Optimization of the Polynomial Greedy Solution for the Set Covering Problem

Stefan Spasovski, Ana Madevska Bogdanova
Faculty of computer science and engineering
"Ss. Cyril and Methodius" University,
Skopje, Macedonia

ABSTRACT

This paper proposes a modification of the known and widely used approximate greedy solution for the Set Cover Problem - GREEDYSETCOVER algorithm. Additionally the already known optimizations are presented and ability of their cohesion with the newly presented algorithm is confirmed. The proposed modification of the algorithm,OPTIMIZEDSETCOVER, in the best case, gives optimal results opposite to the GREEDYSETCOVER algorithm. In the worst case, it gives the same results as the GREEDYSETCOVER solution without going out of the polynomial time boundary.

I. INTRODUCTION

Set Cover Problem (SCP) is known to be NP Hard problem, and furthermore it is NP complete, which gives us an idea to try to get the best solution within the polynomial time frame. If set X is consisted of m elements $\{1, 2, 3...m\}$ and n sets whose union makes up set X are given, the SCP is to find the minimal union that still comprises X. This problem was defined in 1971, and in 1972 was proven to be NP Complete. In 1981 R. Bar-Yehuda and S. Even [1], presented linear time approximation algorithm for the weighted set-covering problem. The algorithm GREEDYSETCOVER as shown on figure 1 is derived from their solution.

```
1:  procedure GREEDYSETCOVER(X, F)
2:      U ← X
3:      C ← ∅
4:      while U ≠ ∅ do        ▷ while uncovered elements exist
5:          select S ∈ F that maximizes |S ∩ U|
6:          U ← U − S
7:          C ← C ∪ {S}
8:      end while
9:      return C                                  ▷ C is the cover
10: end procedure
```

Figure 1: Greedy approximation of the SETCOVER algorithm

The solution in figure 1 makes priority of the subsets that would cover most of the uncovered elements and adds them to the result set. This algorithm finishes in linear time.

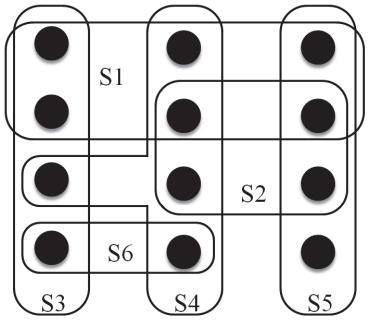

Figure 2: An example depicting a set of 12 elements, with all subsets given, example from [2]

On figure 2 a set of 12 elements (the universe) is presented, including six subsets whose union makes the universe [2]. According to the greedy algorithm, as shown on figure 1, the first chosen set is S_1, (line 5), since it is the biggest set and yet there are no covered elements. The greedy logic is that the bigger the set is, the more uncovered elements it covers. Continuing with S_4 covering 3 elements, than S_5 and at the end the choice will be between S_6 or S_3, since both of them cover the same number of uncovered elements. With this approach the cardinal number of this cover is 4, while the optimal solution, consisted of S_3, S_4 and S_5 has 3 elements.

II. RELATED WORK

The advancement of the solutions for the Set Cover Problem, over the years, can generally be divided in two stages - "Improvement of the ratio" and "Decreasing the cardinality of the cover".

A. Improvement of the ratio

The greedy solution was proposed by Bar-Yehuda and Even [1], in 1981. From than all viable advancements were based on how to achieve better approximation ratio. In that period, the general opinion was that the approximation ratio of the greedy solution was the n-th harmonic number, H(n).

Goldschmidt, Hochbaum and Yu [8] provided a better greedy heuristic and a performance guarantee of $H(n) - \frac{1}{6}$. Series of other improvements appeared on the k-set problem, which is a modification of the Set Cover Problem where every

©2013 Faculty of Computer Science and Engineering

element is limited to be exactly k times in the Set Cover subsets.

In 1997 Slavic [5] showed that the approximation ratio is in fact $ln(m) - ln(ln(m)) + (1)$, where m is the size of the universe.

B. Decreasing the cardinality of the cover

Unlike the aim presented in section II-A, today the improvement of SCP leans toward decreasing the cardinalitiy of the cover and keeping the polynomial time frame.

In 2005 Hassin and Levin proposed an optimization, called greedy algorithm with withdrawals, SETCOVERWITHWITHDRAWALS [6]. They proposed a modification of the GREEDYSETCOVER algorithm, by permitted a subtraction of any subset S from the solution, but only if it is replaced with other subsets that contain the elements covered by S.

In recent publication [7] it is presented that better results can be achieved if an intervention is done in the result set of the GREEDYSETCOVER. They proposed a scanning of every subset S, from the solution against the union of the other subsets of the solution, to determine whether all the elements S covers are already covered by the other sets. If this is true, then S is removed from the solution.

III. METHODS AND METHODOLOGY

In order to present the proposed improved algorithm OPTIMIZEDSETCOVER, we begin with the analysis of GREEDYSETCOVER. If an element e is a member of only one subset S_i i.e. $\exists! i, e \in S_i$, then it is clear S_i must be part of the set cover.

Based on this observation, the OPTIMIZEDSETCOVER can be described with the following steps:
-find all the elements that belong to only one subset
-increment the final result by their number
-remove every element they contain from the universe
-resume with the greedy algorithm.
This method is presented in figure 3.

```
1:  procedure OPTIMIZEDSETCOVER(X, F)
2:      U ← X
3:      C ← ∅
4:      for all a ∈ X do                          ▷ preprocessing
5:          if ∃!S ∈ F that a ∈ S then
6:              U ← U − S
7:              C ← C ∪ {S}
8:          end if
9:      end for
10:     while U ≠ ∅ do      ▷ while uncovered elements exist
11:         select S ∈ F that maximizes |S ∩ U|
12:         U ← U − S
13:         C ← C ∪ {S}
14:     end while
15:     return C                                  ▷ C is the cover
16: end procedure
```

Figure 3: Optimized Greedy Set Covering

IV. RESULTS AND DISCUSSION

A. Analysis of the Optimization OptimizedSetCover

For a given n, the size of the universe, all combinations of subsets were tested whether they compose a cover. In the case of positive outcome, they were used as an input for three algorithms - the naive algorithm, and the algorithms shown in Figure 1 and 3 and we compared the differences in the results among them. Of course not all the inputs are composing a cover of the subset, but their percentage is rapidly declining as the number of elements in the universe is growing.

The total number of combinations of subsets for a set with n elements is described in equation 1. Given a set with n elements, 2^n is the number of all possible subsets. $2^n - 1$ is the number of all possible subsets without the empty set $\emptyset$, and finally $2^{(2^n-1)}$ is the number of all combinations of all subsets.

$$x = 2^{(2^n-1)} \tag{1}$$

Not all combinations of subsets are valid input for the Set Cover problem. The ones that do not provide a cover, are those that fail to have all elements of the universe in the union of their subsets, hence at least one element is missing in all of the subsets. There is a formula to calculate their cardinality 2. It is found due to the correspondence with the $A051381$ array, for which more than one formula has already been published [3].

$$N_{notCover}(n) = \sum_{i=1}^{n} -1^{k+1} * \binom{n}{k} * 2^{2^{n-k}-1} \tag{2}$$

Hence, the number of combinations of subsets that do provide cover is simply the number of all combinations subtracted by the number of the ones that don't. From equations 1 and 2 the equation 3 equation [4] emerges.

$$N_{cover}(n) = \frac{\sum_{i=0}^{n} -1^{k} * \binom{n}{k} * 2^{2^{n-k}}}{2} \tag{3}$$

Because the analysis are done on small number of elements, it is important to show that the percentage of the cases where the combination of subsets that make a cover is increasing so much that the cases where a cover is not obtained can be neglected. For this purpose in Table 1 it is shown how the percentages are progressing (rounding on 9 decimal places) by the increase of the universe size.

Table 1: Probability that a random group of subsets will provide a cover.

# el	# of covers	# of not covers	% of covers
1	1	1	50.000000000%
2	5	3	62.500000000%
3	109	19	85.156250000%
4	32297	471	98.562622070%
5	2147321017	162631	99.992426904%
6	9.22337E+18	12884412819	99.999999860%
7	1.70141E+38	6.45636E+19	100.000000000%
8	5.7896E+76	1.36113E+39	100.000000000%

©2013 Faculty of Computer Science and Engineering

Knowing that only the subsets for which it is known that they will be in the final cover are inserted in the preprocess phase, it is guaranteed that this solution will provide more accurate results than the ordinary greedy solution [1]. We want to answer the remaining question - how much will the results be better when the preprocessing criteria are met ? It is feasible to investigate every possible combination of subsets if the number of elements is small enough. In table 2, it is shown how the OPTIMIZEDSETCOVER performes.

We denote NP as an optimal NP solution result, OSC as the proposed optimized, and GSC as the greedy solution result, so in table 2 in each row we presented the number of solutions that correspond to the criteria in the first column. In the first row the number of the elements in the universe are presented. The last, summary raw, shows the improvement of the proposed Optimized Set Cover algorithm, opposed to the Greedy algorithm [1].

Table 2: Comparison of all possible subset combinations when the universe has up to 5 elements, NP represents the optimal NP solution result , OSC the optimized, and GSC the greedy. The size of the universe is in the first raw

# of elements	2	3	4	5
compose coverage	5	109	32297	2147321017
meet criteria	4	50	3069	2521782
$\|NP = OSC = GSC\|$	4	50	2993	2334726
$\|NP = OSC < GSC\|$	0	0	76	186716
$\|NP < OSC = GSC\|$	0	0	0	340
$\|NP < OSC < GSC\|$	0	0	0	0
$\frac{OSC<GSC}{NP<GSC}$	/	/	100 %	99.818 %

Interesting data can be observed is the last row of table 2. It is the percent of cases in which the greedy solution failed to give the optimal result, but the optimized one did. The first two columns don't have a result since there were no such cases.

The further testing, even for n=6 is impossible. For $n = 6$ the number of tests is 9223372023970362989, so no results were generated.

B. Where the Optimization Meets the Others

Let e belongs to the universe $e \in U$, then S_p is defined as a set of subsets $S_p = \{S_i | !i, e \in S_i\}$, containing all selected subsets in the preprocessing phase of the algorithm presented in figure 3.

Upon merging OPTIMIZEDSETCOVER with SETCOVERWITHWITHDRAWALS in order to make a withdrawal of other subsets that cover the elements of the withdrawn set must be inserted in the solution. Let S_w be the set of withdrawn subsets, then $S_p \cap S_w = \emptyset$ follows, since there is an element $e \exists e \in S \wedge S \in S_p, e \notin (U \backslash S)$, i.e. the element e is in exactly one subset from S_p and nowhere else. From $S_p \cap S_w = \emptyset$ follows that these optimizations are not affecting each other, thus improve different parts of the greedy solution.

When used OPTIMIZEDSETCOVER alongside SETCOVERWITHPOSTPROCESS, let S_{pp} be the set which contains all subsets that were subtracted during the post processing. In order subset $S \in S_{pp}$ to belong in S_{pp} it needs $\forall e \in S, e \in (S_{pp} \backslash S)$, and since $e \in S \in S_p$ such that $e \notin (U \backslash S)$, conclude that $S_p \cap S_{pp} = \emptyset$. By this it is shown that OPTIMIZEDSETCOVER improvements are not subset of those of the SETCOVERWITHPOSTPROCESS i.e. can be used together.

V. CONCLUSION AND FUTURE WORK

In this paper we proposed a modification of the known and widely used approximate greedy solution for the set coverage problem, the OPTIMIZEDSETCOVER algorithm. We proposed an addition to greedy solution of the Set Cover Problem [1], measured the improvements for concrete cardinality of the universe and from the cases for which GREEDYSETCOVER did not return the optimal solution, the vast majority OPTIMIZEDSETCOVER did. We also provided a feasibility proof of using the OSP opposed to other optimization algorithms [7], [6].

REFERENCES

[1] Bar-Yehuda, R. and S. Even. *A linear time approximation algorithm for the weighted vertex cover problem.* Journal of Algorithms, 2:198–203, 1981.

[2] Thomas H. Cormen, Charles E. Leinserson, Ronald L. Rivest, Clifford Stein *Introduction to Algorithms, Second Edition* - The MIT Press, 1033–1038, 2002

[3] V. Jovovic, G. Kilibarda,$http://oeis.org/A051381$,*On the number of Boolean functions in the Post classesF_8^{mu}, Diskretnaya Matematika, 11* (1999), no. 4, 127-138 (translated in Discrete Mathematics and Applications, 9, (1999), no. 6).

[4] $http://oeis.org/A003465$ V. Jovovic , May 30 2004

[5] P. Slavk *A Tight Analysis of the Greedy Algorithm for Set Cover* Journal of Algorithms, Vol 25, Issue 2, November 1997, Pages 237 – 254

[6] R. Hassin and A. Levin *A better-than-greedy approximation algorithm for the minimum set cover problem* - SIAM J. Comput, vol 35 – 2006

[7] M. Alom, S. Das and M. A. Rouf *Performance Evaluation of Vertex Cover and Set Cover Problem using Optimal Algorithm* DUET Journal, Vol. 1, Issue 2, June 2011

[8] O. Goldschmidt, D. S. Hochbaum and G. Yu, *A modified greedy heuristic for the set covering problem with improved worst case bound* Information Processing Letters, 48, 1993, 305- - 310.

©2013 Faculty of Computer Science and Engineering