UNIVERSIDAD POLITÉCNICA DE CATALUNYA

Frank Wolfe and RSD Algorithms

Mañé Bosch, Ignasi

Professor: Esteve Codina Sancho

Índice

1. Traffic Assignment Problem

Restricted Simplicial Decomposition algorithm works well for some large scale programming problems such as the traffic assignment problem. According to Hern (1984), *"The restricted simpliced algorithm is a restricted version of the Simplicial Decomposition algorithm defined by a parameter r, that restricts the maximum amount of extreme vertexes to be used in the master problem. When the parameter is equal to 1, the algorithm becomes equivalent to the well-known Frank and Wolfe (1956) algorithm and when r is larger than the number of variables, the method is the original simplicial decomposition algorithm. Thus, this algorithm offers to the user a trade-off between memory and computer efficiency.*

A traffic network, defined by its node-link incidence matrix along with the vectors of costs parameters d_{ij} and capacities c_{ij} associated to each arc, is given (Net 5). The volume delay function, expressed as a function of both the volume of flow that goes throw the the arc and the cost parameters, defined as follows:

$$s_{ij} = c_{ij} + d_{ij}x$$

where c_{ij} and d_{ij} are taken from the corresponding values of the arc (i, j).

Since the above link-cost functions are non-negative, non-decreasing and continuous, then the objective function of the traffic assignment problem can be written as follows:

$$minf(v_{ij}) = \sum_{(i,j)\in A} \int_0^{v_{ij}} s_{ij}(x)dx = \sum_{(i,j)\in A} c_{ij}v_{ij} + \frac{cap_{ij}v_{ij}^2}{2}$$

We pick up two origins and two destinations and define four origin destination pairs of flows for our traffic assignment problem. According to the net represented in the Figure 1, the defined pairs O-D flows are the following:

$$O - D : (1, 5) = 10$$
$$O - D : (1, 7) = 90$$
$$O - D : (3, 5) = 35$$
$$O - D : (3, 7) = 70$$

Figure 1 summarizes the scheme of the traffic assignment problem. Our network is made off 7 nodes and 14 arcs that link the seven nodes. Origin nodes are node 1 and node 3 and destination nodes are node 7 and node 5. The parameters of the costs function associated to each arc are given and their values are:

$$c = (4, 1, 4, 3, 4, 5, 4, 5, 2, 1, 4, 4, 2, 5)$$

$$d = (3, 3, 1, 5, 1, 5, 2, 1, 1, 2, 1, 5, 3, 3)$$

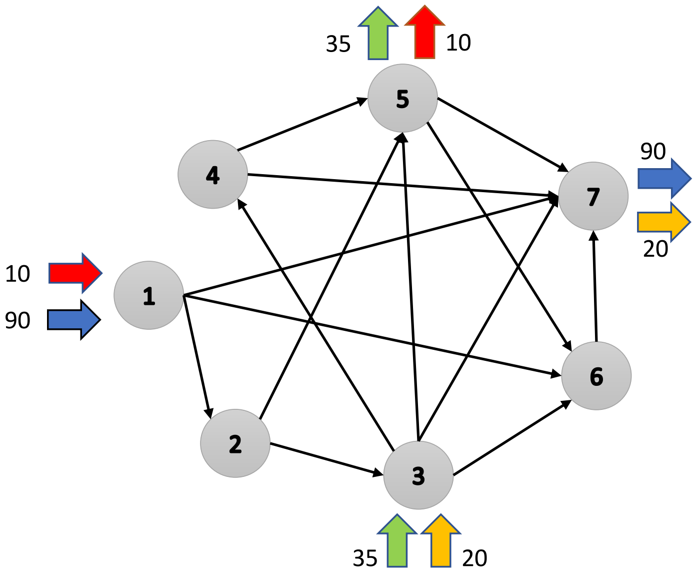

Figura 1: Traffic net scheme

Finally, the node-link incident matrix that summarizes de network represented in Figure 1 is:

1	1	1	0	0	0	0	0	0	0	0	0	0	0
0	-1	0	1	1	0	0	0	0	0	0	0	0	0
0	0	0	-1	0	1	1	1	1	0	0	0	0	0
0	0	0	0	0	0	0	0	-1	1	1	0	0	0
0	0	0	0	-1	0	-1	0	0	0	-1	1	1	0
0	0	-1	0	0	0	0	-1	0	0	0	-1	0	1
-1	0	0	0	0	-1	0	0	0	-1	0	0	-1	-1

Cuadro 1: Node-link incidence matrix of the network

2. Questions and Results

We define $r = 2$ (in our case ρ) to implement the restricted algorithm using ample and MINOS as a solver, which is capable of efficiently optimizing non-linear objective functions.

After defining a maximum of 500 iterations and a relative gap respect to best lower bound of 0.0005, from an initial feasible solution we lunch the algorithm implemented in AMPL and at each iteration the results included in Table 2 are obtained. Notice the maximum allowed number of iterations is reached before the defined relative gap.

The results in Table 2 show the iteration number, the value of the objective function provided by the master problem that we wish to minimize, the relative gap, the cardinality of sets W_x and W_s, and the value of alpha (beta in the implemented algorithm), which is the step length when the cardinality of set W is two or the weight of the convex combination of the extreme points that belong to set W when cardinality of W is larger than two.

	F.Objective	Gap	|Wx|	|Ws|	rho	alphaWx	alphaWs1	alphaWs2
1	9396.35	-2.16266	1	1	2	0.59	0.41	0.00
2	8547.21	-6.35745	1	2	3	0.50	0.35	0.15
3	8543.16	0.05423	1	1	2	0.98	0.02	0.00
4	8537.57	0.04244	1	2	3	0.94	0.03	0.03
5	8533.89	0.04176	1	2	3	0.98	0.01	0.01
6	8530.82	0.04131	1	2	3	0.98	0.02	0.00
7	8521.20	0.04093	1	2	3	0.92	0.03	0.05
8	8515.28	0.03976	1	2	3	0.97	0.02	0.01
9	8512.78	0.03904	1	2	3	0.98	0.00	0.02
10	8506.93	0.02742	1	2	3	0.92	0.05	0.03
11	8501.13	0.02672	1	2	3	0.97	0.01	0.02
12	8499.30	0.02602	1	2	3	0.98	0.01	0.00
13	8495.68	0.02282	1	2	3	0.94	0.02	0.04
14	8492.14	0.02238	1	2	3	0.98	0.01	0.01
15	8491.77	0.02196	1	1	2	1.00	0.00	0.00
16	8490.78	0.02191	1	2	3	0.99	0.01	0.00
17	8487.97	0.02179	1	2	3	0.96	0.02	0.03
18	8485.87	0.02145	1	2	3	0.98	0.01	0.01
19	8484.75	0.02120	1	2	3	0.99	0.00	0.01
20	8483.19	0.02107	1	2	3	0.98	0.02	0.01
495	8421.23	0.00156	1	2	3	1.00	0.00	0.00
496	8421.21	0.00156	1	2	3	1.00	0.00	0.00
497	8421.20	0.00156	1	2	3	1.00	0.00	0.00
498	8421.19	0.00156	1	2	3	1.00	0.00	0.00
499	8421.18	0.00155	1	2	3	1.00	0.00	0.00
500	8421.17	0.00155	1	2	3	1.00	0.00	0.00

Cuadro 2: Results of the algorithm at each iteration for $\rho = 2$

As it is shown in 3,which represents the value of the log relative gap by iteration ,the algorithm was improving the log relative gap very slowly, so the rate of convergence near the solution seems to be very slow for parameter $\rho = 2$. According to the theory, the rate of convergence should improve for larger values of ρ. Actually, the implemented algorithm in AMPL is general in this sense. The reader could try to change the definition of ρ in the data file provided along with this document and check whether the algorithm reaches the final solution faster.

Table 2 shows the number of the iterations needed for the algorithm to reach the relative gap. When $\rho = 1$ the algorithm is equivalent to the Frank-Wolfe algorithm. Notice that as ρ increases the rte of converges also increases, making the algorithm more efficient. However, since it is needed to store more extreme points in set W, the memory consumption is also higher. That is the cost that the user has to pay for increasing speed as we already told.

ρ	Gap	Iterations
1	0.0005	500
2	0.0005	500
3	0.0005	13
4	0.0005	10
5	0.0005	7

Figura 2: Number of iterations to reach ϵ by value of ρ (500 means that the algorithm was forced to stop before)

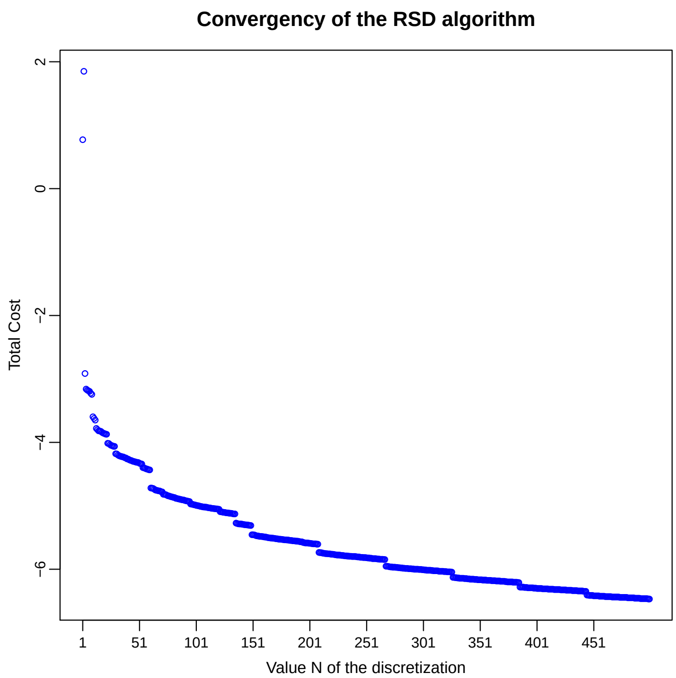

Figura 3: Log Relative gap at each iteration for $\rho = 2$

Now we chose, for each pair origin-destination, two possible paths to evaluate the travel costs and determine whether they are being used or not. The results included in Table 4 show, for each pair origin destination chosen, the costs associated to each path both those obtained through RSD and those obtained directly solving the algorithm.

O	D	Path	cost	Opt
1	5	1-2-3-5	130.039	129.705
1	5	1-2-5	98.2638	98.4597
1	7	1-7	136.771	136.803
1	7	1-6-7	136.973	136.803
3	5	3-5	46.1126	46.0218
3	5	3-4-5	46.2565	46.0218
3	7	3-7	52.9411	53.1197
3	7	3-6-7	105.98	105.852

Figura 4: Costs and Equilibrium

We see that the total costs of the selected paths are very close to the corresponding costs at the optimal solutions. Taking into account that $u^*_{w_j} = min(C_1^{w_j}(h), ..., C_{n_j}^{w_j}(h))$ -that is, the optimal cost for each pair origin-destination- and that in equilibrium only paths whose cost $C_{n_j}^{w_j}$ is equal to $u^*_{w_j}$ are being used, we can try to asses whether the selected paths are being used or not.

For pair origin destination 1-5, the path 1-2-3-5 is not being used since path 1-2-5 has lesser cost. Path 1-2-5, however, is being used since it is the cheapest path to reach node 5 form origin one. Paths 1-7 and 1-6-7 are being used because both are in equilibrium and according to the solution, being 7 a strict destination node, the flow from 1-7 is non empty. Therefore, the cost form both paths is equal to $u^*_{w_j}$. The same happens for paths 3-5 and 3-4-5. However, for pair 3-7, the path 3-6-7 is not being used because the cost of path 3-7, whose cost value is optimal and therefore is being used, is lesser.

Finally, we compare the approximate solution obtained through the RSD algorithm with the global optima that can be obtained solving the problem directly without applying decomposition techniques. Using MINOS and defining the problem Q in AMPL, at the optimal point the following values for the objective function is obtained:

$$\sum_{(i,j)\in A} c_{ij}v_{ij} + \frac{cap_{ij}v_{ij}^2}{2} = 8414{,}157674$$

A value that is very close to the that obtained through the RSD algorithm:

$$\sum_{(i,j)\in A} c_{ij}v_{ij} + \frac{cap_{ij}v_{ij}^2}{2} = 8421{,}234$$

Regarding the flows, Table 5 shows the optimal flows for each arc obtained by both methods. Again the total flows coincide quite well.

Arc	Optimal Flow	RSD Flow
(1, 2)	23.7818	23.7656
(1, 6)	31.9507	31.9365
(1, 7)	44.2675	44.2979
(2, 3)	1.6675	1.75837
(2, 5)	22.1143	22.0072
(3, 4)	26.0327	26.0665
(3, 5)	21.0109	21.1168
(3, 6)	0	0
(3, 7)	9.62395	9.57511
(4, 5)	13.9891	4.1056
(4, 7)	12.0435	11.9609
(5, 6)	0	0
(5, 7)	12.1143	12.2296
(6, 7)	31.9507	31.9365

Figura 5: Results Optimisation problem MINOS)