198, 211, 148, 220, 273, 12, 37, 241, 246, 15, 169, 176, 50, 240, 27, 162, 78, 215, 53, 65, 87, 57, 153, 271, 149, 265, 213, 217, 90, 42, 159, 270, 132, 152, 291, 4, 6, 188, 221, 233} |

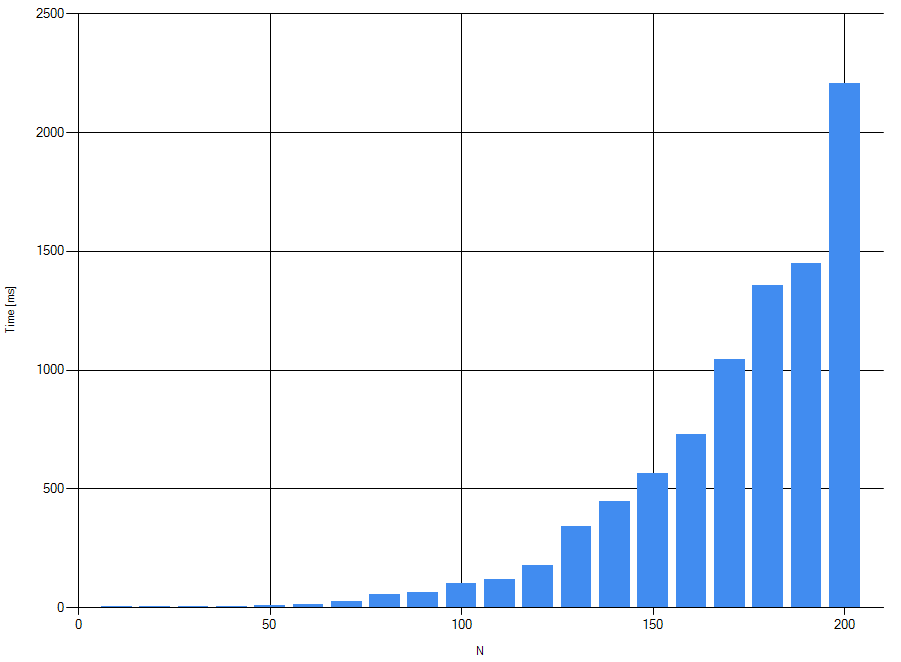
W średnim przypadku rozwiązania uzyskano w następujących czasach w zależności od ilości wierzchołków w grafie:

|  |  |
| --- | --- |
| N | Time (ms) |
| 50 | 8.15443 |
| 100 | 71.86912 |
| 150 | 636.90453 |
| 200 | 2260.70705 |
| 250 | 4873.3684 |
| 300 | 11005.53579 |

Co przedstawiono również na poniższym wykresie:

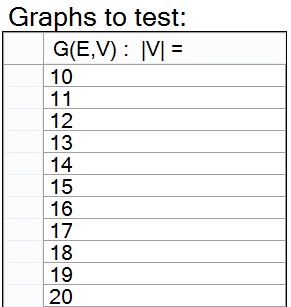


Jak widać algorytm przybliżony skaluje się bardzo dobrze dla rosnącego rozmiaru problemu, wynik poniżej 12 sekund dla grafu o 300 wierzchołkach jest imponujący. Poniżej jeszcze wykres wyników czasowych testu dla mniejszych grafów.

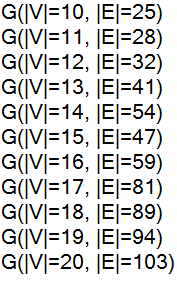


# Testy porównawcze

W testach porównawczych trzeba było pójść na pewien kompromis ze względu na bardzo wolne wykonywanie się algorytmu dokładnego dla grafów większych niż 15. Przetestowano, więc 10 grafów wygenerowanych losowo w rozmiarach od 10 do 20 wierzchołków.



Wygenerowane zostały według wspomnianych ustawień następujące grafy:



Po długim czasie obliczeń uzyskano następujące rezultaty:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Graph | 2-approximation max-cut size | 2-approximation max-cut set | Exact max-cut size | Exact max-cut set |
| G(|V|=10, |E|=25) | 18 | {0, 1, 3, 5, 7} | 18 | {0, 1, 3, 5, 7, 8} |
| G(|V|=11, |E|=28) | 20 | {1, 2, 3, 6, 9, 10} | 20 | {1, 4, 5, 7, 8, 9} |
| G(|V|=12, |E|=32) | 23 | {0, 1, 2, 4, 6, 8, 10} | 23 | {0, 1, 2, 4, 5, 6, 8, 10} |
| G(|V|=13, |E|=41) | 27 | {0, 1, 2, 3, 4, 5, 8} | 29 | {0, 2, 3, 4, 8, 9, 12} |
| G(|V|=14, |E|=54) | 37 | {0, 1, 3, 5, 9, 13, 6} | 37 | {2, 4, 7, 8, 10, 11, 12} |
| G(|V|=15, |E|=47) | 35 | {0, 1, 3, 4, 8, 9, 10, 12} | 36 | {1, 2, 6, 7, 8, 11, 13, 14} |
| G(|V|=16, |E|=59) | 39 | {0, 2, 3, 4, 6, 11, 14, 12} | 41 | {0, 1, 4, 5, 7, 9, 10, 13, 15} |
| G(|V|=17, |E|=81) | 51 | {0, 2, 3, 4, 5, 8, 9, 16} | 53 | {2, 3, 6, 8, 10, 11, 13, 14, 15} |
| G(|V|=18, |E|=89) | 54 | {0, 2, 3, 4, 5, 6, 8, 11, 13} | 57 | {3, 5, 7, 9, 10, 11, 13, 14, 15, 16} |
| G(|V|=19, |E|=94) | 59 | {0, 2, 3, 4, 5, 6, 8, 10, 11} | 61 | {3, 5, 6, 7, 9, 13, 14, 15, 16, 17, 18} |
| G(|V|=20, |E|=103) | 65 | {0, 1, 2, 3, 4, 5, 7, 10, 12, 14} | 68 | {0, 1, 3, 4, 7, 8, 13, 14, 15, 17, 18} |

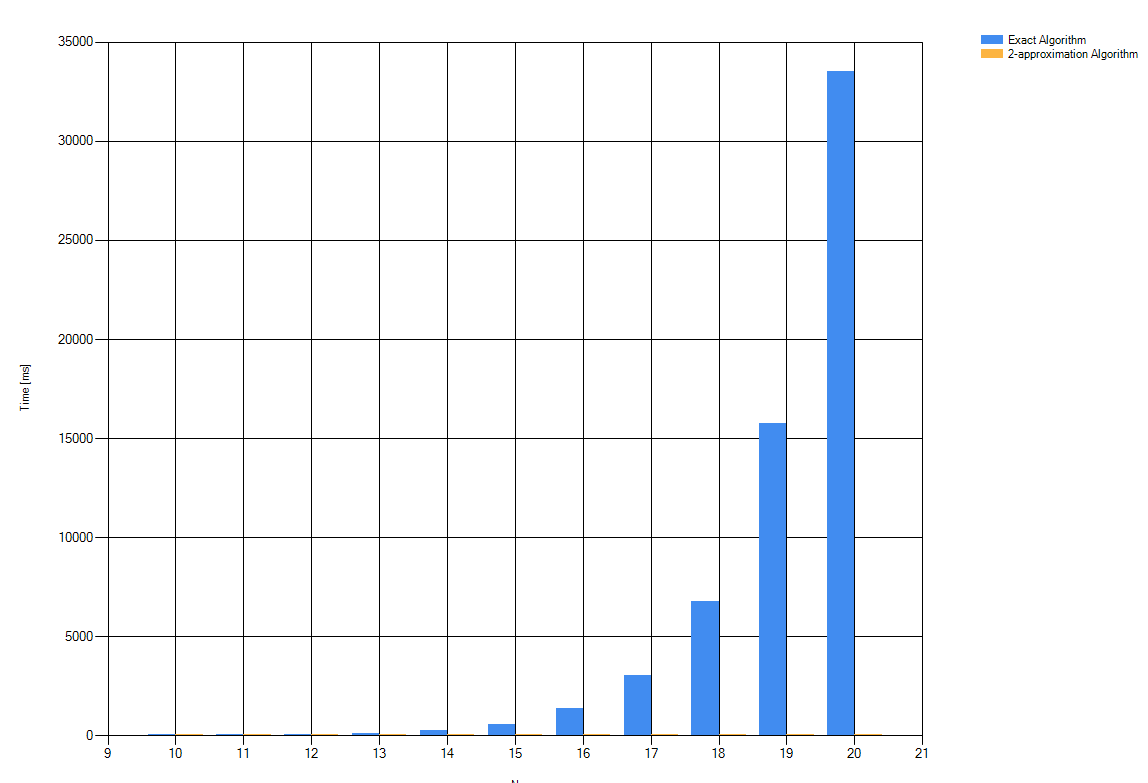
Jak widać algorytm przybliżony myli się zazwyczaj niewiele lub wcale, czyli bez problemu spełnia wymóg o 2-przybliżeniu.

Wyniki wydajnościowe algorytmów przedstawiono poniżej.

|  |  |  |
| --- | --- | --- |
| N | Time of 2-approximation algorithm (ms) | Time of exact algorithm (ms) |
| 10 | 0.16839 | 11.90736 |
| 11 | 0.02602 | 22.22316 |
| 12 | 0.03182 | 50.34129 |
| 13 | 0.02984 | 140.00697 |
| 14 | 0.05813 | 295.78242 |
| 15 | 0.05039 | 585.76954 |
| 16 | 0.07226 | 1368.59743 |
| 17 | 0.08586 | 3051.61262 |
| 18 | 0.06596 | 6771.76768 |
| 19 | 0.08805 | 15733.36275 |
| 20 | 0.08978 | 33499.4028 |

Jak widać dla algorytmu przybliżonego takie rozmiary grafów są banalne, sprawiają za to sporo trudu algorytmowi dokładnemu.

Przedstawiając wyniki na wykresie widać wykładniczy charakter złożoności algorytmu dokładnego (dodając do grafu jeden wierzchołek zwiększamy dwukrotnie czas wykonania algorytmu). Algorytm przybliżony jest za to na tyle szybki, że praktycznie go nie widać na wykresie.



Z ciekawości przetestowano jeszcze algorytmy dla grafów wygenerowanych losowo o ilości wierzchołków od 1 do 18. Poniżej rezultaty.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Graph | 2-approximation max-cut size | 2-approximation max-cut set | Exact max-cut size | Exact max-cut set |
| G(|V|=1, |E|=0) | 0 | {} | 0 |  |
| G(|V|=2, |E|=0) | 0 | {} | 0 |  |
| G(|V|=3, |E|=2) | 2 | {0, 2} | 2 | {0, 2} |
| G(|V|=4, |E|=5) | 3 | {0, 1} | 4 | {1, 3} |
| G(|V|=5, |E|=3) | 3 | {2} | 3 | {0, 1, 3, 4} |
| G(|V|=6, |E|=9) | 7 | {0, 1, 2} | 7 | {3, 4, 5} |
| G(|V|=7, |E|=14) | 10 | {0, 1, 2, 5} | 10 | {0, 3, 4, 6} |
| G(|V|=8, |E|=12) | 10 | {1, 2, 6} | 10 | {0, 3, 4, 5, 7} |
| G(|V|=9, |E|=18) | 13 | {0, 1, 2, 4} | 14 | {1, 3, 4, 6, 8} |
| G(|V|=10, |E|=22) | 15 | {0, 1, 2, 4} | 16 | {2, 3, 5, 7, 8, 9} |
| G(|V|=11, |E|=31) | 21 | {0, 1, 3, 5, 6, 10} | 22 | {1, 2, 6, 7, 8, 9} |
| G(|V|=12, |E|=31) | 19 | {0, 1, 2, 3, 5, 8} | 22 | {1, 4, 5, 7, 8, 9, 10} |
| G(|V|=13, |E|=44) | 29 | {0, 1, 2, 3, 4, 9, 11} | 30 | {2, 3, 6, 7, 8, 10, 12} |
| G(|V|=14, |E|=46) | 32 | {2, 3, 4, 5, 6, 7, 12, 13} | 32 | {2, 3, 4, 5, 6, 7, 12, 13} |
| G(|V|=15, |E|=49) | 32 | {1, 2, 3, 4, 5, 7, 13} | 36 | {0, 3, 4, 6, 8, 9, 12, 14} |
| G(|V|=16, |E|=72) | 47 | {0, 2, 3, 4, 5, 11, 15, 10} | 48 | {2, 6, 7, 8, 9, 10, 11, 12, 13} |
| G(|V|=17, |E|=80) | 52 | {0, 1, 2, 3, 5, 8, 9, 16} | 53 | {0, 2, 4, 6, 10, 11, 12, 14, 16} |
| G(|V|=18, |E|=81) | 55 | {0, 2, 4, 5, 6, 7, 12, 10, 13} | 56 | {0, 2, 4, 6, 7, 10, 11, 12, 13, 14} |

Po raz kolejny okazuje się, że algorytm przybliżony podaje wynik identyczny lub bardzo bliski algorytmowi dokładnemu. W najgorszym z przetestowanych przypadków myli się 1 i 1/3 razy (podał 3 zamiast 4), więc nadal bardzo dobrze mieści się w definicji algorytmu 2-przybliżonego.