Set Cover Problem

using

Adaptive Large Neighborhood Search (ALNS)

Y. Harun KIVRIL
Sefa KAYRAKLIK

Problem Description

- Identifying the union of sub subsets $S = \{1, 2, 3, \ldots m\}$ with minimum total cost that includes all elements in a given universe $U = \{1, 2, 3, \ldots n\}$ where union of all subsets equals to ***U***. It is proven to be NP-complete by Karp in 1972 [1].

$$min \sum_{i \in S} c_i . x_i$$

$$\sum_{i \in S,\ j \in U} a_{ij} . x_i \geq 1$$

$$x_j \in \{0, 1\}$$

Data Sets

- The datasets are taken from OR- Library
- Two large datasets:
  - SCPNGR1 as test, SCPNGR2 as train
  - 1000 subsets, 10000 elements
- Two small datasets:
  - SCP49 as test, SCP410 as train
  - 200 subsets, 1000 elements

Problem number	Number of subgradient iterations	Optimal value	Total time Cray-1S seconds
4.9	385	641	6.3
4.10	131	514	1.8

Problem	Size	B-Lag	JB-Ann	BC-Gen	PD-Lag	Lower bounds
SCPNRG1	$(1000 \times 10,000)$	184	179	176	176	159.5
SCPNRG2	$(1000 \times 10,000)$	163	158	155	155	141.8

Adaptive Large Neighborhood Search (ALNS)

- Large neighborhood search (LNS) that was invented by Shaw [2]
- ALNS is an extension of LNS that proposed by Pisinger and Røpke [3]
- Improvment by making changes on the solution
  - Destroy Method
  - Repair Method
- Local optima problem
- Using more than one destroy and repair methods
- Updating probabilities to give more chance to the successful ones.

ALNS

procedure ALNS ()

Generate feasible solution x (PURE: can be done in many ways)

$x^b = x$, $p^- =$ (uniform), $p^+ =$ (uniform)

repeat

select destroy and repair methods d (PURE: More than one method) and r using p^- and p^+

$x^t = r(d(x))$

if accept (x^t, x) then $x = x^t$

if $c(x^t) < c(x^b)$ then $x^b = x^t$

update p^- and p^+

until time limit

return x^b

$$p_d^- = \lambda p_d^- + (1-\lambda)\omega$$
$$p_r^+ = \lambda p_r^+ + (1-\lambda)\omega$$

$$\omega = \begin{cases} w_4 \; if\ new\ solution\ is\ new\ global\ best \\ w_3 \; if\ new\ solution\ is\ better\ than\ the\ current\ one \\ w_2 \; if\ new\ solution\ is\ accepted \\ w_1 \; if\ new\ solution\ is\ rejected \end{cases}$$

ALNS

- Acceptance
  - Greedy / Simulated Annealing
- Repair
  - Greedy
  - Regret
- Destroy
  - Random Removal
  - Worst Removal
  - Related Removal

Specialization Description

- Accaptence
  - Allowance for accepting temporary solution
- Repair
  - Greedy Repair $score = \frac{weight}{n}$ eq(1)
  - Other Repair $score = \frac{x_0 + x_1/k}{weight}$ eq(2)

```
procedure GreedyRepair ()
  repeat
    Calculate each subsets' score using eq. 1
    Select the subset that has the lowest score
    Insert the selected subset
  until a feasible solution is obtained
```

```
procedure OtherRepair (k)
  repeat
    Calculate each subsets' score using eq. 2
    Select the subset that has the highest score
    Insert the selected subset
  until a feasible solution is obtained
```

- Destroy (removes n selected subsets from the solution)
  - Frequency

    Score of a subset is determined by the total number of frequencies of the elements in the subset (Def1)
  - Weight

    Score of a subset is determined by weight/(reg+1), reg is the number of unique elements (Def2)
  - Mixed

    Score of a subset is calculated as the summation of the regularized scores from Frequency and Weight (Def3)
  - Random

    It removes n subsets from the solution randomly.

procedure Freq (n)
Calculate scores of subsets in x^t using def1
Select the first n subsets that have the highest score
Remove the selected subsets

procedure Weight (n)
Calculate scores of subsets in x^t using def2
Select the first n subsets that have the highest score
Remove the selected subsets

Procedure Mixed (n)
Calculate scores of subsets in x^t using def3
Select the first n subsets that have the highest score
Remove the selected subsets

procedure Random (n)
Calculate scores of subsets in x^t randomly
Select the first n subsets that have the highest score
Remove the selected subsets

Parameter Tuning

- *Datasets: SCP410 and SCPNRG2*
- *Runing time: 30, 60 seconds respectively*
- *Run three times for each parameter combination*

Parameters	**Values**
n	5 10 20
lambda	0.1 0.5 0.9
w1	0 (fixed)
w2	0.2 (fixed)
w3	0.4 0.6
w4	1 1.5
k	3 10
allowance	0.1 (fixed)

$$val_{rel} = \frac{abs(result - bestknown)}{bestknown} * 100$$

	Average Percentage Gaps							
	lambda=0.1							
	w3=0.4				w3=0.6			
	w4=1		w4=1.5		w4=1		w4=1.5	
	k=3	k=10	k=3	k=10	k=3	k=10	k=3	k=10
n=5	7.157258	6.688325	6.175144	5.813006	7.06427	[illegible]	[illegible]	[illegible]
n=10	5.54868	5.07825	4.50851	5.657305	5.200515	[illegible]	[illegible]	[illegible]
n=20	3.428417	3.498616	3.283196	3.599256	3.495455	[illegible]	[illegible]	[illegible]
	lambda=0.5							
	w3=0.4				[illegible]			
	w4=1		w4=1.5		w4=1		[illegible]	
	k=3	k=10	k=3	k=10	k=3	k=10	[illegible]	[illegible]
n=5	6.167325	6.508005	6.393391	5.423254	6.989247	[illegible]	[illegible]	[illegible]
n=10	5.301155	4.192118	5.46168	3.969212	4.656559	[illegible]	[illegible]	[illegible]
n=20	3.854433	2.887456	[illegible]	2.092981	3.648329	[illegible]	[illegible]	[illegible]
	lambda=0.9							
	w3=0.4				[illegible]			
	w4=1		w4=1.5		w4=1		[illegible]	
	k=3	k=10	k=3	k=10	k=3	k=10	k=3	k=10
n=5	6.054376	6.549259	6.789132	5.32095	5.974862	5.615718	6.728082	5.212492
n=10	5.452531	5.072095	5.389818	4.733079	5.944586	4.786477	4.888947	4.823073
n=20	3.593268	2.850693	3.770427	2.644423	3.752129	2.493379	3.454201	3.238782

Parameters	Values
n	20
lambda	0.5
w1	0 (fixed)
w2	0.2 (fixed)
w3	0.4
w4	1.5
k	10
allowance	0.1 (fixed)

	Average Spreading							
	lambda=0.1							
	w3=0.4				w3=0.6			
	w4=1		w4=1.5		w4=1		w4=1.5	
	[illegible]	k=10	k=3	k=10	k=3	k=10	k=3	k=10
[illegible]	[illegible]	0.6483	0.802342	1.020665	0.711746	1.190213	1.014821	1.139726
[illegible]	[illegible]	0.455949	1.051588	0.987491	1.218468	0.897138	0.900008	1.551664
[illegible]	[illegible]	0.463243	1.341786	0.855813	0.729684	1.322447	0.812207	0.920734
	lambda=0.5							
	w3=0.4				w3=0.6			
	w4=1		w4=1.5		w4=1		w4=1.5	
	[illegible]	k=10	k=3	k=10	k=3	k=10	k=3	k=10
[illegible]	[illegible]	0.350486	1.236617	1.177091	0.471097	0.438695	0.413289	0.963125
[illegible]	[illegible]	0.224664	0.74529	0.798054	0.578064	0.737061	0.983835	0.484824
[illegible]	[illegible]	1.046711	0.9169	1.037556	0.995525	0.943579	0.552589	0.792692
	lambda=0.9							
	w3=0.4				w3=0.6			
	w4=1		w4=1.5		w4=1		w4=1.5	
	k=3	k=10	k=3	k=10	k=3	k=10	k=3	k=10
n=5	1.52453	1.038206	1.048468	0.738379	0.368425	0.775432	0.38913	0.747465
n=10	0.603324	0.612039	0.646892	0.543364	0.741969	0.409936	0.649786	0.925408
n=20	1.518418	0.836227	0.985107	1.197571	1.398915	0.655319	0.982949	0.398836

Test

- Datasets: SCP49 and SCPNRG1

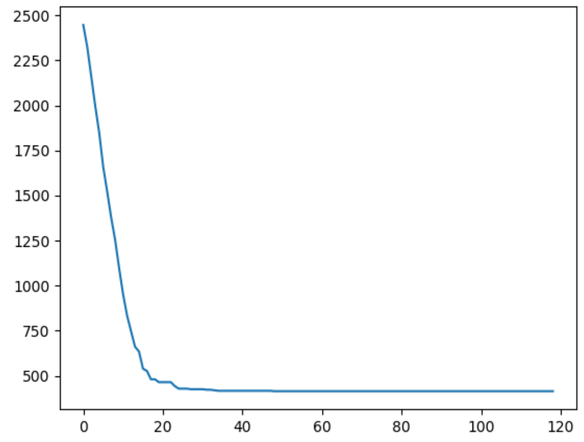
$$val_{rel} = \frac{abs(result - bestknown)}{bestknown} * 100$$

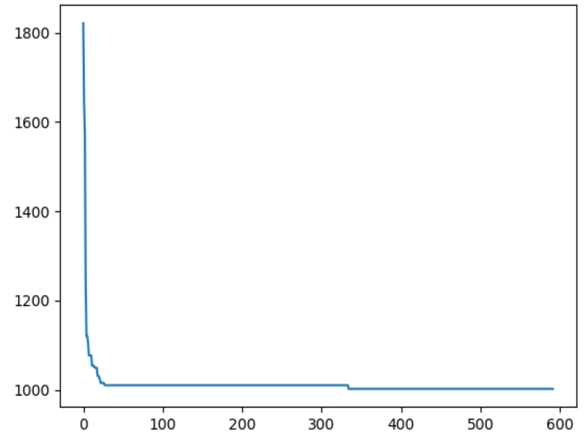
	ITERATIONS										
	1	2	3	4	5	6	7	8	9	10	Mean
SCP49	219	209	207	200	183	194	193	174	200	196	197,5
SCPRNG1	75	58	57	56	54	57	62	58	62	61	60

	RESULTS												
	1	2	3	4	5	6	7	8	9	10	Error	Std.	Best Known
SCP49	1165	1214	1201	1211	1217	1194	1206	1199	1214	1207	87,6443	2,366866	641
SCPRNG1	457	454	457	463	456	459	468	463	467	463	161,761	2,718974	176

Discussion and Conclusion

- Local search
- Four destroy & Two repair methods
- Parameters are tuned
- Not very promising for both datasets
- Getting stuck at local optima
- Even with random removal it is not able to diversify enough
- More advanced repair and destroy methods
- Wrong focus on keeping iteration number high and finding feasible solution quickly





References:

- [1]Richard M. Karp (1972). "Reducibility Among Combinatorial Problems" (PDF). In R. E. Miller; J. W. Thatcher; J.D. Bohlinger (eds.). Complexity of Computer Computations. New York: Plenum. pp. 85–103. doi:10.1007/978-1-4684-2001-2_9
- [2] P. Shaw. Using constraint programming and local search methods to solve vehicle routing problems. In CP-98 (Fourth International Conference on Principles and Practice of Constraint Programming), *volume 1520 of Lecture Notes in Computer Science*, pages 417–431, 1998.
- [3] D. Pisinger and S. Røpke. An adaptive large neighborhood search heuristic for the pickup and delivery problem with time windows. *Transportation Science*, 40(4):455–472, 2006.