

# CSE373 Homework 4 Solutions & Hints

## Set Cover Problem (100 pts)

In this solution, I (the course TA) will use C.

### 1 Data structures & Algorithms

Data structures: Firstly, in order to make our backtrack program faster, we use a stack to hold the partial cover and results (e.g. elements covered so far). Secondly, we must use the data structure 'set' in some way. The best way I found is to use a set of bits. We use the first byte (8 bits) to represent whether the first 8 elements are covered or not, and so on and so forth. The advantage here is that we can manipulate set unions in a very efficient way, i.e., just 'bit or' ( $|$  operator) the bytes.

Algorithms: I will present two ways. Both ways use the greedy algorithm to calculate an approximate solution. The most straight forward greedy algorithm is to select at each stage the set that covers the greatest number of remaining elements that are uncovered. Sophisticated analysis shows us that the greedy answer is no larger than the optimal answer times  $(1 + 1/2 + \dots + 1/S)$ , where  $S$  is the largest cardinality (# of elements in a set) of all the sets. When we do backtracking, if we meet a situation which is no better than the best solution so far, we discard it (pruning).

Alg1: sort the sets by their # of elements contained. Then choose the sets one by one. For example:

s-X-12-6

---

12

6

1 2 3 4 5 6

5 6 8 9

1 4 7 10

2 5 7 8 11

3 6 9 12

10 11

---

First sort:

1 2 3 4 5 6 / 2 5 7 8 11 / 5 6 8 9 / 1 4 7 10 / 3 6 9 12 / 10 11

We know the greedy solution is 4, so we enumerate the  $C(6,3)=20$  ways of choosing 3 sets from the total 6. If we can find a set cover, we can improve our solution.

There are many ways of pruning. First, as stated above, if we meet a situation which is no better than the best solution so far, we prune. Second, if the union of current cover and all of the unpicked sets is not the universe, we prune. Third, if # of sets picked + sum of sizes of unpicked sets  $\geq$  best solution, we prune. Fourth, if the current set is a subset of the current cover, we prune.

All of the situations are easy to detect.

Alg2: use 'inverted list'. We again take s-X-12-6 as an example.

Let  $p_i$  be the preposition that set  $i$  has to be chosen.

We know in order to cover element 1, we have to choose set 1 or 3. Thus  $(p_1 \text{ or } p_3)$  holds.

After we run over all 12 elements, we have:

$(p_1 \text{ or } p_3)$  and  $(p_1 \text{ or } p_4)$  and ... and  $(p_4 \text{ or } p_6)$  and  $p_5$

Now we have an idea: let  $P_i$  denote the preposition that element  $i$  has to be covered (e.g.,  $P_3 = p_1 \text{ or } p_5$ ,  $P_{11} = p_4 \text{ or } p_6$ ). Then we sort  $P_i$  by their length (# of sets which have element  $i$  in it), from small to large. Then we do backtracking. At each level, we select the  $P_i$  (here  $P_i$  is considered as a set), with element  $i$  uncovered, and which has the smallest length, as the candidate set (here we choose  $P_{12}$ , i.e., set #5 as the root).

We can also do a simple pruning here. For example, suppose the candidate set of level  $i$  is set # $a$  and # $b$  (i.e.,  $P_i = p_a \text{ or } p_b$ ), and we have already calculated the set # $a$  case. When we calculate case # $b$  at a level deeper than  $i$ , if we want to try set # $a$ , we can prune.

## 2 Results

I use the second algorithm. Environment: Intel Core 2 Duo E4400 2.0GHz; DDR2-667 2.00GB RAM; Microsoft Windows Vista Home Premium with Service Pack 1; Microsoft Visual Studio 2005 Team Suite with Service Pack 1; Compile Option: RELEASE.

File Name	Answer	Greedy Result	Time
s-X-12-6	3	4	<1s
s-k-100-175	19	23	6min10s
s-k-150-225	N/A	41	>10min
s-k-150-250	N/A	25	>10min
s-k-20-30	6	7	<1s
s-k-20-35	6	7	<1s
s-k-200-300	N/A	44	>10min
s-k-30-50	9	11	<1s
s-k-30-55	9	10	<1s
s-k-35-65	10	12	<1s
s-k-40-60	14	15	<1s
s-k-40-80	9	10	<1s
s-k-50-100	9	10	<1s
s-k-50-95	12	14	<1s
s-rg-109-35	22	22	<1s
s-rg-118-30	20	21	<1s
s-rg-155-40	27	27	<1s
s-rg-197-45	30	31	<1s
s-rg-245-50	35	38	<1s
s-rg-31-15	9	9	<1s
s-rg-40-20	10	10	<1s
s-rg-413-75	52	56	<1s
s-rg-63-25	16	17	<1s
s-rg-733-100	76	81	35s
s-rg-8-10	4	4	<1s

### 3 Grading Standards

50% -- program

20% -- algorithm description

30% -- results:

0 pts -- any wrong answer

1 ~ 9 pts -- 1 ~ 4 results

10 ~ 19 pts -- 5 ~ 9 results

20 ~ 29 pts -- 10 ~ 19 results

30 pts -- 20 ~ 25 results