applied sciences

MDPI

Article

A Methodology for Increasing Convergence Speed of Traffic Assignment Algorithms Based on the Use of a Generalised Averaging Function

Marilisa Botte [1,*], Mariano Gallo [2], Mario Marinelli [2] and Luca D'Acierno [1]

[1] Department of Civil, Architectural and Environmental Engineering, Federico II University of Naples, Via Claudio 21, 80125 Naples, Italy; luca.dacierno@unina.it

[2] Department of Engineering, University of Sannio, Piazza Roma 21, 82100 Benevento, Italy; gallo@unisannio.it (M.G.); mario.marinelli@unisannio.it (M.M.)

* Correspondence: marilisa.botte@unina.it; Tel.: +39-081-768-3356

Received: 17 July 2020; Accepted: 11 August 2020; Published: 17 August 2020

check for updates

Abstract: In this paper, we propose a generalisation of the Method of Successive Averages (MSA) for solving traffic assignment problems. The generalisation consists in proposing a different step sequence within the general MSA framework to reduce computing times. The proposed step sequence is based on the modification of the classic 1/k sequence for improving the convergence speed of the algorithm. The reduction in computing times is useful if the assignment problems are subroutines of algorithms for solving Network Design Problems—such algorithms require estimation of the equilibrium traffic flows at each iteration, hence, many thousands of times for real-scale cases. The proposed algorithm is tested with different parameter values and compared with the classic MSA algorithm on a small and on two real-scale networks. A test inside a Network Design Problem is also reported. Numerical results show that the proposed algorithm outperforms the classic MSA with reductions in computing times, reaching up to 79%. Finally, the theoretical properties are studied for stating the convergence of the proposed algorithm.

Keywords: traffic assignment; MSA algorithm; fixed-point problems; network design

1. Introduction

The equilibrium traffic assignment problem has been widely studied since the early 1950s [1], both with respect to theoretical properties and proposing solution methods and algorithms. The Method of Successive Averages (MSA) is one of the most commonly used algorithms for solving the problem; it was proposed by References [2,3], and its convergence was stated by Reference [4]. The MSA proposed by References [2,3] operates on link flows (MSA-FA). On the other hand, Reference [5] proposed the MSA-CA which operates on link costs; while, Reference [6] devised the MSA-ACO based on Ant Colony Optimisation. Solution algorithms in the case of elastic demand were proposed and analysed by References [5,7,8].

The MSA algorithm at each iteration, k, averages the traffic flows calculated by stochastic uncongested network loading, with weight $1/k$, with the results of the previous iteration, with weight $(k–1)/k$. Alternative step sequences (or weighting methods) were proposed, among others, by References [9–11]. More recently, Reference [12] proposed three revised versions of the traditional MSA method, namely: (i) Restarting MSA (RMSA), where the iteration counter is re-initialised after a specific number of iterations; (ii) Double RMSA (R2MSA), where the iteration counter is re-initialised to a variable value; (iii) Weighted MSA (WMSA), based on the self-regulated averaging method proposed by Reference [13] for solving the stochastic user equilibrium problem. The above-mentioned MSA schemes have been adopted by Reference [14] for solving a stochastic frequency-based assignment considering pre-trip/en-route path choice behaviour. Similarly, Reference [15], which extended the approach proposed in Reference [16], compared different MSA-based resolution techniques in the case of a stochastic assignment with multi-vehicle types (i.e., connected, automated, and autonomous vehicles, as well as traditional ones).

A generalisation of the averaging procedure, not limited to the traffic assignment problem, was provided by Reference [17]; while, Reference [18] proposed a refresh memory method for solving the combined assignment-control problem.

With the computing capabilities of modern PCs, the solution time of the traffic assignment problem is fairly short, also for large-size networks, if the problem has to be solved only once or a few times. Network Design Problems (NDPs) are usually solved by adopting procedures that use assignment algorithms as subroutines (bi-level approach); some reviews on NDPs can be found in References [19–22]. Indeed, for evaluating the objective function corresponding to a solution to the NDP, it is necessary to estimate traffic flows on the network. In this case, in most approaches, the assignment problem has to be solved a number of times equal to the solutions examined by the NDP algorithm; hence, several thousands of times in real-scale applications. Therefore, reducing the computing time of assignment algorithms is an important target since it may lead to a significant reduction in computing times for solving NDPs. In this context, Reference [23] proposed an MSA-based algorithm acting on the traffic flows adopted in the initialisation phases for reducing calculation times of NDPs when the topology of the network configuration remains unchanged.

With respect to the previous literature, in this paper we propose a new method that adopts a variant to the MSA algorithm: It is based on a parameter that can be set for each network to minimise the computing time. Moreover, in order to show its promising performance, it has been compared to other methods proposed in the literature. Finally, after the setting phase, the algorithm can be included within the NDP algorithm, minimising the whole computing time for solving the problem.

The remaining part of the paper is organised as follows: Section 2 introduces the model formulation; the proposed algorithm is analysed in Section 3, and Section 4 summarises the numerical results; Section 5 concludes the paper and outlines the research prospects. In Appendix A the convergence of the proposed algorithm is stated, and Appendix B presents a comparison with some MSA-based algorithms is provide. In Appendices C and D, the Network Design Problem and the algorithm used for its solution in the numerical test are illustrated.

2. Model Formulation

The traffic assignment problem consists of estimating traffic flows on a transportation network. In order to solve this problem, we need three elements: the travel demand (summarised in an Origin-Destination matrix), the transportation supply (modelled with a supply model) and the user path choice behaviour (modelled with a path choice model). Several models can be used for solving the problem depending on the transportation system (road network, mass-transit network, multimodal transportation network, etc.), the assumptions on the travel demand (fixed or variable), the approach to studying the interactions between demand and supply (user equilibrium or dynamic approach), the dependence or otherwise of link costs on flows (uncongested or congested), the assumption on the route choice model (deterministic or stochastic), and so forth. An in-depth analysis of traffic assignment models can be found elsewhere (see, for instance, Reference [24]). Even if the proposed algorithm can also be used to solve other assignment problems, in this paper, we consider the congested road network, fixed demand, user equilibrium approach and stochastic route choice model.

On a road network, the link flow vector, $\boldsymbol{f}$, once the transport demand has been fixed, depends on link costs, and hence, on the link cost vector, $\boldsymbol{c}$:

$$\boldsymbol{f} = \boldsymbol{\varphi}(\boldsymbol{c}) \tag{1}$$

with:

$$\varphi(\boldsymbol{c}) = \boldsymbol{A}\ \boldsymbol{P}(\boldsymbol{A}^{\mathrm{T}}\boldsymbol{c})\ \boldsymbol{d}$$

where $\boldsymbol{A}$ is the link-path incidence matrix, whose components, a_{lp}, are equal to 1 if link l belongs to path p and 0 otherwise; $\boldsymbol{P}$ is the path choice probability matrix, with a column for each od pair and a row for each path p; the generic element, $P_{p,od}$, of this matrix represents the probability that a user will use path p from origin o to destination d (if path p does not connect the od pair, $P_{p,od}$ is equal to 0); $\boldsymbol{d}$ is the demand vector, whose components are the demand values d_{od} for each od pair.

The generic cost, c_l, of link l can be assumed dependent only on the flow on the same link, f_l.

Formally, we can write:

$$\boldsymbol{c} = \chi(\boldsymbol{f}) \tag{2}$$

Substituting (2) in (1), we obtain the following:

$$\boldsymbol{f} = \varphi(\chi(\boldsymbol{f})) \tag{3}$$

which relates flows, $\boldsymbol{f}$, and costs, $\boldsymbol{c}$. Equation (3) is a fixed-point problem; the solution is represented by the equilibrium traffic flows, $\boldsymbol{f}^*$, which is congruent with the corresponding link costs, $\boldsymbol{c} = \chi(\boldsymbol{f}^*)$.

Formally, we can write:

$$\boldsymbol{f}^* = \varphi(\chi(\boldsymbol{f}^*)) \qquad \text{or} \qquad \boldsymbol{f}^* = \boldsymbol{A}\ \boldsymbol{P}(\boldsymbol{A}^{\mathrm{T}}\ \chi(\boldsymbol{f}^*))\ \boldsymbol{d} \tag{4}$$

In this case, it may be stated, under some assumptions, that the solution to the fixed-point problem (4) exists and is unique [5] and that the MSA algorithm converges to it [4].

3. Solution Algorithm

In order to solve the fixed-point problems (4) we propose an algorithm based on the MSA general framework. The first formulation of MSA [2] generates a succession of feasible link flow vectors, $\boldsymbol{f}^{\,k}$, starting from an initial solution (usually $\boldsymbol{f}^{\,0} = \mathbf{0}$). At each iteration k, the solution $\boldsymbol{f}^k$ is generated by combining traffic flows obtained by a Stochastic Uncongested Network (SUN) assignment, $\boldsymbol{f}_{\mathrm{SUN}}{}^k = \varphi_{\mathrm{SUN}}(\boldsymbol{c}^k)$, with the previous solution, $\boldsymbol{f}^{\,k-1}$

At each iteration k, in order to generate the solution $\boldsymbol{f}^k$, the results of the SUN assignment, $\boldsymbol{f}_{\mathrm{SUN}}{}^k$, are averaged, with weight $1/k$, with the results of the previous iteration, $\boldsymbol{f}^{\,k-1}$, with weight $(k-1)/k$:

$$\boldsymbol{f}^k = ((k-1)/k) \cdot \boldsymbol{f}^{\,k-1} + (1/k) \cdot \boldsymbol{f}_{\mathrm{SUN}}{}^k \tag{5}$$

that is:

$$\boldsymbol{f}^k = \boldsymbol{f}^{\,k-1} + 1/k\ (\boldsymbol{f}_{\mathrm{SUN}}{}^k - \boldsymbol{f}^{\,k-1}) \tag{6}$$

Since the algorithm works directly on flows, it is also known as MSA-FA (Flow Averaging). Algorithm 1 reports the code of the traditional MSA-FA algorithm, where $\boldsymbol{\varepsilon}$ represents the stop threshold.

It is worth noting that the averages applied to the flows belonging to successive iterations have to be considered an algorithmic trick and should not be confused with the weights that simulate the learning process in the dynamic assignment models. Indeed, the MSA algorithms apply to stationary models, while the simulation of learning processes (comparison between expected and experienced performances) are typical of day-to-day dynamics models [24,25].

Since, as shown in Appendix B, most of the MSA-based algorithms proposed in the literature for solving the traffic assignment problem are based on a modification of the weight formulation (i.e., term $1/k$ in the traditional MSA algorithm), our proposal consists in providing a further variant of the MSA-FA algorithm substituting the term k in Equation (5) or (6) with the following function:

$$\xi(k) = 1 + (k\text{-}1) \cdot \eta \tag{7}$$

where η is a parameter between 0 and 1; for $\eta = 1$ we obtain the classic MSA algorithm, and for $\eta = 0$ we obtain that $f^k = f_{\mathrm{SUN}}{}^k$ (in this case the algorithm does not consider at each iteration the results of the previous ones). This method tends to muffle the weight given to $f^{\,k-1}$ (with $\eta < 1$) with respect to the corresponding weight given with the classic MSA algorithm. The weights of $f^{\,k-1}$, w_{-1}, and $f_{\mathrm{SUN}}{}^k$, w_{SUN}, at each iteration k become:

$$w_{-1} = ((k\text{-}1) \cdot \eta)/(1 + (k\text{-}1) \cdot \eta) \tag{8}$$

$$w_{\mathrm{SUN}} = 1/(1 + (k\text{-}1) \cdot \eta) \tag{9}$$

Algorithm 1 Traditional MSA-FA algorithm

```
1:  $k = 0$
2:  $f^0 = \mathbf{0}$
3:  do while $|f_{\mathrm{SUN}}{}^k - f^{\,k-1}|/f^{\,k-1} \geq \varepsilon$
4:      $k = k + 1$
5:      $c^k = \chi(f^{\,k-1})$
6:      $f_{\mathrm{SUN}}{}^k = \varphi_{\mathrm{SUN}}(c^k)$
7:      $f^k = f^{\,k-1} + 1/k \cdot (f_{\mathrm{SUN}}{}^k - f^{\,k-1})$
8:  loop
9:  $f^* = f^k$
10: end
```

In Algorithm 2 the code of the generalised MSA-FA algorithm (i.e., GMSA-FA) is reported. Figure 1 reports the values at each iteration k of w_{SUN} (equal to $1 - w_{-1}$) for values of ηbetween 1 (classic MSA) and 0 and for the first 100 iterations.

Algorithm 2 Generalised MSA-FA algorithm

```
1:  $k = 0$
2:  $f^0 = \mathbf{0}$
3:  do while $|f_{\mathrm{SUN}}{}^k - f^{\,k-1}|/f^{\,k-1} \geq \varepsilon$
4:      $k = k + 1$
5:      $c^k = \chi(f^{\,k-1})$
6:      $f_{\mathrm{SUN}}{}^k = \varphi_{\mathrm{SUN}}(c^k)$
7:      $f^k = f^{\,k-1} + 1/\xi(k) \cdot (f_{\mathrm{SUN}}{}^k - f^{\,k-1})$
8:  loop
9:  $f^* = f^k$
10: end
```

It can be noted that the weight of $f_{\mathrm{SUN}}{}^k$ decreases less rapidly for lower values of parameter η. Convergence of the proposed algorithm is stated in Appendix A, where the algorithm is shown to converge for each value of $\eta > 0$.

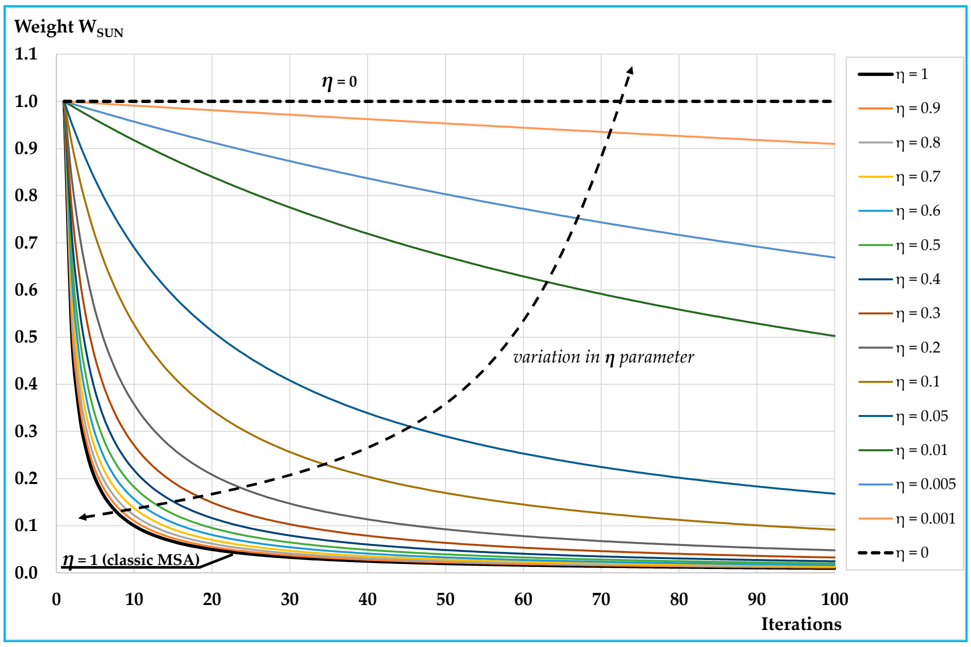

Figure 1. Values of weights w_{SUN}.

4. Numerical Results

Numerical tests were conducted on a small test network and on two real-scale networks (an urban and a rural network). Each network, by means of the graph theory, was represented as a set of links and nodes. The links represent the phases of the trips (as for instance, the roads) and the nodes represent the transitions between two consecutive links. Among the nodes, it is possible to identify the centroid nodes that represent the origin and destination of each trip.

4.1. Small Network

The small network consisted of 16 oriented links (see Figure 2), whose link performances (i.e., the link travel times) were expressed by means of a BPR function [26], that is:

$$t_l = (L_l/v_{0,l}) \cdot (1 + \alpha \cdot (f_l/Cap_l)^{\beta}) \tag{10}$$

where t_l is the travel time associated to link l; L_l is the length of link l; $v_{o,l}$ is the free-flow speed associated to link l; f_l is the link flow associated to link l; Cap_l is the capacity of link l; α and β are function parameters. The main features of the considered network are summarised in Table 1.

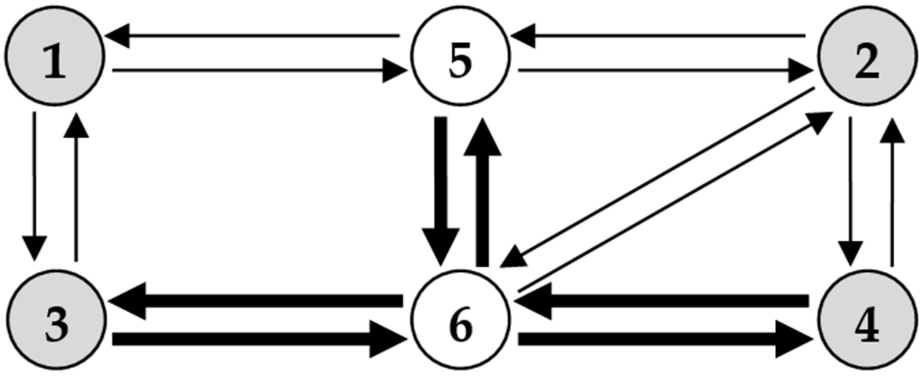

Figure 2. Small network.

Table 1. Features of the small network.

Link	Length (km)	Capacity (veh/h)	Free-Flow Speed (km/h)	BPR Param. α	BPR Param. β
1–5	2	1800	60	2.0	4.0
5–1	2	1800	60	2.0	4.0
1–3	2	1800	50	2.0	4.0
3–1	2	1800	50	2.0	4.0
3–6	3	3600	90	0.2	4.0
6–3	3	3600	90	0.2	4.0
5–6	2	3600	80	0.2	4.0
6–5	2	3600	80	0.2	4.0
5–2	4	1800	50	2.0	4.0
2–5	4	1800	50	2.0	4.0
6–2	5	1800	70	2.0	4.0
2–6	5	1800	70	2.0	4.0
6–4	5	3600	90	0.2	4.0
4–6	5	3600	90	0.2	4.0
4–2	3	1800	60	2.0	4.0
2–4	3	1800	60	2.0	4.0

Bold links have higher capacity and free-flow speed with respect to the others. We assume that nodes 1, 2, 3 and 4 are also the origins and destinations of the trips, according to the OD vector, $\boldsymbol{d}$, reported in Table 2. In the tests, we adopted Multinomial Logit as the stochastic path choice model and assumed that the cost of each link was equal to the running time expressed in minutes with parameter θ equal to 0.5 (coefficient of variation about 0.2). We implemented the proposed algorithm in Visual Basic. We tested the proposed algorithm with the following values of parameter η: 1 (classic MSA), 0.9, 0.8, 0.7, 0.6, 0.5, 0.4, 0.3, 0.2, 0.1, 0.05, 0.01, 0. For each value, we tested the OD vector reported in Table 2 multiplied by the following coefficients: 0.6, 0.8, 1.0, 1.2, 1.4, 1.6, 1.8, and 2.0. The use of different OD vectors, as well as different multipliers, is to analyse the algorithm's performance for different congestion levels. Indeed, the vectors thus obtained produce average saturation degrees (flow/capacity ratio) on the network between 0.249 and 0.838, as reported in Table 3. The convergence threshold was assumed to be equal to 1% for all tests.

Table 2. Origin-destination vector of the small network.

Origin	Destination	OD Flow (veh/h)
1	2	1700
2	1	600
1	3	500
3	1	700
1	4	650
4	1	100
2	3	800
3	2	450
2	4	750
4	2	450
3	4	1250
4	3	200

Table 3. Average saturation degrees on the small network.

Matrix	0.6 $\boldsymbol{d}$	0.8 $\boldsymbol{d}$	1.0 $\boldsymbol{d}$	1.2 $\boldsymbol{d}$	1.4 $\boldsymbol{d}$	1.6 $\boldsymbol{d}$	1.8 $\boldsymbol{d}$	2.0 $\boldsymbol{d}$
Av. Sat. Degree	0.249	0.332	0.415	0.499	0.583	0.668	0.753	0.838

In all cases, the proposed algorithm was able to reduce the iterations for the convergence with some values of η lower than 1 (MSA). It can be noted that, upon reducing the value of η, the number of iterations starts to decrease until a minimum value is reached. The number of iterations then begins to increase until the case of $\eta = 0$ when, according to also to the theoretical properties examined in Appendix A, the algorithm may not converge.

In Table 4, the number of iterations required for convergence for all cases is reported (the algorithm was stopped in any event when the iterations reached 999); minimum values are written in bold. In the same table, the percentage variations in terms of iteration number are reported. However, it is worth noting that all variants of the algorithm provide the same results. The difference among them is only the number of iterations necessary to achieve the convergence condition. In Figure 3, the results of Table 4 are summarised.

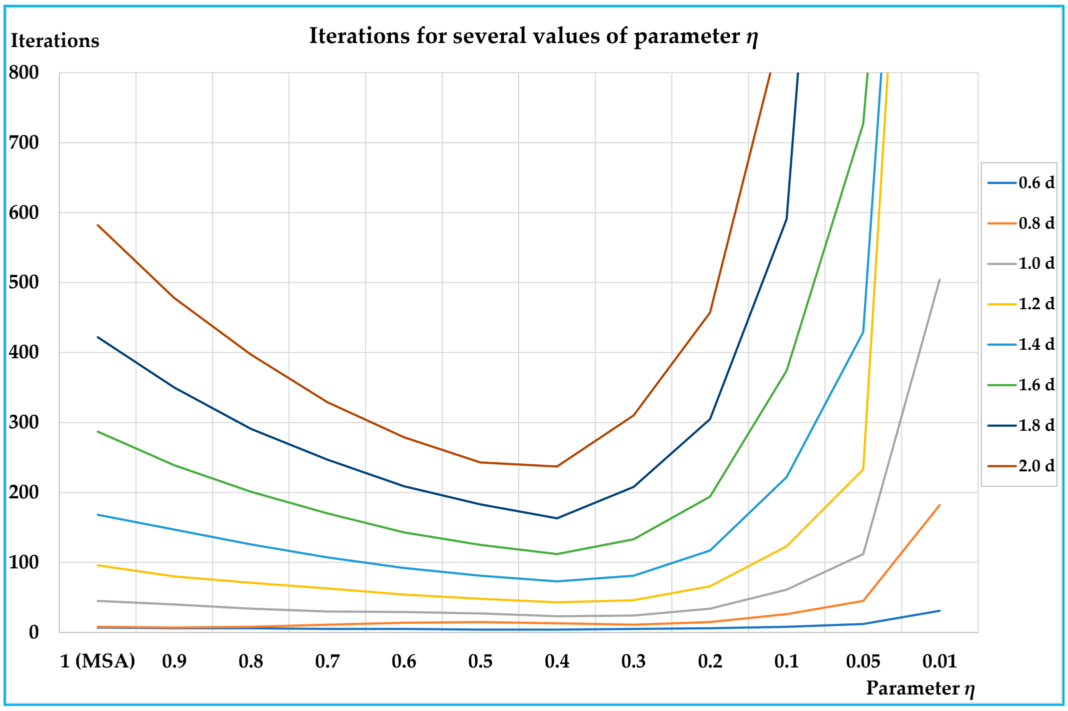

Figure 3. Results on the small network.

Table 4. Results on the small network.

	Iterations Needed for Convergence							
Matrix	**0.6 *d***	**0.8 *d***	**1.0 *d***	**1.2 *d***	**1.4 *d***	**1.6 *d***	**1.8 *d***	**2.0 *d***
η = 1(MSA)	7	8	45	96	168	287	422	582
η = 0.9	6	**7**	40	80	147	239	350	478
η = 0.8	6	8	34	71	126	201	291	397
η = 0.7	5	11	30	63	107	170	247	329
η = 0.6	5	14	29	54	92	143	209	279
η = 0.5	**4**	15	27	48	81	125	183	243
η= 0.4	**4**	13	**23**	**43**	**73**	**112**	**163**	**237**
η = 0.3	5	11	24	46	81	133	208	310
η = 0.2	6	15	34	66	117	194	305	457
η = 0.1	8	26	61	123	222	374	591	892
η = 0.05	12	45	112	233	429	727	>999	>999
η = 0.01	31	182	504	>999	>999	>999	>999	>999

Table 4. *Cont.*

	Percentage Variations with Respect to MSA							
Matrix	**0.6 *d***	**0.8 *d***	**1.0 *d***	**1.2 *d***	**1.4 *d***	**1.6 *d***	**1.8 *d***	**2.0 *d***
η = 1(MSA)	-	-	-	-	-	-	-	-
η = 0.9	−14%	**−13%**	−11%	−17%	−13%	−17%	−17%	−18%
η = 0.8	−14%	0%	−24%	−26%	−25%	−30%	−31%	−32%
η = 0.7	−29%	38%	−33%	−34%	−36%	−41%	−41%	−43%
η = 0.6	−29%	75%	−36%	−44%	−45%	−50%	−50%	−52%
η = 0.5	**−43%**	88%	−40%	−50%	−52%	−56%	−57%	−58%
η = 0.4	**−43%**	63%	**−49%**	**−55%**	**−57%**	**−61%**	**−61%**	**−59%**
η = 0.3	−29%	38%	−47%	−52%	−52%	−54%	−51%	−47%
η = 0.2	−14%	88%	−24%	−31%	−30%	−32%	−28%	−21%
η = 0.1	14%	225%	36%	28%	32%	30%	40%	53%
η = 0.05	71%	463%	149%	143%	155%	153%	-	-
η = 0.01	343%	2175%	1020%	-	-	-	-	-

In bold are reported the best results.

Note that the reduction in the number of iterations, and hence, in computing time, provided by the proposed algorithm can be very significant for medium-high congested networks, i.e., up to −61% compared with the traditional version of the MSA algorithm.

4.2. Real-Scale Urban Network

The real-scale urban test network is the network of Benevento, a town in the south of Italy with about 61,000 inhabitants. The network, as represented in Figure 4, has 1577 oriented links, comprising about 216 km of roads, 678 nodes and 80 centroids (66 internal and 14 external). The peak-hour OD matrix was made available by studies carried out while drawing up the Urban Traffic Plan. The route choice model is Multinomial Logit with an implicit enumeration of paths according to the algorithm proposed in Reference [27].

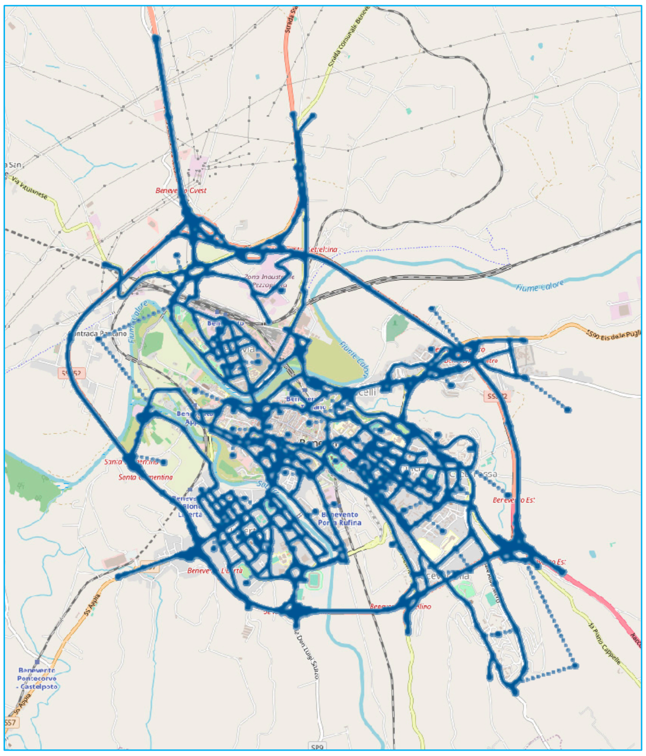

Figure 4. Urban real-scale network.

In order to test the proposed algorithms under several network congestion levels, over the available OD matrix, seven other matrices were generated, multiplying all cells of the OD matrix by the following factors: 0.6, 0.8, 1.2, 1.4, 1.6, 1.8, and 2.0. The corresponding average saturation degrees are summarised in Table 5.

Table 5. Average saturation degrees on the urban real-scale network.

Matrix	**0.6 *d***	**0.8 *d***	**1.0 *d***	**1.2 *d***	**1.4 *d***	**1.6 *d***	**1.8 *d***	**2.0 *d***
Av. Sat. Degree	0.246	0.324	0.400	0.472	0.545	0.618	0.691	0.764

On this network, we tested the proposed algorithm with the same values of parameter η as those adopted for the small test network; we tested the algorithm with two different values of the coefficient of variation (standard deviation to mean ratio), C_v, equal to 0.10 and 0.05. Moreover, in these cases, the proposed algorithm was able to reduce the iterations for convergence with some values of η lower than 1 (MSA). In this case, the best performances were obtained with values between 0.4 and 0.6 (see Table 6) for C_v = 0.10 and between 0.3 and 0.6 (see Table 7) for C_v = 0.05; the reductions in the number of iterations reached −76% (C_v = 0.10) and −79% (C_v = 0.05) with respect to the traditional version of the MSA algorithm. In Figures 5 and 6, the results of Tables 6 and 7 are illustrated. Moreover, in this case, the utility of the proposed algorithm is more significant for medium-high demand levels.

Table 6. Results on the urban real-scale network (C_v = 0.10).

	Iterations Needed for Convergence							
Matrix	**0.6 *d***	**0.8 *d***	**1.0 *d***	**1.2 *d***	**1.4 *d***	**1.6 *d***	**1.8 *d***	**2.0 *d***
η = 1(MSA)	5	8	17	32	50	52	55	61
η = 0.9	5	8	14	25	36	38	39	43
η = 0.8	5	8	12	19	27	28	28	32
η = 0.7	5	8	10	15	20	20	21	29
η = 0.6	5	**7**	**9**	12	15	**15**	**20**	26
η = 0.5	**4**	**7**	**9**	**11**	**12**	**15**	21	**25**
η = 0.4	**4**	**7**	10	**11**	**12**	18	24	27
η = 0.3	**4**	8	11	13	15	21	30	33
η = 0.2	**4**	10	14	17	20	28	41	45
η = 0.1	**4**	14	22	28	33	49	74	79
η = 0.05	**4**	22	37	48	58	88	136	146
η = 0.01	**4**	77	147	196	244	385	619	664
	Percentage Variations with Respect to MSA							
Matrix	**0.6 *d***	**0.8 *d***	**1.0 *d***	**1.2 *d***	**1.4 *d***	**1.6 *d***	**1.8 *d***	**2.0 *d***
η = 1(MSA)	-	-	-	-	-	-	-	-
η = 0.9	0%	0%	−18%	−22%	−28%	−27%	−29%	−30%
η = 0.8	0%	0%	−29%	−41%	−46%	−46%	−49%	−48%
η = 0.7	0%	0%	−41%	−53%	−60%	−62%	−62%	−52%
η = 0.6	0%	**−13%**	**−47%**	−63%	−70%	**−71%**	**−64%**	−57%
η = 0.5	**−20%**	**−13%**	**−47%**	**−66%**	**−76%**	**−71%**	−62%	**−59%**
η = 0.4	**−20%**	**−13%**	−41%	**−66%**	**−76%**	−65%	−56%	−56%
η = 0.3	**−20%**	0%	−35%	−59%	−70%	−60%	−45%	−46%
η = 0.2	**−20%**	25%	−18%	−47%	−60%	−46%	−25%	−26%
η = 0.1	**−20%**	75%	29%	−13%	−34%	−6%	35%	30%
η = 0.05	**−20%**	175%	118%	50%	16%	69%	147%	139%
η = 0.01	0%	0%	−18%	−22%	−28%	−27%	−29%	−30%

In bold are reported the best results.

Table 7. Results on the urban real-scale network (C_v = 0.05).

	Iterations Needed for Convergence							
Matrix	**0.6 *d***	**0.8 *d***	**1.0 *d***	**1.2 *d***	**1.4 *d***	**1.6 *d***	**1.8 *d***	**2.0 *d***
η = 1(MSA)	5	8	20	47	49	49	53	59
η = 0.9	5	7	16	34	36	36	38	42
η = 0.8	4	7	13	25	26	26	34	38
η = 0.7	4	6	10	18	19	21	35	39
η = 0.6	4	6	9	14	**14**	**20**	**31**	**34**
η = 0.5	**3**	6	**8**	**10**	**14**	24	34	36
η = 0.4	**3**	**5**	9	**10**	16	27	40	43
η = 0.3	**3**	6	11	**10**	20	34	50	54
η = 0.2	**3**	8	14	12	27	46	71	76
η = 0.1	**3**	11	23	26	49	84	132	141
η = 0.05	4	17	39	45	90	158	251	269
η = 0.01	4	54	160	186	402	724	>999	>999
	Percentage Variations with Respect to MSA							
Matrix	**0.6 *d***	**0.8 *d***	**1.0 *d***	**1.2 *d***	**1.4 *d***	**1.6 *d***	**1.8 *d***	**2.0 *d***
η = 1(MSA)	-	-	-	-	-	-	-	-
η = 0.9	0%	−13%	−20%	−28%	−27%	−27%	−28%	−29%
η = 0.8	−20%	−13%	−35%	−47%	−47%	−47%	−36%	−36%
η = 0.7	−20%	−25%	−50%	−62%	−61%	−57%	−34%	−34%
η = 0.6	−20%	−25%	−55%	−70%	**−71%**	**−59%**	**−42%**	**−42%**
η = 0.5	**−40%**	−25%	**−60%**	**−79%**	**−71%**	−51%	−36%	−39%
η = 0.4	**−40%**	**−38%**	−55%	**−79%**	−67%	−45%	−25%	−27%
η = 0.3	**−40%**	−25%	−45%	**−79%**	−59%	−31%	−6%	−8%
η = 0.2	**−40%**	0%	−30%	−74%	−45%	−6%	34%	29%
η = 0.1	**−40%**	38%	15%	−45%	0%	71%	149%	139%
η = 0.05	−20%	113%	95%	−4%	84%	222%	374%	356%
η = 0.01	−20%	575%	700%	296%	720%	1378%	-	-

In bold are reported the best results.

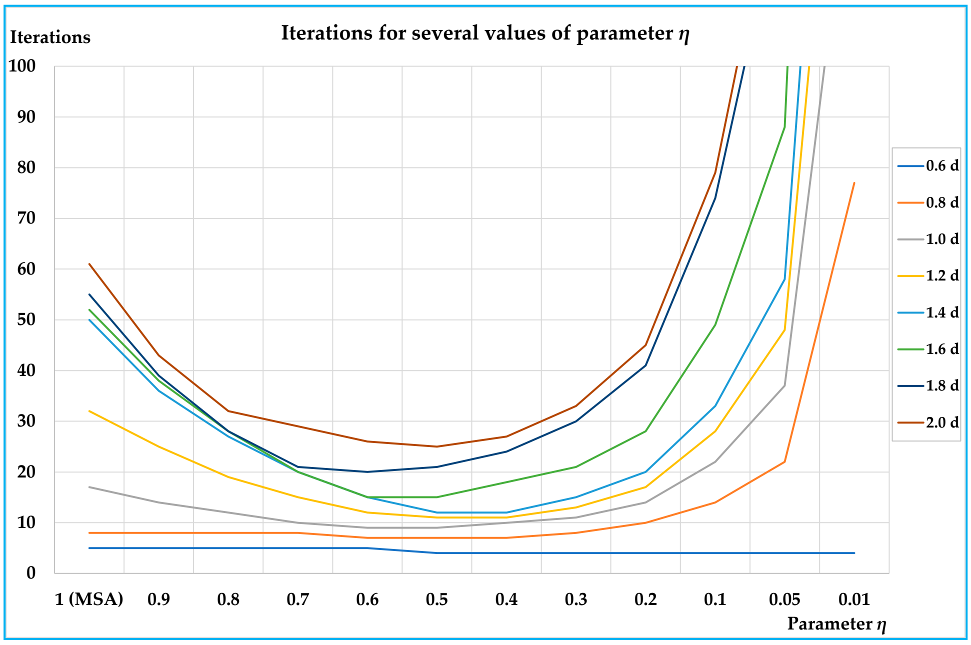

Figure 5. Results on the urban real-scale network (C_v = 0.10).

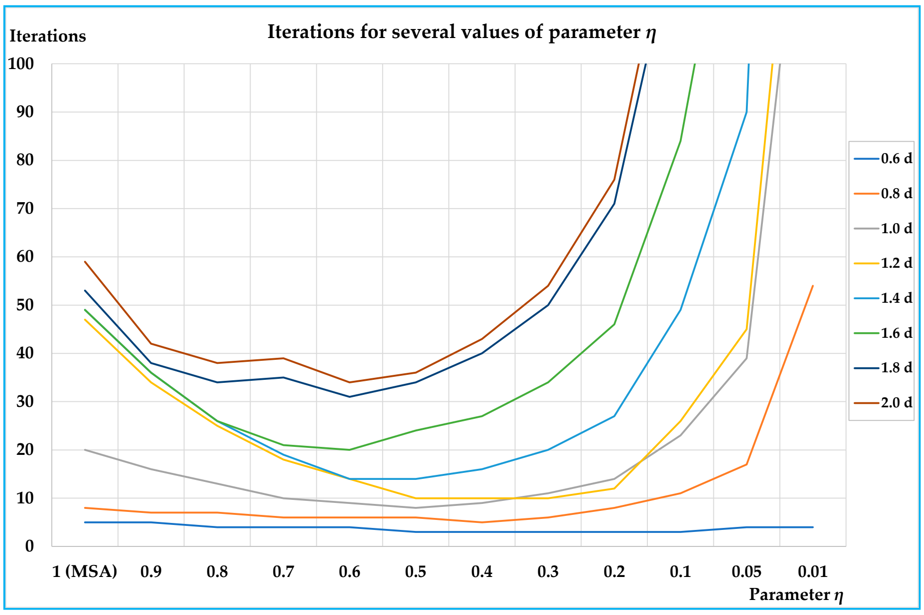

Figure 6. Results on the urban real-scale network (C_v = 0.05).

4.3. Real-Scale Rural Network

The real-scale rural test network is the regional road network of Campania, an Italian Region with more than 5.8 million inhabitants. The network, as represented in Figure 7, has 764 road links, representing about 8700 km of roads, 262 road nodes and 91 centroids.

We tested the proposed approach considering the OD matrix in the morning peak-hour, where the total trips are 165,328 veh/h; we also tested values of demand between 0.6 and 2.0 times the actual value, producing an average saturation degree between 0.4 and 1.25 (see Table 8). Moreover, in this case, we use the Multinomial Logit route choice model with the implicit enumeration of paths [27].

Table 8. Average saturation degrees on the rural real-scale network.

Matrix	0.6 *d*	0.8 *d*	1.0 *d*	1.2 *d*	1.4 *d*	1.6 *d*	1.8 *d*	2.0 *d*
Av. Sat. Degree	0.40	0.53	0.65	0.77	0.90	1.01	1.13	1.25

On this network, we tested the proposed algorithm with the same values of parameter η previously adopted, assuming a coefficient of variation equal to 0.3. Moreover, in these cases, the proposed algorithm was able to reduce the iterations for convergence with some values of η lower than 1 (MSA). The best performances were obtained with values between 0.3 and 0.6 (see Table 9); the reductions in the number of iterations reached −69% with respect to the traditional version of the MSA algorithm. In Figure 8, the results of Table 9 are illustrated.

Table 9. Results on the rural real-scale network.

	Iterations Needed for Convergence							
Matrix	**0.6 *d***	**0.8 *d***	**1.0 *d***	**1.2 *d***	**1.4 *d***	**1.6 *d***	**1.8 *d***	**2.0 *d***
η = 1(MSA)	19	26	26	29	36	54	63	70
η = 0.9	16	32	20	22	27	38	46	51
η = 0.8	14	24	18	17	21	28	33	39
η = 0.7	12	19	13	14	17	23	26	30
η = 0.6	10	15	11	**11**	**14**	18	**21**	**26**
η = 0.5	9	12	**10**	**11**	**14**	**17**	22	**26**
η = 0.4	8	11	11	13	16	21	27	35

Table 9. *Cont.*

	Iterations Needed for Convergence							
Matrix	**0.6 *d***	**0.8 *d***	**1.0 *d***	**1.2 *d***	**1.4 *d***	**1.6 *d***	**1.8 *d***	**2.0 *d***
$\eta = 0.3$	**7**	**10**	12	15	19	26	34	44
$\eta = 0.2$	9	12	16	21	27	36	48	62
$\eta = 0.1$	13	19	26	35	47	63	87	115
$\eta = 0.05$	19	31	45	61	84	117	163	218
$\eta = 0.01$	60	114	>999	258	366	525	747	>999

	Percentage Variations with Respect to MSA							
Matrix	**0.6 *d***	**0.8 *d***	**1.0 *d***	**1.2 *d***	**1.4 *d***	**1.6 *d***	**1.8 *d***	**2.0 *d***
$\eta = 1$(MSA)	-	-	-	-	-	-	-	-
$\eta = 0.9$	−16%	23%	−23%	−24%	−25%	−30%	−27%	−27%
$\eta = 0.8$	−26%	−8%	−31%	−41%	−42%	−48%	−48%	−44%
$\eta = 0.7$	−37%	−27%	−50%	−52%	−53%	−57%	−59%	−57%
$\eta = 0.6$	−47%	−42%	−58%	**−62%**	**−61%**	−67%	**−67%**	**−63%**
$\eta = 0.5$	−53%	−54%	**−62%**	**−62%**	**−61%**	**−69%**	−65%	**−63%**
$\eta = 0.4$	−58%	−58%	−58%	−55%	−56%	−61%	−57%	−50%
$\eta = 0.3$	**−63%**	**−62%**	−54%	−48%	−47%	−52%	−46%	−37%
$\eta = 0.2$	−53%	−54%	−38%	−28%	−25%	−33%	−24%	−11%
$\eta = 0.1$	−32%	−27%	0%	21%	31%	17%	38%	64%
$\eta = 0.05$	0%	19%	73%	110%	133%	117%	159%	211%
$\eta = 0.01$	216%	338%	-	790%	917%	872%	1086%	-

In bold are reported the best results.

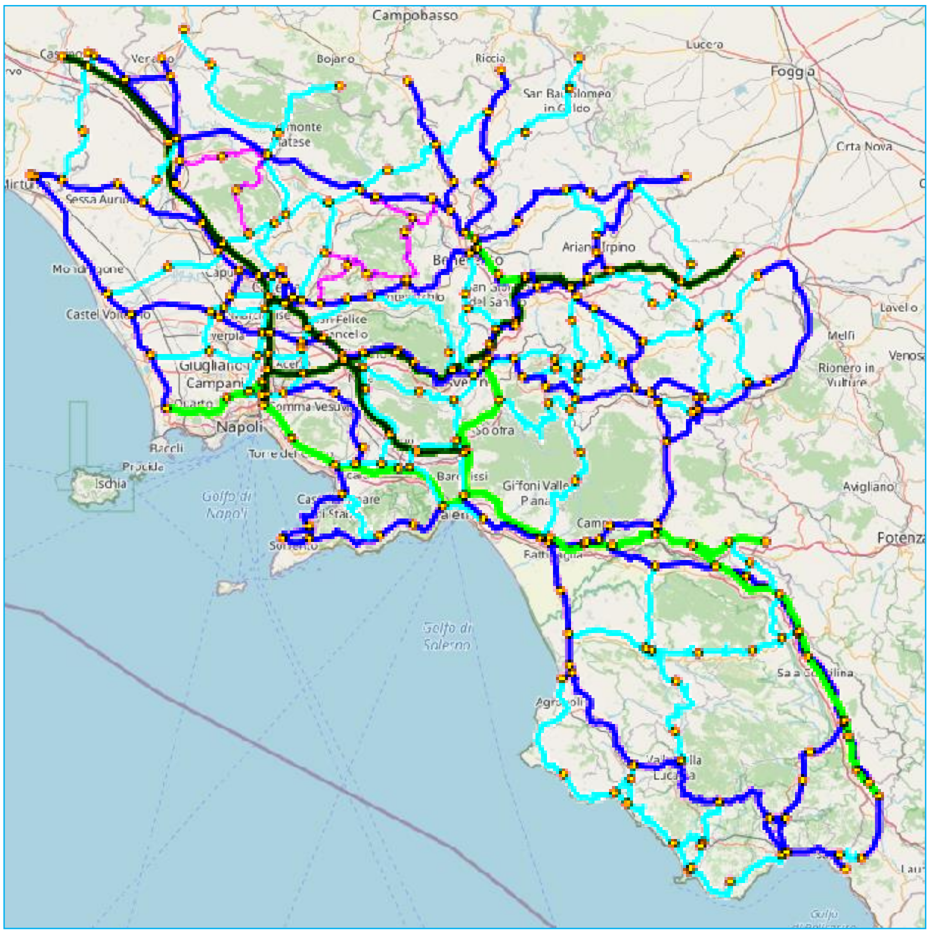

Figure 7. Rural real-scale network.

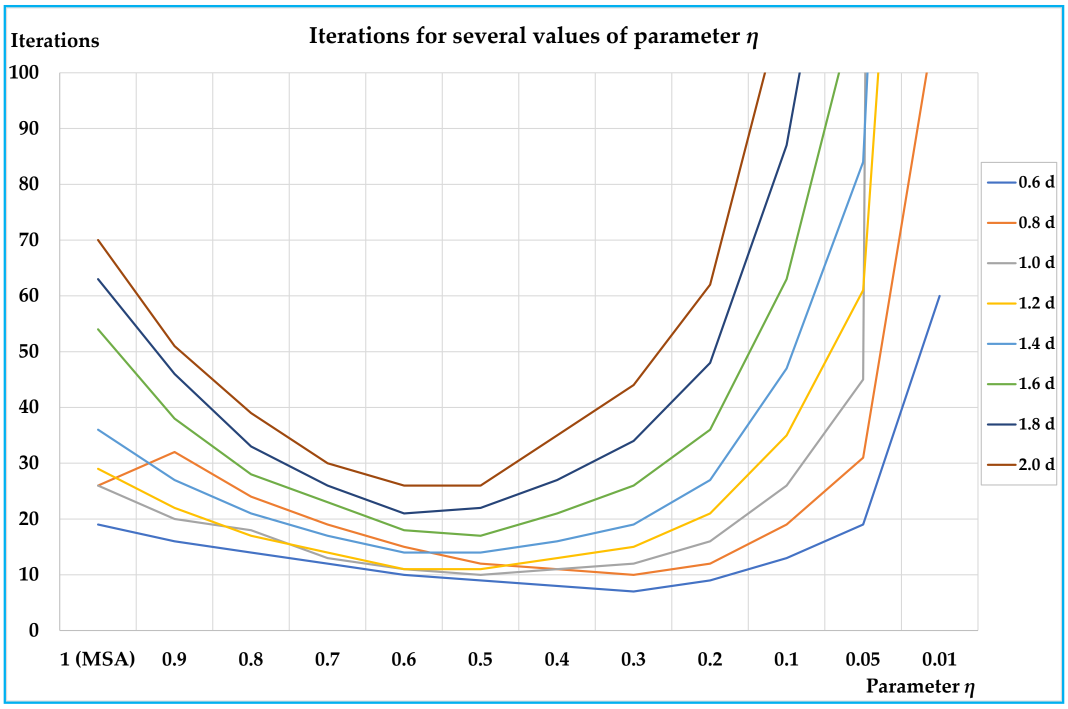

Figure 8. Results on the rural real-scale network.

4.4. Application to a Network Design Problem

The proposed algorithm was tested inside a procedure for solving Network Design Problem (NDP) on the rural real-scale network described in Section 4.3. In particular, we considered a network capacity expansion problem [28] where there were limited resources for improving the performances of a rural road network. In this problem, the current configuration of the road network is known, and no other roads can be added to it; instead, each existing road can be improved, increasing its capacity and free-flow speed. The decision variables of this problem are binary, $y_i = 0/1$, where $y_i = 1$ indicates that road i is improved in the solution, while $y_i = 0$ indicates that road i remains in the starting configuration. The details of NDP and of its model formulation are reported in Appendix C.

In order to compare the results obtained with the proposed muffled algorithm and the classic MSA, we tested the Neighbourhood Search Algorithm (NSA) with steepest descent method; it was used, for instance, in References [28,29]. A brief description of the algorithm is reported in Appendix D.

The tests were conducted under the actual demand (1.0 $\boldsymbol{d}$), and therefore, assuming the parameter $\eta = 0.5$ in the muffled proposed algorithm, since it is the best value for this demand level (see Table 9). The results of the comparison between MSA and muffled algorithm are summarised in Table 10; note that the computing time saved is about 44% (over 2.5 h). The different numbers of examined neighbourhoods and objective function values are only due to the convergence test; indeed, even if the convergence test is the same, the flows generated at each iteration of the algorithms can be slightly different and may influence the final solution. In Figure 9, the comparison among MSA and muffled algorithm is reported in terms of computing times and objective function values. These results underline the utility of the proposed algorithm inside Network Design Problem solution procedures; indeed, we have to consider also that several meta-heuristic algorithms use NSAs as subroutines (Scatter Search, Tabu Search, etc.), and then, the computing time saved should be multiplied by the number of NSAs performed for reaching the final solution.

Table 10. Comparison between algorithms.

	MSA (η = 1.0)	Muffled (η = 0.5)	% Difference
Examined solutions	6019	6223	3.39%
Examined neighbourhoods	59	61	3.39%
SNLs	72,893	39,103	−46.36%
Computing time [h]	5.83	3.25	−44.18%
Starting objective function value [€]	38,010,674,174	37,971,126,906	
Final objective function value [€]	29,363,169,248	29,305,075,507	
% variation of objective function value	−22.75%	−22.82%	

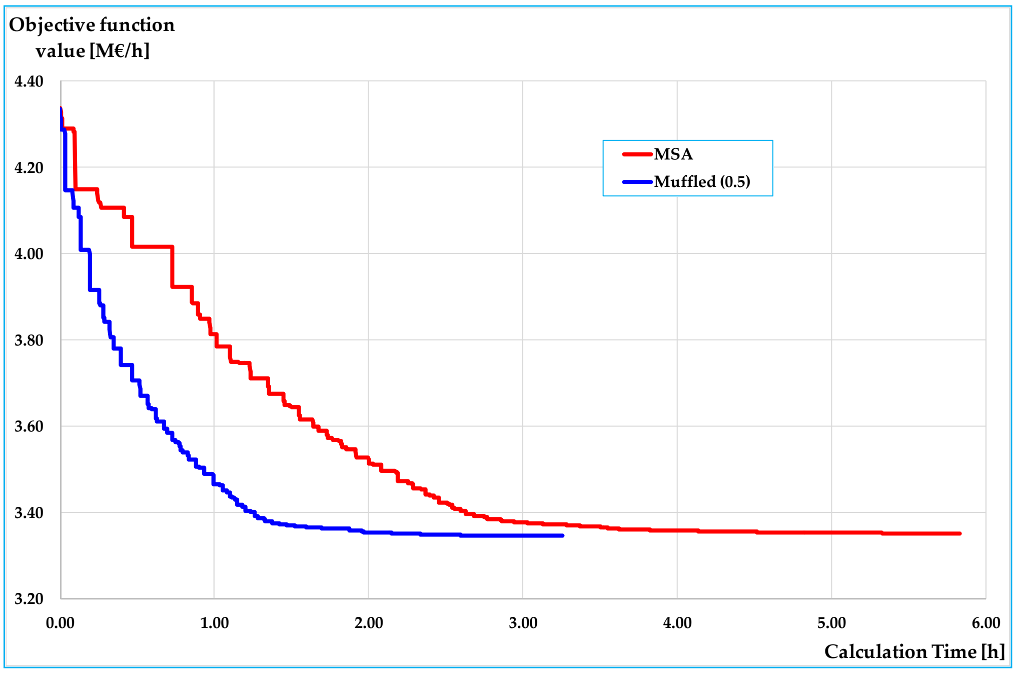

Figure 9. Comparison between algorithms in terms of computing times vs. objective function values.

5. Conclusions and Research Prospects

The results obtained on the small and on the real-scale networks show that the proposed algorithm is able to reduce the number of iterations for convergence, and hence, computing times compared with the classic MSA approach; in particular, the reductions arrive in percentage up to –79% with respect to the classic MSA for the best value of the parameter. On all networks the results are similar, showing that the benefits of the proposed algorithm are significant for medium-high congested demand levels (average saturation degrees over about 0.4); less significant, but not negligible, are the benefits for lower demand levels.

In all tests, the best values of parameter η lie between 0.3 and 0.6. An important characteristic of the proposed algorithm is that there is a parameter that can be opportunely chosen, so to optimise the performances before starting with network design procedure; moreover, examining Figures 3, 5, 6 and 8 it is evident that, especially for medium-high levels of demand, a minimum can be found. In our test, we adopt a 0.1 step, but the parameter η can assume any value between 1 (MSA) and 0 (0 excluded), and therefore, other values can be tested in order to find the one that minimises the computing times.

The reduction in computing times can be not important if the assignment procedure is performed only once or a few times: with current computing capabilities, the savings are lower than a minute on the real-scale network. By contrast, if the assignment is a subroutine of the Network Design Problem, that requires the calculation of the equilibrium traffic flows many thousands of times, the reduction in computing time of an assignment is transferred to the computing time of the network design algorithm, allowing as much as several days to be saved in the calculation. In our test on the rural road network, the proposed algorithm is able to save about 44% of computing time, equivalent to about 2.5 h for a single NSA procedure.

Future research will focus on testing other real-scale networks, testing the algorithm on multimodal networks and under the assumption of elastic demand and proposing a similar algorithm for solving the combined assignment-control problem. Moreover, testing the proposed algorithm within other real-scale network design procedures will be the subject of future studies.

Author Contributions: Conceptualisation, M.G. and L.D.; Data curation, M.B., M.G., M.M., L.D.; Formal analysis, M.G. and L.D.; Investigation, M.G. and L.D.; Methodology, M.G. and L.D.; Validation, M.B. and M.M.; Writing—original draft, M.B., M.G., M.M., L.D.; Writing—review and editing, M.B., M.G., M.M., L.D. All authors have read and agreed to the published version of the manuscript.

Funding: This research was partially funded by Federico II University of Naples (Italy), grant number 000009-ALTRO-R-2020-LD "Department research DICEA" and by University of Sannio, University Research Fund FRA_2018.

Conflicts of Interest: The authors declare no conflict of interest. The funders had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript, or in the decision to publish the results.

Appendix A. Theoretical Properties

There are three theoretical properties of a mathematical problem and its solution algorithm: existence, uniqueness and convergence. The first two properties are related to the mathematical problem and consist in proving that there is at least one solution (existence), and there is at most one solution (uniqueness). The combination of these two properties implies that the solution to the problem exists and is unique.

Once the existence and uniqueness of the problem are stated, it is necessary to adopt or develop suitable solution algorithms that are able to provide the numerical solution to the problem. Hence, it is necessary to state theoretically the convergence (i.e., the ability to provide the solution in a finite number of iterations) of the proposed algorithms.

The conditions ensuring the existence and the uniqueness of traffic assignment problem can be found in Reference [5]. As for convergence, the same paper proposed an extension of Blum's theorem [30] where it was shown that a convergent solution algorithm for solving the fixed-point problem of a function $\boldsymbol{y} = \boldsymbol{\lambda}(\boldsymbol{x})$ which satisfies some suitable conditions could be formulated by the following recursive equation:

$$\boldsymbol{x}^{k+1} = \boldsymbol{x}^k + \mu_k\left(\boldsymbol{\lambda}\left(\boldsymbol{x}^k\right) - \boldsymbol{x}^k\right) \text{ with } \boldsymbol{x}^k \in \boldsymbol{S_x} \tag{A1}$$

If the sequence $\{\mu_k\}_{k>0}$ satisfies the following conditions:

$$\sum_{k>0} \mu_k = +\infty \tag{A2}$$

$$\sum_{k>0} (\mu_k)^2 = M < +\infty \tag{A3}$$

In particular, the above-mentioned suitable conditions are:

1. $\boldsymbol{x} \in \boldsymbol{S_x}$ and $\boldsymbol{y} \in \boldsymbol{S_x}$ where $\boldsymbol{S_x}$ is a non-empty, compact and convex set;
2. a unique solution to the fixed-point problem $\boldsymbol{x}^* = \boldsymbol{\lambda}\left(\boldsymbol{x}^*\right)$;
3. a function $\varphi(\boldsymbol{x}) \geq 0 \ \forall \boldsymbol{x} \in \boldsymbol{S_x}$ where $\varphi(\boldsymbol{x})$ is continuous with first $\nabla\varphi(\boldsymbol{x})$ and second $\nabla^2\varphi(\boldsymbol{x})$ derivative continuous;

4. $\nabla^2\varphi(x)^T[\lambda(x) - x] < 0\ \forall x \in S_x, x \notin S_{\tilde{x}}$ and $\nabla^2\varphi(\tilde{x})^T[\lambda(\tilde{x}) - \tilde{x}] = 0\ \forall \tilde{x} \in S_{\tilde{x}}$ where $S_{\tilde{x}} \subseteq S_x$ and $x^* \in S_{\tilde{x}}$;
5. $|\varphi(x) - \varphi(x^*)| > 0\ \forall x \in S_x, x \notin S_{\tilde{x}}$ and $|\varphi(\tilde{x}) - \varphi(x^*)| = 0\ \forall \tilde{x} \in S_{\tilde{x}}$;
6. $x^T\nabla^2\varphi(x\prime)x = M < +\infty\ \forall x, x\prime \in S_x$.

In Reference [5], it is showed that if the sequence $\{\mu_k\}_{k>0}$ satisfies the condition:

$$\mu_k \in [0\ ,\ 1] \tag{A4}$$

then the elements of the sequence described by Equation (A1) belong to set S_x, which is convex. Moreover, by fixing $\nabla\varphi(x) = c(x) - c^*$, with $x = f$ and $c^* = c(f^*)$, all Conditions (1)–(6) are satisfied. Hence, the convergence proof is only related to checking Conditions (A2)–(A4).

Finally, Reference [5] showed that the traditional approach of MSA, i.e., $\xi(k) = k$, satisfies (A2)–(A4) conditions, since:

$$\sum_{k>0} \frac{1}{\xi(k)} = \sum_{k>0} \frac{1}{k} = +\infty \tag{A5}$$

$$\sum_{k>0} \left(\frac{1}{\xi(k)}\right)^2 = \sum_{k>0} \frac{1}{k^2} = \frac{\pi}{6} < +\infty \tag{A6}$$

In the case of the proposed generalised function, i.e., $\xi(k) = 1 + ((k - 1) \cdot \eta)$, condition (A4) is satisfied, and since for $k \to +\infty$ we obtain $\xi(k) \sim \eta \cdot k$, it is possible to show that:

$$\sum_{k>0} \frac{1}{\xi(k)} \sim \sum_{k>0} \frac{1}{\eta \cdot k} = \frac{1}{\eta} \cdot \sum_{k>0} \frac{1}{k} = +\infty \tag{A7}$$

$$\sum_{k>0} \left(\frac{1}{\xi(k)}\right)^2 \sim \sum_{k>0} \frac{1}{(\eta \cdot k)^2} = \frac{1}{\eta^2} \cdot \sum_{k>0} \frac{1}{k^2} = \frac{1}{\eta^2} \cdot \frac{\pi}{6} < +\infty \tag{A8}$$

i.e., all convergence conditions are satisfied for the proposed algorithm for each value of $\eta > 0$.

Appendix B. Algorithm Performance Comparison

Most of the MSA-based algorithms proposed in the literature for solving the traffic assignment problem are based on a modification of the weight formulation (i.e., term $1/k$ in the traditional MSA algorithm) by adopting a $1/\xi(k)$ function, that is:

$$f^k = f^{k-1} + 1/\xi(k)\ (f_{\text{SUN}}{}^k - f^{k-1}) \tag{A9}$$

The differences in formulation among these algorithms are related to the adopted $\xi(k)$ function. Indeed, the traditional MSA algorithm [2,3] assumes:

$$\xi(k) = k \tag{A10}$$

Likewise, our proposal described in Section 3 adopts:

$$\xi(k) = 1 + (k\text{-}1) \cdot \eta \tag{A11}$$

Obviously, in the case of $\eta = 1$, our proposal provides the traditional MSA formulation, that is (A11) degenerates into (A10).

In order to provide some comparisons in terms of convergence speed of the proposed algorithm with some other MSA-based algorithms, we have considered:

- the *Refresh Memory method* (RM), also indicated as *Restarting MSA method* (RMSA), proposed by Reference [18] and applied by Reference [12], whose $\xi(k)$ function may be formulated as follows:

$$\xi(k) = 1, 2, \ldots, \zeta, 2, 3, \ldots, 2\,\zeta, 4, 5, \ldots, 4\,\zeta, \ldots, x, x{+}1, \ldots, x\,\zeta, \ldots \tag{A12}$$

with:

$$x \text{ integer } x = 1, 2, 4, 8, \ldots \tag{A13}$$

where we have assumed $\zeta = 10$ (method acronym 'RM 10') as proposed by Reference [18];

- the *Polyak method* (POL) [9], whose $\xi(k)$ function is formulated as follows:

$$\xi(k) = k^{2/3} \tag{A14}$$

- the *Nagurney and Zhang method* (NAZ) [10], whose $\xi(k)$ function is formulated as follows:

$$\xi(k) = 1, 2, 2, 3, 3, 3, 4, 4, 4, 4, \ldots, x, x, \ldots, x\ (x \text{ times}), \ldots \tag{A15}$$

- the *Bar-Gera and Boyce method* (BGB) [11], whose $\xi(k)$ function is formulated as follows:

$$\xi(k) = \zeta \tag{A16}$$

where we have assumed ζ equal to 5 (method acronym 'BGB 5') and equal to 10 (method acronym 'BGB 10')

Our proposal has been classified as *muffled method* by adopting the acronym 'M η', so that 'M 1' represents the traditional MSA formulation.

The comparison among algorithm formulations has been implemented in the case of the urban network of Benevento (described in Section 4.2) by adopting a *Cv* value equal to 0.05. Numerical results are shown in Table A1 and Figure A1.

Numerical results have shown that the proposed approach (i.e., the muffled method), at least in the case under consideration (the urban network of Benevento with a *Cv* coefficient equal to 0.05) almost always provides the best results, i.e., it requires the minimum number of iterations to reach convergence.

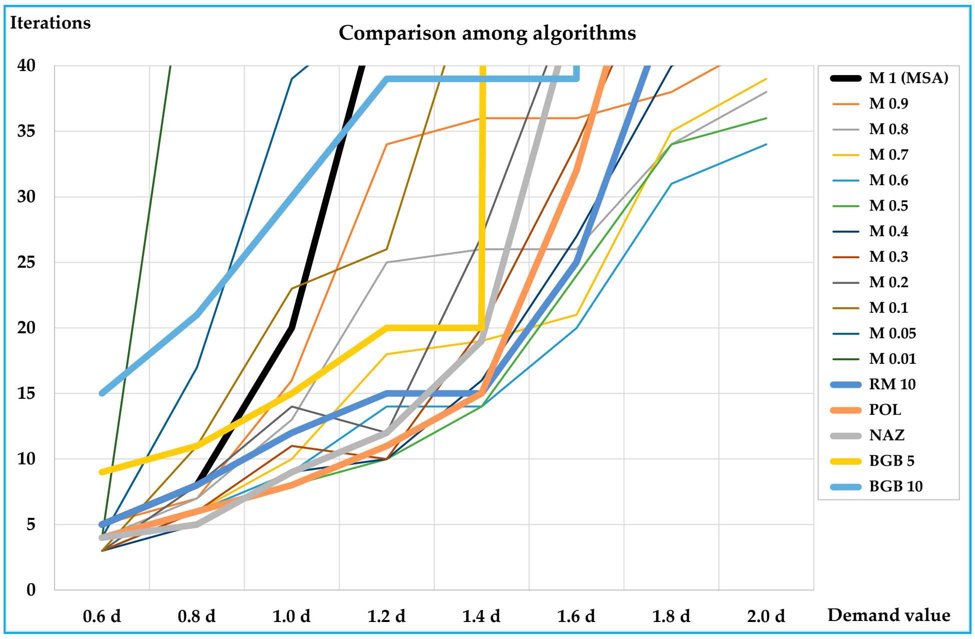

Figure A1. Comparison among MSA-based algorithms.

It is worth noting that, although the convergence can be demonstrated theoretically (as already shown in Appendix A), a theoretical proof cannot be performed in the case of the convergence speed property which represents an empirical element to be verified by means of numerous and appropriate tests.

However, although it is not possible to generalise the optimal performance of the proposed methodology in an absolute way, the results obtained give hope that at least the proposed method can be included among the ones providing optimal performances. Obviously, the confirmation of this statement can only be obtained through a large number of tests in the case of different networks both in terms of context (urban vs. rural), network dimension (urban, provincial or regional network), demand and/or congestion levels.

Table A1. Comparison among MSA-based algorithms.

	Iterations Needed for Convergence							
Matrix	**0.6 *d***	**0.8 *d***	**1.0 *d***	**1.2 *d***	**1.4 *d***	**1.6 *d***	**1.8 *d***	**2.0 *d***
M 1 (MSA)	5	8	20	47	49	49	53	59
M 0.9	5	7	16	34	36	36	38	42
M 0.8	4	7	13	25	26	26	34	38
M 0.7	4	6	10	18	19	21	35	39
M 0.6	4	6	9	14	**14**	**20**	**31**	**34**
M 0.5	**3**	6	**8**	**10**	**14**	24	34	36
M 0.4	**3**	**5**	9	**10**	16	27	40	43
M 0.3	**3**	6	11	**10**	20	34	50	54
M 0.2	**3**	8	14	12	27	46	71	76
M 0.1	**3**	11	23	26	49	84	132	141
M 0.05	4	17	39	45	90	158	251	269
M 0.01	4	54	160	186	402	724	>999	>999
RM 10	5	8	12	15	15	25	45	46
POL	4	6	8	11	15	32	57	62
NAZ	4	5	9	12	19	45	96	108
BGB 5	9	11	15	20	20	>999	>999	>999
BGB 10	15	21	30	39	39	39	>999	>999
	Percentage Variations with Respect to MSA							
Matrix	**0.6 *d***	**0.8 *d***	**1.0 *d***	**1.2 *d***	**1.4 *d***	**1.6 *d***	**1.8 *d***	**2.0 *d***
M 1 (MSA)	-	-	-	-	-	-	-	-
M 0.9	0%	−13%	−20%	−28%	−27%	−27%	−28%	−29%
M 0.8	−20%	−13%	−35%	−47%	−47%	−47%	−36%	−36%
M 0.7	−20%	−25%	−50%	−62%	−61%	−57%	−34%	−34%
M 0.6	−20%	−25%	−55%	−70%	**−71%**	**−59%**	**−42%**	**−42%**
M 0.5	**−40%**	−25%	**−60%**	**−79%**	**−71%**	−51%	−36%	−39%
M 0.4	**−40%**	**−38%**	−55%	**−79%**	−67%	−45%	−25%	−27%
M 0.3	**−40%**	−25%	−45%	**−79%**	−59%	−31%	−6%	−8%
M 0.2	**−40%**	0%	−30%	−74%	−45%	−6%	34%	29%
M 0.1	**−40%**	38%	15%	−45%	0%	71%	149%	139%
M 0.05	−20%	113%	95%	−4%	84%	222%	374%	356%
M 0.01	−20%	575%	700%	296%	720%	1378%	-	-
RM 10	0%	0%	−40%	−68%	−69%	−49%	−15%	−22%
POL	−20%	−25%	**−60%**	−77%	−69%	−35%	8%	5%
NAZ	−20%	**−38%**	−55%	−74%	−61%	−8%	81%	83%
BGB 5	80%	38%	−25%	−57%	−59%	-	-	-
BGB 10	200%	163%	50%	−17%	−20%	−20%	-	-

In bold are reported the best results.

Appendix C. Network Design Problem (NDP)

The NDP tackled in the test reported in Section 4.4 is a network capacity expansion problem where there are limited resources for improving the performances of a rural road network. The problem is the same tackled in Reference [28], slightly simplified, considering only the demand in the morning peak-hour. The problem is formulated as follows:

$$\boldsymbol{y}^{opt} = \text{Arg}_{\boldsymbol{y}} \min \left(\beta_1 \cdot (\Sigma_l\, c_l(\boldsymbol{y}, f_l^*(\boldsymbol{y})) \cdot f_l^*(\boldsymbol{y})) + \beta_2 \cdot (\Sigma_i\, y_i\, cr_i) + \beta_3 \cdot (\Sigma_l\, (Rec_{km} \cdot \Sigma_l\, f_l^*(\boldsymbol{y}) \cdot L_l))\right) \quad \text{(A17)}$$

subject to:

$$y_i = 0/1 \quad \forall\, i \in I \quad \text{(A18)}$$

$$\Sigma_\text{i}\, y_i\, cr_i \leq B \quad \text{(A19)}$$

$$\boldsymbol{f}^*(\boldsymbol{y}) = \boldsymbol{A}\, \boldsymbol{P}(\boldsymbol{A}^\text{T} \chi(\boldsymbol{f}^*, \boldsymbol{y}))\, \boldsymbol{d} \quad \text{(A20)}$$

where y_i is the decision variable related to road i; $\boldsymbol{y}$ is the decision variable vector; β_1, β_2 and β_3 are the weights of the objective function terms; $c_l(.)$ is the cost function on road link l; f_l* is the equilibrium flow on road link l; $\boldsymbol{f}$* is the equilibrium flow vector; cr_i represents the cost of the improving intervention on road i (on an hourly basis, assuming a useful life of the intervention of 20 years); Rec_{km} is the average environmental cost produced by a car travelling 1 km (€/km); L_l is the length of road link l; B is the total available budget (on an hourly basis); $\boldsymbol{A}$ is the link-path incidence matrix; $\boldsymbol{P}$ is the path choice probability matrix; $\chi(\ )$ is the cost function vector; $\boldsymbol{d}$ is the demand vector.

The objective function to be minimised, shown in (A17), is given by the sum of user costs, improvement costs and environmental costs in terms of air pollution produced by cars. Other objective function formulation could also have been used (for instance, which would have considered the impact of the design solutions on road accident phenomena or on the severity of accidents) which would obviously have produced different optimal solutions. But the purpose of this application is to show the benefits of adopting the proposed assignment algorithm in providing the same solution as traditional algorithms, but in significantly shorter calculation times, rather than providing an analysis of the best objective function to be used in network design problems.

However, the decision variable y_i is equal to 1 if the road i is improved, 0 otherwise. Equation (A18) identifies the constraint on the binary nature of decision variables, Equation (A19) is the budget constraint and Equation (A20) represents the equilibrium assignment constraint.

In order to apply the proposed model to test network described in Section 4.3, we have to identify the roads, i, that can be improved, and for each of them, the corresponding improvement costs. Each road, i, is composed by several road links, l. In this test network, we identify 102 roads, and therefore, 102 decision variables, y_i. In Figure A2, the 102 roads are reported. In Table A2 the possible interventions are summarised, in terms of performance improvements and costs.

Table A2. Intervention table.

Road	Current Configuration		Improved Configuration			Road	Current Configuration		Improved Configuration		
	v_0	*Cap*	v_0	*Cap*	cr_i (€/h)		v_0	*Cap*	v_0	*Cap*	cr_i (€/h)
1	70	4000	110	4500	27.12	52	50	2000	70	4000	7.10
2	70	4000	110	4500	30.14	53	50	2000	70	4000	4.66
3	70	4000	110	4500	36.40	54	50	2000	70	4000	3.62
4	70	4000	110	4500	12.29	55	50	2000	70	4000	9.51
5	70	4000	110	4500	31.54	56	50	2000	70	4000	4.53
6	70	4000	110	4500	30.63	57	50	2000	70	4000	3.35
7	70	4000	110	4500	68.12	58	50	2000	70	4000	11.13
8	70	4000	110	4500	51.39	59	50	2000	70	4000	6.52
9	70	4000	110	4500	28.21	60	50	2000	70	4000	11.95
10	70	4000	110	4500	37.67	61	50	2000	70	4000	8.32
11	70	4000	110	4500	35.93	62	50	2000	70	4000	3.58
12	70	4000	110	4500	21.26	63	50	2000	70	4000	4.94
13	70	4000	110	4500	19.99	64	50	2000	70	4000	0.28
14	70	4000	110	4500	13.14	65	50	2000	70	4000	2.85
15	70	4000	110	4500	20.73	66	50	2000	70	4000	4.59
16	70	4000	110	4500	16.52	67	50	2000	70	4000	11.33
17	70	4000	110	4500	4.73	68	50	2000	70	4000	7.47
18	70	4000	110	4500	12.07	69	50	2000	70	4000	3.09
19	70	4000	110	4500	27.58	70	50	2000	70	4000	1.40
20	70	4000	110	4500	14.78	71	50	2000	70	4000	5.40
21	50	2000	70	4000	10.48	72	50	2000	70	4000	5.67
22	50	2000	70	4000	5.75	73	50	2000	70	4000	2.65
23	50	2000	70	4000	11.24	74	50	2000	70	4000	0.42
24	50	2000	70	4000	15.75	75	50	2000	70	4000	2.50
25	50	2000	70	4000	14.69	76	50	2000	70	4000	0.73
26	50	2000	70	4000	1.14	77	50	2000	70	4000	6.04
27	40	1500	50	2000	5.79	78	50	2000	70	4000	0.80
28	40	1500	50	2000	6.23	79	50	2000	70	4000	2.82
29	50	2000	70	4000	10.15	80	50	2000	70	4000	3.12
30	50	2000	70	4000	4.53	81	50	2000	70	4000	2.60
31	50	2000	70	4000	16.17	82	50	2000	70	4000	5.69
32	50	2000	70	4000	4.03	83	50	2000	70	4000	11.31
33	40	1500	50	2000	4.16	84	50	2000	70	4000	11.98
34	40	1500	50	2000	3.39	85	50	2000	70	4000	2.03
35	40	1500	50	2000	8.49	86	50	2000	70	4000	3.24
36	40	1500	50	2000	2.22	87	50	2000	70	4000	1.33
37	40	1500	50	2000	4.15	88	50	2000	70	4000	6.07
38	50	2000	70	4000	7.59	89	50	2000	70	4000	2.72
39	50	2000	70	4000	7.86	90	50	2000	70	4000	4.18
40	50	2000	70	4000	9.18	91	50	2000	70	4000	4.51
41	50	2000	70	4000	3.91	92	50	2000	70	4000	2.33
42	50	2000	70	4000	10.89	93	50	2000	70	4000	8.02
43	50	2000	70	4000	3.93	94	50	2000	70	4000	7.16
44	50	2000	70	4000	6.18	95	50	2000	70	4000	7.07
45	50	2000	70	4000	9.17	96	50	2000	70	4000	1.29
46	50	2000	70	4000	1.23	97	50	2000	70	4000	1.25
47	50	2000	70	4000	7.69	98	50	2000	70	4000	10.69
48	50	2000	70	4000	8.38	99	50	2000	70	4000	1.66
49	50	2000	70	4000	6.12	100	50	2000	70	4000	10.51
50	50	2000	70	4000	5.33	101	50	2000	70	4000	9.63
51	50	2000	70	4000	8.13	102	50	2000	70	4000	2.63

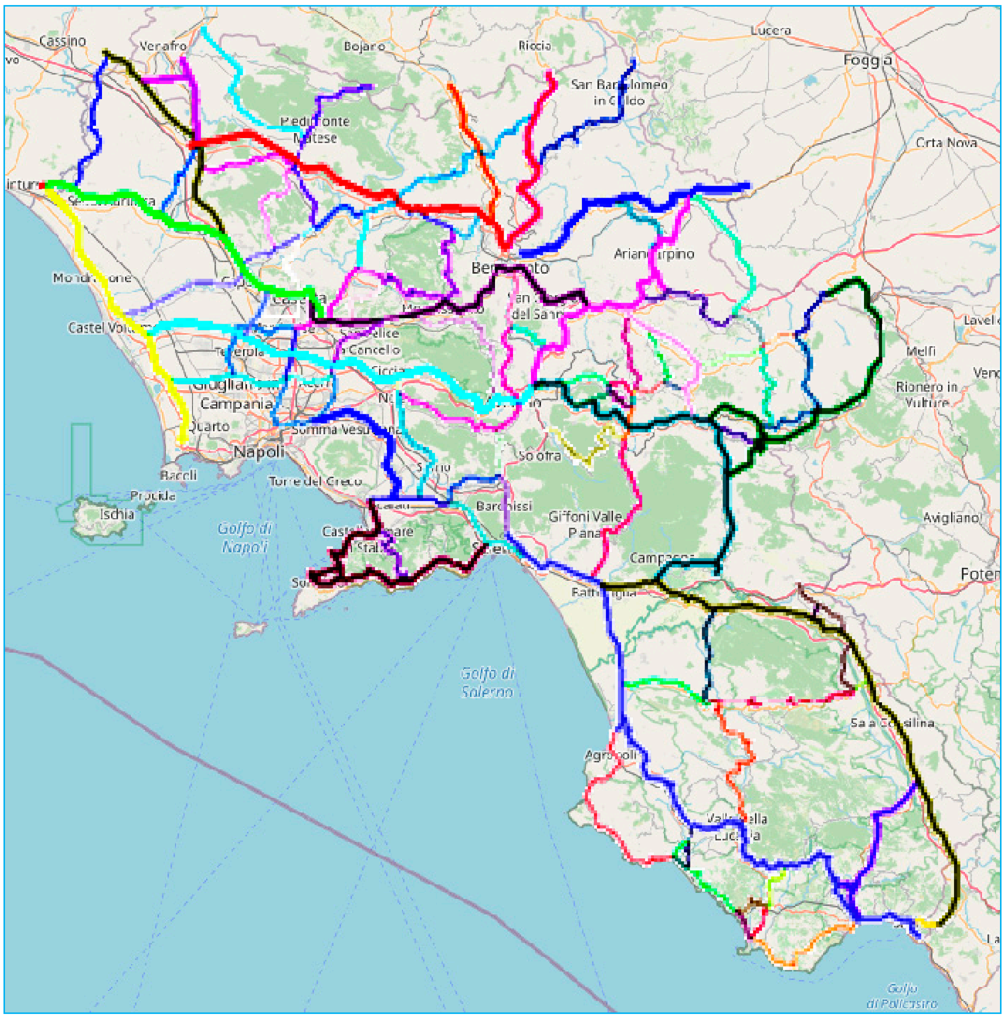

Figure A2. Identification of decision variables.

Appendix D. Network Design Solution Algorithms

In order to solve the NDP described in Appendix C, we have used a steepest descent Neighbourhood Search Algorithm (NSA); this algorithm was previously used for solving NDPs as a subroutine of meta-heuristic algorithms, among others, by References [28,29].

Given a solution, $\boldsymbol{y}$, the set of solutions that can be obtained from solution $\boldsymbol{y}$ changing only one value of a variable y_i, from 0 to 1 or vice versa, is called neighbourhood of $\boldsymbol{y}$, $N(\boldsymbol{y})$; note that other kinds of neighbourhoods can be defined, depending on the kind of problem and on the dimension of neighbourhood (for instance, changing more than one variable per time). The NSA, starting from a solution, $\boldsymbol{y}^k$, generates the following solution, $\boldsymbol{y}^{k+1}$, so that:

$$\boldsymbol{y}^{k+1} \in N(\boldsymbol{y}^k) \tag{A21}$$

and, with the steepest descent approach,

$$z(\boldsymbol{y}^{k+1}) = \text{Min}\ \{z(\boldsymbol{y});\ \forall\ \boldsymbol{y} \in N(\boldsymbol{y}^k)\} \tag{A22}$$

where $z(\boldsymbol{y})$ formally represents the objective function. The procedure then generates, at each iteration, a solution better than the previous one, choosing, among all solutions belonging to the neighbourhood, the one with the best objective function value. The procedure ends when solution $\boldsymbol{y}^k$ is a local optimum, that is when:

$$z(\boldsymbol{y}^k) \leq z(\boldsymbol{y}) \ \forall \ \boldsymbol{y} \in N(\boldsymbol{y}^k) \tag{A23}$$

This algorithm is able to find a local optimum, and therefore, is often used inside meta-heuristic algorithms that need local search subroutines. In the test reported in this paper, we use the NSA only one time, starting from the current solution: $\boldsymbol{y}^0 = \mathbf{0}$.

References

1. Wardrop, J.G. Some theoretical aspects of road traffic research. *Proc. Inst. Civ. Eng. Part II* **1952**, *1*, 325–378. [CrossRef]
2. Sheffi, Y.; Powell, W.B. A comparison of stochastic and deterministic traffic assignment over congested networks. *Transp. Res. Part B Methodol.* **1981**, *15*, 53–64. [CrossRef]
3. Sheffi, Y.; Powell, W.B. An algorithm for the equilibrium assignment problem with random link times. *Networks* **1982**, *12*, 191–207. [CrossRef]
4. Powell, W.B.; Sheffi, Y. The convergence of equilibrium algorithms with predetermined step sizes. *Transp. Sci.* **1982**, *6*, 45–55. [CrossRef]
5. Cantarella, G.E. A general fixed-point approach to multimodal multi-user equilibrium assignment with elastic demand. *Transp. Sci.* **1997**, *31*, 107–128. [CrossRef]
6. D'Acierno, L.; Montella, B.; De Lucia, F. A stochastic traffic assignment algorithm based on Ant Colony Optimisation. *Lect. Notes Comput. Sci.* **2006**, *4150*, 25–36.
7. Cantarella, G.E.; de Luca, S.; Di Gangi, M.; Di Pace, R. Stochastic equilibrium assignment with variable demand: Literature review, comparisons and research needs. *WIT Trans. Built Environ.* **2013**, *130*, 349–364.
8. Cantarella, G.E.; Cartenì, A.; de Luca, S. Stochastic equilibrium assignment with variable demand: Theoretical and implementation issues. *Eur. J. Oper. Res.* **2015**, *241*, 330–347. [CrossRef]
9. Polyak, B.T. New method of stochastic approximation type. *Automat. Rem. Contr.* **1990**, *51*, 937–946.
10. Nagurney, A.; Zhang, D. *Projected Dynamical Systems and Variational Inequalities with Applications*; Kluwer: Boston, MA, USA, 1996.
11. Bar-Gera, H.; Boyce, D. Solving a non-convex combined travel forecasting model by the method of successive averages with constant step sizes. *Transp. Res. Part B Methodol.* **2006**, *40*, 351–367. [CrossRef]
12. Cantarella, G.E.; De Luca, S.; Di Gangi, M.; Di Pace, R. Approaches for solving the stochastic equilibrium assignment with variable demand: Internal vs. external solution algorithms. *Optim. Methods Softw.* **2015**, *30*, 338–364. [CrossRef]
13. Liu, H.X.; He, X.; He, B. Method of Successive Weighted Averages (MSWA) and self-regulated averaging schemes for solving stochastic user equilibrium problem. *Netw. Spat. Econ.* **2009**, *9*, 485–503. [CrossRef]
14. Di Gangi, M.; Cantarella, G.E.; Vitetta, A. Solving stochastic frequency-based assignment to transit networks with pre-trip/en-route path choice. *EURO J. Transp. Logist.* **2019**, *8*, 661–681. [CrossRef]
15. Cantarella, G.E.; Di Febbraro, A.; Di Gangi, M.; Giannattasio, O. Solving stochastic assignment to transportation networks with TVs and AVs. *Transp. Res. Procedia* **2020**, *42*, 7–18. [CrossRef]
16. Cantarella, G.E.; Di Febbraro, A. Transportation Systems with Autonomous Vehicles: Models and algorithms for equilibrium assignment. *Transp. Res. Procedia* **2017**, *27*, 349–356. [CrossRef]
17. Mokkadem, A.; Pelletier, M. A generalization of the averaging procedure: The use of two-time-scale algorithms. *SIAM J. Control. Optim.* **2011**, *49*, 1523–1543. [CrossRef]
18. Cascetta, E.; Gallo, M.; Montella, B. Models and algorithms for the optimization of signal settings on urban networks with stochastic assignment. *Ann. Oper. Res.* **2006**, *144*, 301–328. [CrossRef]
19. Magnanti, T.; Wong, R. Network design and transportation planning: Models and algorithms. *Transp. Sci.* **1984**, *18*, 181–197. [CrossRef]
20. Yang, H.; Bell, M.G.H. Models and algorithms for Road Network Design: A review and some new developments. *Transp. Rev.* **1998**, *18*, 257–278. [CrossRef]
21. Feremans, C.; Labbé, M.; Laporte, G. Generalized network design problems. *Eur. J. Oper. Res.* **2003**, *148*, 1–13. [CrossRef]

22. Farahani, R.Z.; Miandoabchi, E.; Szeto, W.Y.; Rashidi, H. A review of urban transportation network design problems. *Eur. J. Oper. Res.* **2013**, *229*, 281–302. [CrossRef]
23. Zilioniene, D.; D'Acierno, L.; Botte, M.; Gallo, M. A general methodology for reducing computing times of road network design algorithms. *Int. J. Supply Oper. Manag.* **2019**, *6*, 126–141.
24. Cascetta, E. *Transportation Systems Analysis: Models and Applications*; Springer: New York, NY, USA, 2009.
25. Cantarella, G.E.; Watling, D.; de Luca, S.; Di Pace, R. *Dynamics and Stochasticity in Transportation Systems: Tools for Transportation Network Modeling*; Elsevier: Amsterdam, The Netherlands, 2019.
26. Bureau of Public Roads. *Traffic Assignment Manual*; U.S. Department of Commerce, Urban Planning Division: Washington, DC, USA, 1964.
27. Dial, R.B. A probabilistic multi-path traffic assignment model which obviates path enumeration. *Transp. Res.* **1971**, *5*, 83–111. [CrossRef]
28. Gallo, M.; D'Acierno, L.; Montella, B. A meta-heuristic algorithm for solving the road network design problem in regional contexts. *Procedia Soc. Behav. Sci.* **2012**, *54*, 84–95. [CrossRef]
29. Gallo, M.; D'Acierno, L.; Montella, B. A meta-heuristic approach for solving the Urban Network Design Problem. *Eur. J. Oper. Res.* **2010**, *201*, 144–157. [CrossRef]
30. Blum, J.R. Multidimensional stochastic approximation methods. *Ann. Math. Stat.* **1954**, *25*, 737–744. [CrossRef]