WORK ZONE COST ANALYSIS

UNDER DETOUR SETTINGS

A Thesis

Presented to the Faculty of the Graduate School

of Cornell University

in Partial Fulfillment of the Requirements of the Degree of

Master of Science

by

Kai Huang

February 2016

ABSTRACT

A new approach of work zone delay cost analysis is developed to understand the implications of detours. This method, unlike any of the previous methods, is based on the impacts within the transportation network and let users find the shortest paths which will eventually come to equilibrium. The total work zone cost consists of total delay cost, work zone set-up cost and accident cost. The analysis is based on an illustrative transportation network, the Sioux-Falls network. The total delay cost is calculated using User Equilibrium assignment to distribute all the traffic. To solve for User Equilibrium, the Frank-Wolfe algorithm is implemented via Matlab. The total delay cost is then converted to a monetary value using value of time (VOT). The work zone set-up cost and accident cost relate to number of work zone segments and total length of work zone. This thesis illustrates the importance of considering the opportunity for detours when considering the design of work zones for road construction.

BIOGRAPHICAL SKETCH

Kai Huang completed her bachelor's degree in Highway and Bridge Engineering in Southeast University, Nanjing, China.

She is a graduate student with major in Transportation Systems Engineering in Civil and Environmental Engineering. During the two years in pursuit of master's degree, she took a lot of courses in her major field as well as other fields, such as Systems Engineering, Operations Research, and MBA etc.

Outside academics, she is very involved in University activities. She worked for Cornell Dining at Kennedy Hall for one semester. She was also the lounge manager of CEE graduate students association.

ACKNOWLEDGEMENTS

This paper is made possible through the support of everyone: professors at Cornell, friends and family.

First and foremost, I would like to express my sincere gratitude to my advisor Prof. Linda Nozick. We had numerous conversations about the idea, approach and outcome for the research. She has always been patient and willing to give me a lot of guidance and support through the whole research process. She also opened the door to transportation network modeling for me as the professor of the very first class I took at Cornell.

Secondly, I want to thank my minor advisor Prof. Ricardo Daziano, and the other CEE and ORIE professors. I can still remember those classes I took at Cornell and I have to say I enjoyed all of them. Thanks to these knowledgeable and passionate professors who gave informative and illuminating lectures which helped me shape this research idea.

Thirdly, I want to thank Jun Zhou for his moral and intellectual support, for giving me suggestions and inspirations. He encouraged me when I was desperate and was more than happy when I had pieces of the results from Matlab.

Last but not least, thanks to my parents back in China for their encouragement and financial support during the two years program. I am grateful for their endless care and love.

This thesis would not have been possible without all of their help.

TABLE OF CONTENTS

INTRODUCTION

A work zone is defined in the Highway Capacity Manual (HCM) as “an area of highway in which maintenance and construction operations are taking place that impinge on the number of lanes available to moving traffic or affect the operational characteristics of traffic flowing through the area”[1].

With the increasing demand for traveling, shipping or other purposes of transportation, reconstruction or redevelopment of existing roadways occurs frequently on highways and freeways. Work zones have a huge impact on road users, including reduced traffic lanes, reduced speed, increased queues, additional delay costs, noise and increased possibility of accidents.

Work zone cost is comprised of agency cost, user delay and potential accident costs. Longer work zones result in greater user costs but lower agency cost. In reality, under many circumstances, detour is provided as an alternative for road users when road construction is in progress. With the option of detour, the queuing and delay condition is different from a work zone without a detour.

This paper develops a simplified means to determine optimal work zone division with the option of detour to minimize the total work zone cost for local freeways.

Literature Review

Previous research on work zone cost analysis is very limited. Martinelli and Xu(1996) proposed mathematical approach to model the performance of vehicles in construction work zone and estimated the traffic delay at certain circumstances. In this article, the emphasis is also laid on comparing two factors of work zone delay, speed-reduction and congestion delay, and analyzing how the different work zone set-ups will affect traffic. In the last, it is concluded that the ideal work zone pattern is determined by several aspects including, road user cost, accident rates, and operation cost of traffic control facilities.

Chien and Schonfeld (2001) presented a simplified and useful model for estimating the delay cost using the annual average daily traffic (AADT) and finding the optimum work zone segment length in a four-lane freeway with one lane closure.

Fazil T. Najafi and Roberto Soares (2001) considered three major factors (investor, road user, construction contractor) that influence the design of a work zone. The five most frequently used approaches, manual on uniform traffic control devices (MUTCD), the Highway Capacity Manual (HCM), computer programs, delay models, and work zone user cost models, are discussed to establish the work zone user cost.

Jiang and Adeli (2003) developed a model to minimize the total work zone cost. Using average hourly traffic data and based on Boltzmann-simulated annealing neural network, the model finds the optimal work zone segment length and best starting time of construction. It also considers factors like number of lane closures and darkness.

Zhang, Shen, Nie and Ma (2008) developed a computerized program, NetZone, to estimate some of the major parameter concerning to traffic performance. With the help of NetZone, the

user is capable of conducting the analysis of roadway congestion level, traffic queue length, traffic delay.

Antonoius, Psarianos and Foll (2011) established a simulation based approach to set up a road construction work zone. 2+1 and 1+1 work zone configuration were adapted and compared under different traffic density. Due to lack of data, this simulation is based on the assumption that all the road segments are in ideal condition.

Wang and Goodrum (2005) developed road user cost (RUC) table to obtain road user cost in Kentucky. In this paper, the authors divided the research area into four county groups based on personal hourly income. Taking value of time (VOT) and other factors like annual average daily traffic (AADT) and length of work zone into consideration, RUC tables are estimated. Then, With the help of RUC table, user can quickly analyze and select a construction method based on multiple variables (highway type, numbers of active traffic lanes, physical features of site area, annual average daily traffic and construction time framing to perform) in preliminary design phase.

It is important to notice that none of these paper focuses on the integration of the possibility of making use of a detour route into the analysis. This thesis begins to address this possibility.

Model Formulation and Solution Procedure

Work zone cost is comprised of two components: agency cost and user cost. Agency cost refers to work zone set-up cost and maintenance cost and can be simplified into the following formula:

$$C_a = C_1 + C_2 * l \quad (1)$$

C_a: Work zone agency cost

C_1: Fixed set-up and maintenance cost

C_2: Set-up and maintenance cost per mile

l: Length of work zone segment

User cost mainly consists of delay cost and accident costs. Delay cost is the monetary value of the extra time that road users have to spend because of the construction. Due to the full road closure setting in this research, the delay cost will be related to the detour adopted. Accident cost associated with work zones or detours refers to the change of accident rates in the presence of work zones.

To simplify and only consider the main factors mentioned above, we have the functions below:

$$WZC = AC + UC \quad (2)$$

$$WZC = SMC + UDC + WZAC \quad (3)$$

WZC: Work zone costs

AC: Agency costs

UC: User costs

SMC: Set-up and maintenance costs

UDC: User delay cost

WZAC: Work zone related accident costs

The model takes into account these factors:

(1) Number of maximum possible work zone segments(n) from user's input

(2) Length of detour(*l'*)

(3) Work zone set-up and maintenance time(t)

Several assumptions which are made to simplify the problem are listed below:

(1) The work zone will be under full road closure and traffic will be diverted onto other links;

(2) The user delay cost may be represented by an average cost per vehicle hour;

(3) The capacity and length of all arcs;

(4) Users are all homogeneous and rational decision makers who are fully aware of the information of the network.

Prior to optimizing the work zone, the base case user costs are computed where there are no work zones. This is simply as a point of comparison. Then different configurations for the work zone are analyzed as follows.

(1) A single work zone. That is a work zone from an origin node to destination node as a whole segment.

(2) A work zone which divides the work to be done into two segments. The first portion of work zone will be fully closed first for construction and in the meantime users are able to use the

rest of mainline. After the first segment of work has been completed, the users will not have access to the second segment until it has been finished.

…

(*n*) Work zone will be divided into *n* segments

The *n* segments will rotate for construction similarly as the work flow above. Users will only be able to use the roads that are not under construction.

The user equilibrium assignment is based on Wardrop's first principle, which states that no driver can unilaterally reduce his/her travel costs by shifting to another route. The following objective function and conditions are equal to user equilibrium assignment[2]:

$$\min z(x) = \sum_a \int_0^{x_a} t_a(\omega) d\omega$$

subject to

$$x_a = \sum_{r,s,k} f_k^{rs} \delta_{a,k}^{r,s} \quad \forall a \qquad (4)$$

$$\sum_a f_k^{r,s} = q_{rs} \quad \forall r, s$$

$$f_k^{r,s} \geq 0$$

The travel time under baseline case and work zone cases can be derived from the BPR Formula (Bureau of Public Roads).

$$t(x) = t_f \left[1 + \alpha \left(\frac{x}{c}\right)^{\beta}\right] \qquad (5)$$

t = Congested link travel time

t_f = Link free flow travel time

x = Link volume

c = Link capacity

α, β = Calibration parameter

The equilibrium is to be found by using the Frank Wolfe algorithm. The steps are as follows:

Step 0: Initialization. Perform all-or-nothing assignment based on free flow travel time. This yields $\{x_a^1\}$. Set n=1.

Step 1: Update. Set

$$t_a^n = t_a(x_a^n) \quad (6)$$

Step 2: Direction Finding. Perform all-or-nothing assignment based on $\{t_a^n\}$. This yields a set of auxiliary flows $\{y_a^n\}$.

Step 3: Line Search. Find λ_n that solves

$$\min_{0 \le \lambda_n \le 1} \sum_a \int_0^{x_a^n + \lambda_n (y_a - x_a)} t_a(\omega) d\omega \quad (7)$$

Step 4: Move. Set $x_a^{n+1} = x_a^n + \lambda_n (y_a - x_a) \quad \forall a$ (8)

Step 5: Convergence test. If a stopping criterion is met, stop; else: n=n+1 and go to step 1.

$$\frac{\sum_a (x_a^{n+1} - x_a^n)^2}{\sum_a x_a^n} \le k' \quad (9)$$

After finding UE flow for each scenario, total travel time will be derived from adding up flow on each arc times travel time on each arc for all arcs in the network for the whole construction period.

To determine user delay cost, we also need monetary value of time(VOT), which is a factor varying among people with different traveling purposes and wages, to convert the total delay time into monetary value.

Illustrative Example

User delay cost

The network used is Sioux-falls network, shown in Figure 1[6]. This network was originally used to test the algorithm by Larry LeBlanc to solve the roadway network equilibrium

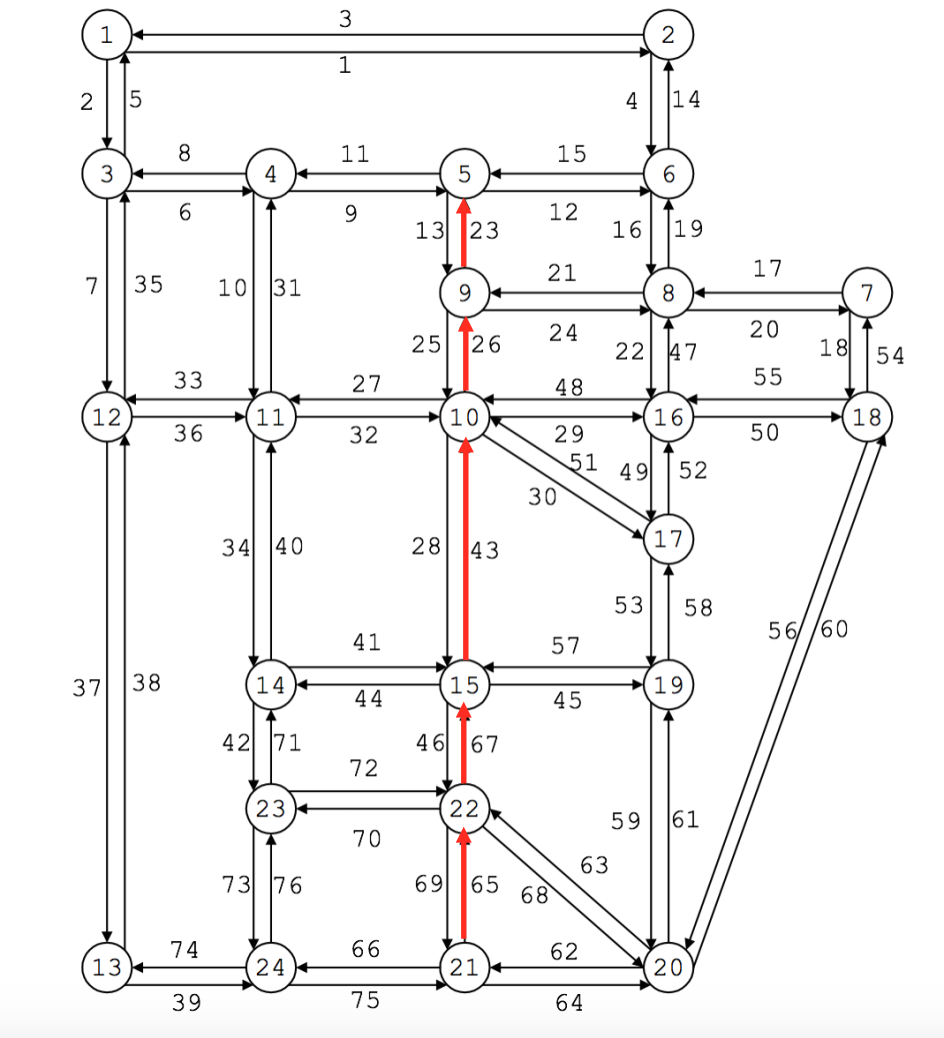


FIGURE 1 SIOUX-FALLS NETWORK

assignment problem. Sioux Falls is the largest city in the U.S. state of South Dakota. It represents Sioux-falls' traffic network but is a simplified network. It has 76 arcs and 24 nodes, all of which are origins and destinations.

Original lengths and capacity data of Sioux-Falls are shown in Table 6-a, 6-b, 6-c, 6-d in Appendices. In the model, both have been divided by 10 so as to fit our work zone. Links 21-22, 22-15, 15-10, 10-9, and 9-5 are assumed to have construction work and are shown in red in Figure 1. Free flow travel time equals to link lengths divided by average speed. In the BPR formula, values of α=0.15 and β=4 are assumed.

Four scenarios will be analyzed based on User Equilibrium assignment.

(1) No work zone

This is the baseline of analysis where traffic will route from node 21 to node 5 without capacity reduction on the mainline.

(2) Work zone from 21 to 5 as a whole segment

This is the case where the whole work zone will be under construction and fully closed until the construction work is completed.

(3) Work zone from 21 to 10 and 10 to 5 as two segments

The first portion of work zone will be fully closed first for construction and in the mean time users are able to use the rest of mainline. After the first segment of work has been completed, the users will not have access to the second segment until it has been finished

(4) Work zone from 12 to 15, 15 to 10 and 10 to 5 as three segments

The three segments will rotate for construction similarly as the work flow above. Users will only be able to use the roads that are not under construction.

Assume the total construction time is comprised of the fixed set-up time 3 days per segment, and additional construction time, 20 days/mile. Therefore, construction time for three work zone scenarios can be estimated.

Path 21-22-15-10-9-5 is assumed to be a freeway segment with an average daily 1200 traffic from start to end. The rest of network arcs are local highways with free flow speed of 30 mph. In the original network without work zone, the users will choose the route to minimize his/her travel time. By choosing those routes, the travel time will update according to BPR Formula and ultimately the traffic pattern will come to user equilibrium.

For the three work zone set-up cases, assume links 21- 22- 15- 10- 9- 5 to be the work zone by assigning the link lengths to 99999 so that users will avoid those links to minimize their travel times. The total travel time of 1200 users within an hour in each of the three cases will be compared with the baseline analysis and work zone delay per hour will be derived. The total work zone delay will be calculated by the following equation:

$$UDC = T_C * AUD * MV \quad (10)$$

UDC = User delay cost

Tc = Construction time (in hours)

AUD = Average user delay per hour

MV = Monetary value of user delay of one hour

Work zone set-up and maintenance cost

Assume that work zone set-up and maintenance cost consists of fixed cost for each segment independent of work zone length and flexible cost proportional to the segment length.

$$C_{set-up} = C_f + C_l * l \quad (11)$$

Assume that Cf=$4,000/zone and Cl=20,000/mile.

Work zone accident cost

Since work zone increase the likelihood of accidents, the accident cost is part of work zone cost as well. The accident cost per work zone incurred by the traffic flow passing through the work zone is determined from the number of accidents, N_a , per 100 million vehicle hours, multiplied by the product of the increased delay, t_d , and the average cost per accident, c_a[3].

$$AC = N_a * t_d * c_a/100{,}000{,}000 \quad (12)$$

Since our work zone has been fully closed, we will apply this computation to the links that accommodate traffic that has had to adopt a detour. To perform the computation we assume that Na=40 and c_a=142,000[5].

Results and Discussion

Construction Time

As mentioned previously, the total construction time is assumed to be the 5-days fixed set-up time per segment, plus the 40 days per mile cost depending on the work zone length. According to this assumption, the total construction time for three work zone scenarios are 5+40*1.9=81 days (1944 hr), 5*2+40*1.9=86 days (2064 hr), and 5*3+40*1.9=91 days(2184 hr) respectively.

User delay cost

The original network before construction is run under the assumption that an average of 1,200 trips are going from 21- 22- 15- 10- 9- 5 per hour on a daily basis. The total travel time per hour for all traffic is estimated by the summation of each arc volume multiplied by flow on that arc when UE has been reached. The original total travel time for all users is 82.95 veh-hr for an hour. The UE flow of the original network is shown in Figure 2 below.

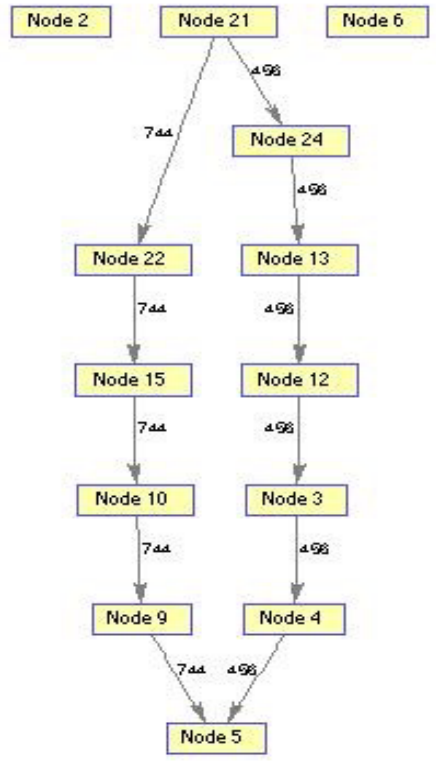


FIGURE 2 ORIGINAL NETWORK FLOW

In the first work zone traffic control plan, the whole corridor is closed and under construction. The total travel time on all arcs is 91.67 veh-hr. Compared to no-work-zone case, the total delay per hour is 91.67-82.95=8.72 veh-hr. The UE flow for this work zone plan is shown in Figure 3.

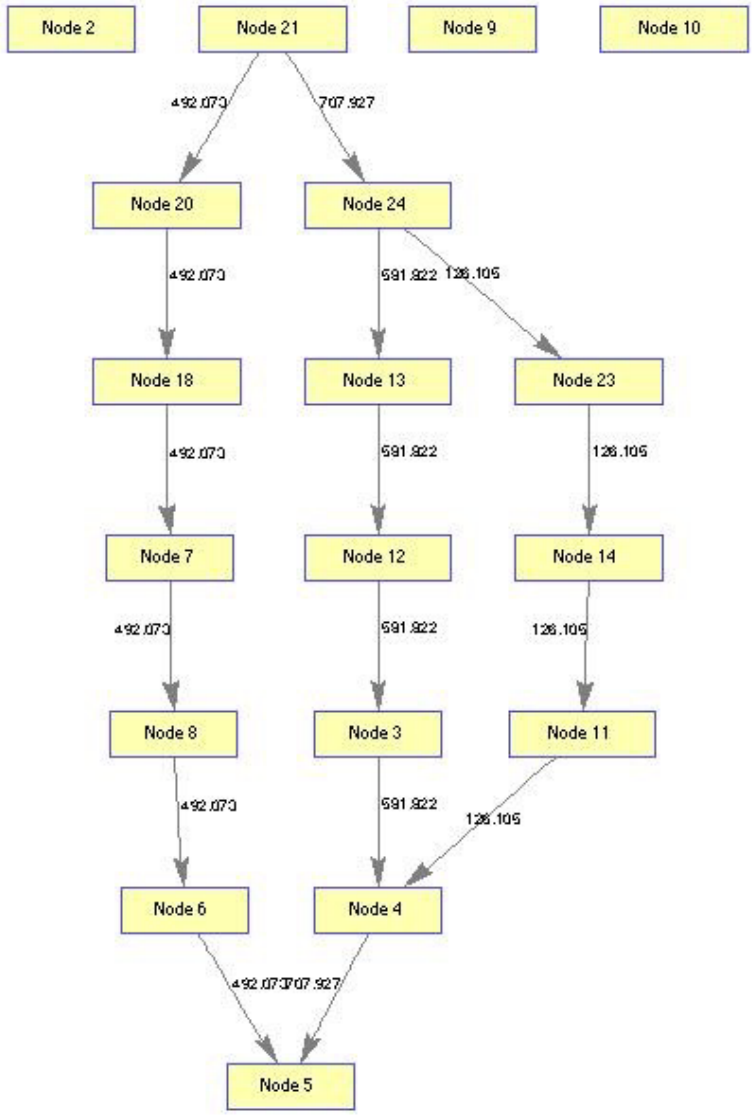


FIGURE 3 ENTIRE SEGMENT CLOSURE FLOW

The second work zone traffic control plan is to divide the work zone into two segments. In the first period, 21-22-15-10 is closed and users will need to detour to get from node 21 to node 10. In the second period, 21-22-15-10 will be open for use again but 10-9-5 is closed. UE condition is found in both phases and weighted average delay is calculated based on their lengths respectively. Figure 4 and Figure 5 represent the flow under User Equilibrium condition. In this scenario, the original travel time is the same as no work zone travel time, 82.95 veh-hr. The total

delay within an hour of first work zone period is 92.19 veh-hr and that of the second period is 91.14 veh-hr. The delay is 9.24 and 8.19 veh-hr compared to no work zone scenario.

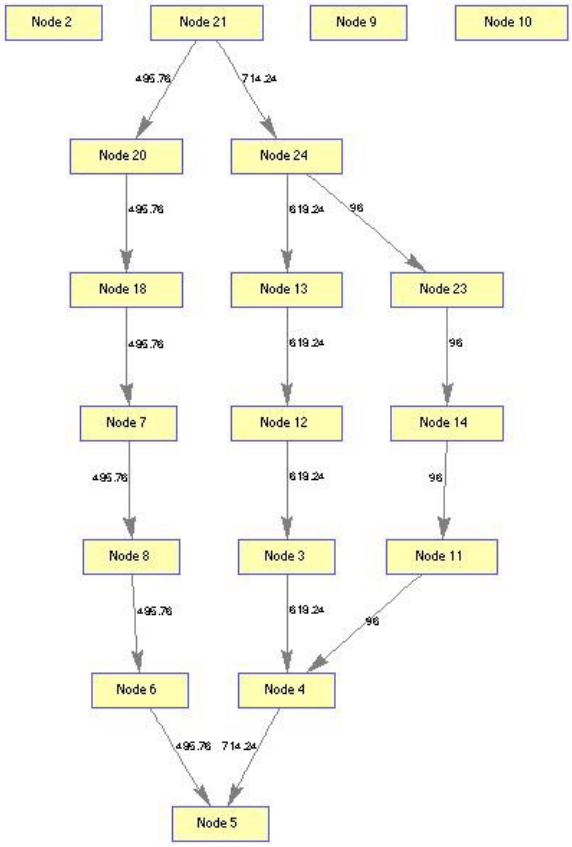


FIGURE 4 TWO SEGMENTS FLOW 1

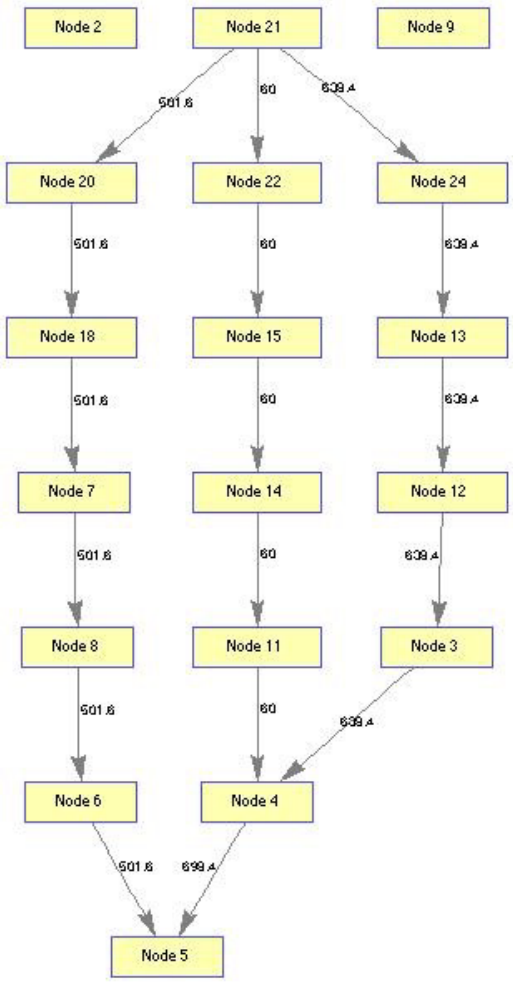


FIGURE 5 TWO SEGMENTS FLOW 2

The third work zone traffic control plan is to divide the work zone into three segments. Route 21-22-15 will be in work zone first and users are allowed to use the rest of path until it's finished. Figure 6 shows the User Equilibrium flow under this condition.

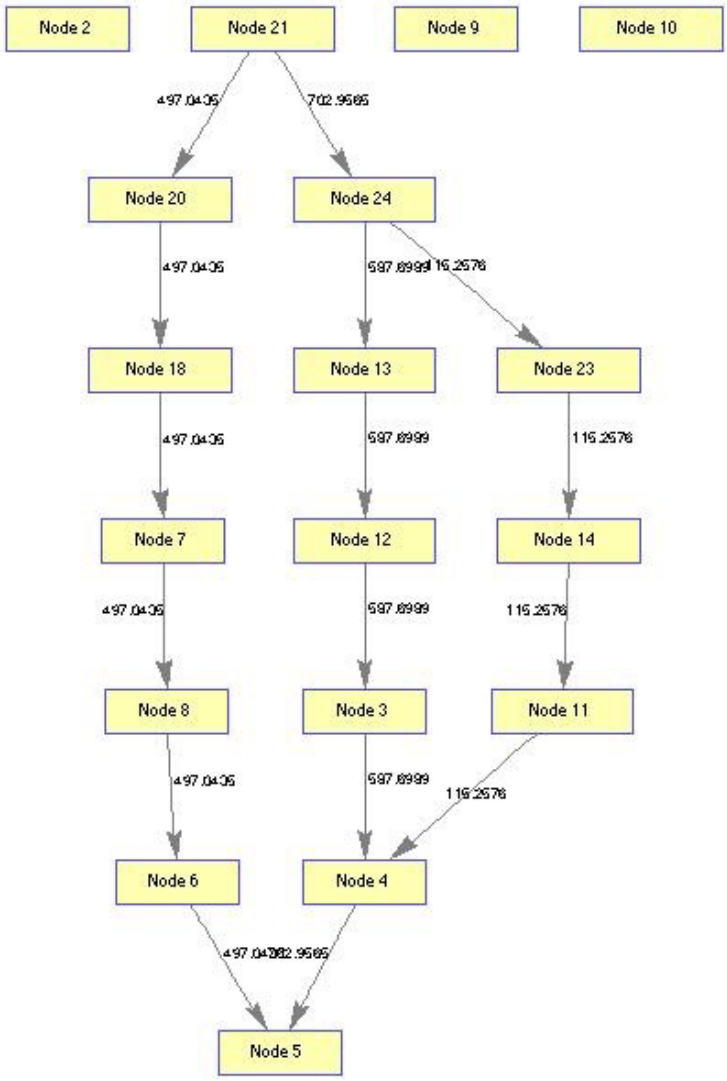


FIGURE 6 THREE SEGMENTS FLOW 1

Path 15-10 will be then under construction and finally 10-9-5. Both conditions have same UE flow. Figure 7 shows the equilibrium flow when 15-10 or 10-9-5 is closed. The travel time from Node 21 to 5 after they each come to equilibrium are 93.23 veh-hr, 89.91 veh-hr, and 89.91 veh-hr. The delay is 10.28, 6.96 and 6.96 veh-hr compared to no work zone scenario in an hour.

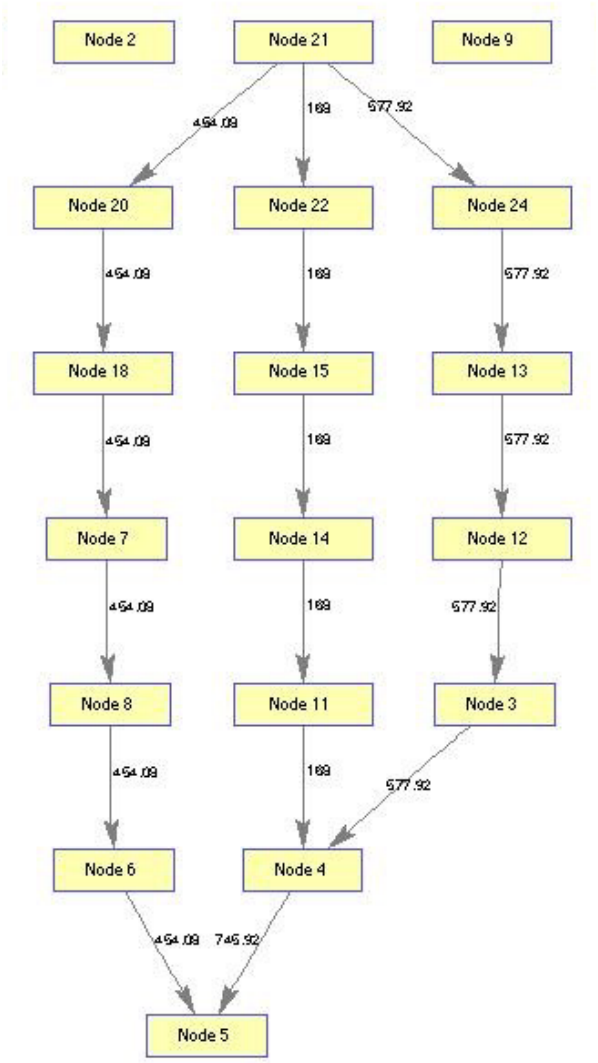


FIGURE 7 THREE SEGMENTS FLOW 2

The total vehicle delay time per hour for the three situations is given in Table 1.

TABLE 1 TOTAL DELAY TIME

	No work zone	One segment	Two segments	Three segments
Travel time(veh-hr)	83	92	92/91	84.1/90/90
Delay per hour(veh-hr)	0	9	9/8	1.1/7/7
Total construction time(hr)	0	1,944	2,064	2,184
Total Delay	0	16,952	18,139	11,748

To convert the total delay to monetary value, we need to know the hourly dollar value of vehicle delay time. The monetary value of travel time is based on the concept that time spent traveling otherwise would have been spent productively, whether for work or recreation [3]. The United States Department of Transportation (USDOT) Office of the Secretary of Transportation (OST) provides guidelines and procedures for calculating the value of travel time saved or lost by road users [5].

To simplify we only consider the dollar value of local personal travel. We do not consider business travel or truck travel scenarios. We also use typical values from the surveys.

The hourly dollar value of road users' personal travel time is estimated based on their wages. According to FHWA, the steps involved in monetizing the personal travel delay time are as follows [3].

Step 1. Determine the percentage of passenger cars on personal travel. This depends on the type of travel: local or intercity. According to the National Household Transportation Survey (NHTS) and the Nationwide Personal Transportation Survey (NPTS) over the past 2 decades, the proportion of passenger cars on personal travel is approximately 90%.

Step 2. Establish the average vehicle occupancy (AVO) of passenger cars. According to NHTS estimates of AVO values, the average AVO for personal travel was 1.67 in 2009.

Step 3. Calculate per hour monetary value of travel time for a person on personal travel. The dollar value of personal travel time (per person–hr) is estimated using the median annual income for all U.S. households reported by the U.S. Census Bureau, in accordance with the OST guidelines.

The total work hours in a year = 8 hr/day* 5 day/wk* 52 wk =2080 hr. For local personal travel, hourly value of personal travel time per person = 50% of median annual household income ÷ 2080 hours. Median annual income for all U.S. households = $52,047 (for 2013).[5]

Hourly value of personal travel time = 0.5* $52,047/2080 = $12.60/person -hr.

Step 4. Compute hourly monetary value of travel time for a vehicle on personal travel. The dollar value of personal travel time for all occupants in a vehicle, in terms of dollar/vehicle-hr, is computed by multiplying the dollar value of hourly travel time per person (Step 3) with an appropriate average vehicle occupancy (AVO) factor (Step 2).

For local personal travel, hourly value of a person's travel time in a vehicle= $12.60/ person-hr. Average vehicle occupancy = 1.67 persons per vehicle. Hourly travel time value of all occupants in a vehicle or the hourly value of a vehicle on personal travel = $12.60* 1.67 = $21.04/vehicle-hr.

Step 5. Compute travel delay costs for passenger cars on personal travel. Multiply the hourly dollar value of vehicle delay time with the delay time of passenger cars on personal travel. For local or personal travel, travel delay costs for passenger cars on personal travel = Total delay time for passenger cars on personal travel * hourly dollar value of vehicle delay time.

In our case, the total delay row from Table 1 is multiplied by hourly dollar value 21.04 and

then multiplied by the total construction hours, which yields Table 2 below.

TABLE 2 TOTAL DELAY COST

	No work zone	One segment	Two segments	Three segments
Total delay cost	0	$356,663	$381,644	$247,178

We can see that the work zone delay cost is not monotonous. The users are more flexible in choosing route as the work zone is divided into more segments, which will cut down the users' travel time. However, as the number of segments increase, the construction will take longer because the set-up time for each work zone will accumulate. The result shows the third option has the lowest work zone delay cost among the three.

Work zone set-up and maintenance cost

The work zone set-up cost consists of fixed cost for each segment independent of work zone length and another cost proportional to the segment length.

$$C_{set\text{-}up}=C_f+C_l*L \quad (13)$$

Assume C_f=$5,000/segment and C_l=$20,000/mile.

The second portion is the same in three cases because they have the same total work zone length, $L=2$ mile. The result is tabulated in Table 3.

TABLE 3 WORK ZONE SET-UP AND MAINTENANCE COST

	No work zone	One segment	Two segments	Three segments
Fixed Cost	0	$5,000	$10,000	$15,000
Flexible Cost	0	$40,000	$40,000	$40,000
Total Set-up Cost	0	$45,000	$50,000	$55,000

Work zone accident cost

Using the total delay from Table 1, and assumption that Na=40 and Ca=142,000[5], we are able to estimate the accident cost for each scenario in Table 4.

TABLE 4 ACCIDENT COST

	No work zone	One segment	Two segments	Three segments
Total delay	0	16,952	18,139	11,748
Accident Cost	0	$963	$1,030	$977

Total work zone cost

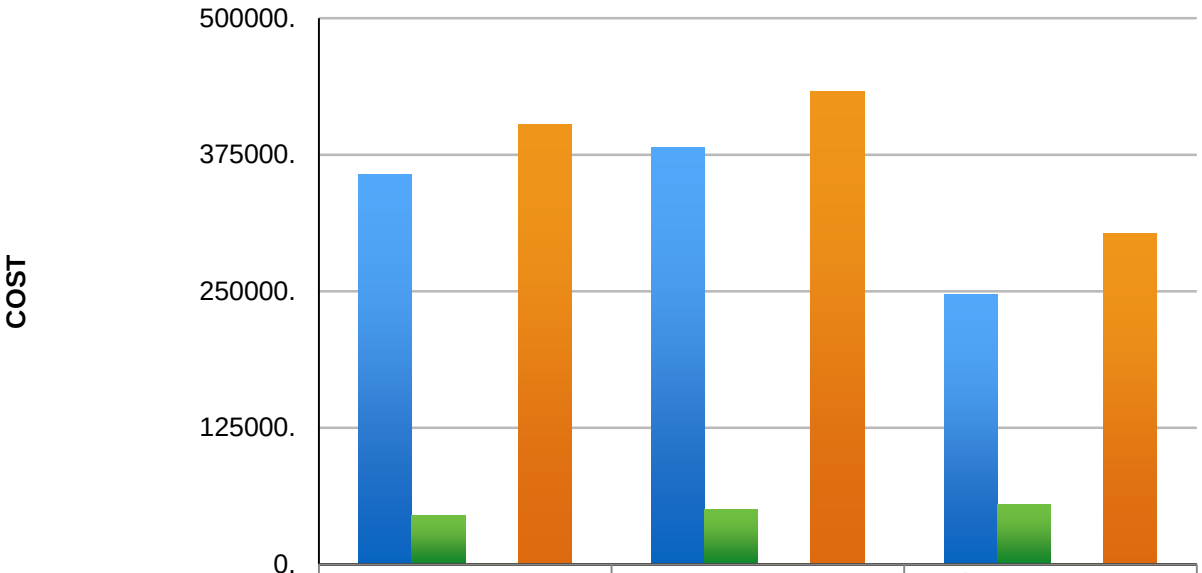
By combining total delay cost, set-up and maintenance cost and accident cost, the total work zone cost is calculated as Table 5 and Figure 8 below.

TABLE 5 TOTAL WORK ZONE COST

	No work zone	One segment	Two segments	Three segments
Total delay cost	0	$356,663	$381,644	$247,178
Set-up Cost	0	$45,000	$50,000	$55,000
Accident Cost	0	$963	$1,030	$667
Total work zone cost	0	$402,626	$432,674	$302,845

The total work zone cost for three plans are shown in Figure 8. From the result we can see that user delay cost is an important contribution to the total cost of work zone. Among the three

work zone plans, the third one has the lowest work zone cost. Although user delay per hour decreases with the number of segments that work zone are divided into, with the total construction time coming into play, the total delay cost is not monotonous. Work zone set-up



	One segment	Two segments	Three segments
Total delay cost	356,663	381,644	247,178
Set-up Cost	45,000	50,000	55,000
Accident Cost	963	1,030	667
Total work zone cost	402,626	432,674	302,845

FIGURE 8 TOTAL WORK ZONE COST

cost increases as the number of segments increases. Accident cost plays a small part compared to other factors and it is proportional to the total delay cost.

Recommendations

This research has illustrated the benefit of considering detours while analyzing how roadway construction might be staged. It has included an explicit costing approach that includes priding options for staging that included detours. In the case study, the work zone cost is derived under three work zone closure staging plans. Since users can select their paths depending on the congestion of the paths, it reflects real life situations better than previous research. It illustrates that it is important for designers/ transportation engineers to pay attention to work zone delay and that they should take advantage of variable-message signs to show road users the roadway closure information as well as alternative paths to choose.

Three factors of work zone cost were also monetized. User delay cost has been calculated along with accident cost and work zone set up and maintenance cost to derive the total work zone cost. It is concluded that the first work zone setting has the least work zone cost in the Sioux-falls network. We can also infer from the statistics that delay cost plays a much more significant role in work zone cost than the other two factors.

The research has its limitations and future work is warranted. This research only considers full road closure. In reality, one lane closure or two lane closure scenarios are widely used in work zone traffic control. Future research may assume one lane closure. Secondly, the traffic flow is assumed to be constant over the 24 hours of the day. To be more accurate, it should reflect the change throughout the day. Lastly, this research does not have a conclusion as for the optimal length of work zone. Future work is needed to create a formulation and solution procedure to identify the optimal segments to use for each stage of the construction and the associated lanes in each of those segments that should be allowed to remain open.

Appendices

Sioux Falls Network

(1)The x and y coordinates of the 24 nodes in Sioux-Falls Network are shown below.

Node	X	Y	
1	50000	510000	;
2	320000	510000	;
3	50000	440000	;
4	130000	440000	;
5	220000	440000	;
6	320000	440000	;
7	420000	380000	;
8	320000	380000	;
9	220000	380000	;
10	220000	320000	;
11	130000	320000	;
12	50000	320000	;
13	50000	50000	;
14	130000	190000	;
15	220000	190000	;
16	320000	320000	;
17	320000	260000	;
18	420000	320000	;
19	320000	190000	;
20	320000	50000	;
21	220000	50000	;
22	220000	130000	;
23	130000	130000	;
24	130000	50000	.

(2) The lengths and capacity of 76 arcs are shown in Table 6.

TABLE 6 SIOUX-FALLS ARC LENGTH AND CAPACITY

Init node	Term node	Capacity	Length
1	2	25900.20064	6
1	3	23403.47319	4
2	1	25900.20064	6
2	6	4958.180928	5
3	1	23403.47319	4
3	4	17110.52372	4
3	12	23403.47319	4
4	3	17110.52372	4
4	5	17782.7941	2
4	11	4908.82673	6
5	4	17782.7941	2
5	6	4947.995469	4
5	9	10000	5
6	2	4958.180928	5
6	5	4947.995469	4
6	8	4898.587646	2
7	8	7841.81131	3
7	18	23403.47319	2
8	6	4898.587646	2
8	7	7841.81131	3
8	9	5050.193156	10
8	16	5045.822583	5
9	5	10000	5

Init node	Term node	Capacity	Length
9	8	5050.193156	10
9	10	13915.78842	3
10	9	13915.78842	3
10	11	10000	5
10	15	13512.00155	6
10	16	4854.917717	4
10	17	4993.510694	8
11	4	4908.82673	6
11	10	10000	5
11	12	4908.82673	6
11	14	4876.508287	4
12	3	23403.47319	4
12	11	4908.82673	6
12	13	25900.20064	3
13	12	25900.20064	3
13	24	5091.256152	4
14	11	4876.508287	4
14	15	5127.526119	5
14	23	4924.790605	4
15	10	13512.00155	6
15	14	5127.526119	5
15	19	14564.75315	3
15	22	9599.180565	3

TABLE 6 (CONTINUED)

Init node	Term node	Capacity	Length
16	8	5045.822583	5
16	10	4854.917717	4
16	17	5229.910063	2
16	18	19679.89671	3
17	10	4993.510694	8
17	16	5229.910063	2
17	19	4823.950831	2
18	7	23403.47319	2
18	16	19679.89671	3
18	20	23403.47319	4
19	15	14564.75315	3
19	17	4823.950831	2
19	20	5002.607563	4
20	18	23403.47319	4
20	19	5002.607563	4
20	21	5059.91234	6
20	22	5075.697193	5
21	20	5059.91234	6
21	22	5229.910063	2
21	24	4885.357564	3
22	15	9599.180565	3
22	20	5075.697193	5
22	21	5229.910063	2

TABLE 6 (CONTINUED)

Init node	Term node	Capacity	Length
22	23	5000	4
23	14	4924.790605	4
23	22	5000	4
23	24	5078.508436	2
24	13	5091.256152	4
24	21	4885.357564	3
24	23	5078.508436	2

TABLE 6 (CONTINUED)

Matlab Code and Output

1. Baseline condition (without work zone)

Code fragments:

```
%Create sparse matrix of the network;
DG=sparse(initial, terminal, time);
h = view(biograph(DG,[ ],'ShowWeights','on'));
alpha=0.15; beta=4;
h = view(biograph(DG,[],'ShowWeights','on'));
[dist,path,pred] = graphshortestpath(DG,21,5)
set(h.Nodes(path),'Color',[1 0.4 0.4])
edges = getedgesbynodeid(h,get(h.Nodes(path),'ID'));
set(edges,'LineColor',[1 0 0])
set(edges,'LineWidth',1.5)

flow1=zeros(76,1);
flow1(65)=1200;
flow1(67)=1200;
flow1(43)=1200;
flow1(26)=1200;
flow1(23)=1200;

DG(21,22)=time(65)*(1+alpha*(1200/capacity(65))^beta);
DG(22,15)=time(67)*(1+alpha*(1200/capacity(67))^beta);
```

```
DG(15,10)=time(43)*(1+alpha*(1200/capacity(43))^beta);

DG(10,9)=time(26)*(1+alpha*(1200/capacity(26))^beta);

DG(9,5)=time(23)*(1+alpha*(1200/capacity(23))^beta);


flow2=zeros(76,1);

flow2(66)=1200;

flow2(74)=1200;

flow2(38)=1200;

flow2(35)=1200;

flow2(6)=1200;

flow2(9)=1200;

%find k that minimize the UE objective function

minobj=99999;

mink=0;

for k=0.01:0.01:1

        flow=flow1*k+flow2*(1-k);

        fun=@(x,c,t) t*(1+alpha*(x/c).^beta);

        obj=integral(@(x)fun(x,capacity(66),time(66)),0,flow(66))+...

        integral(@(x)fun(x,capacity(74),time(74)),0,flow(74))+...

       integral(@(x)fun(x,capacity(38),time(38)),0,flow(38))+...

       integral(@(x)fun(x,capacity(35),time(35)),0,flow(35))+...

       integral(@(x)fun(x,capacity(6),time(6)),0,flow(6))+...

       integral(@(x)fun(x,capacity(9),time(9)),0,flow(9))+...
```

```
            integral(@(x)fun(x,capacity(65),time(65)),0,flow(65))+...
            integral(@(x)fun(x,capacity(67),time(67)),0,flow(67))+...
            integral(@(x)fun(x,capacity(43),time(43)),0,flow(43))+...
            integral(@(x)fun(x,capacity(26),time(26)),0,flow(26))+...
            integral(@(x)fun(x,capacity(23),time(23)),0,flow(23));
             if obj<minobj
                    minobj=obj;
                    mink=k;
                    flow3=flow1*mink+flow2*(1-mink);
             end
    end
     Totaltravel=DG(21,24)*flow3(66)+DG(24,13)*flow3(74)+DG(13,12)*flow3(38)+...
    DG(12,3)*flow3(35)+DG(3,4)*flow3(6)+DG(4,5)*flow3(9)+...
    DG(21,22)*flow3(65)+DG(22,15)*flow3(67)+DG(15,10)*flow3(43)+...
    DG(10,9)*flow3(26)+DG(9,5)*flow3(23)
```

2. Work zone divided into one segment

```
    %create sparse matrix of the network; those in construction are set to 9999
    DG=sparse(initial, terminal, time);
    h = view(biograph(DG,[],'ShowWeights','on'));
    %use typical alpha and beta
    alpha=0.15; beta=4;

    %find the first shortest path x1
```

```
h = view(biograph(DG,[],'ShowWeights','on'));
[dist,path,pred] = graphshortestpath(DG,21,5)
set(h.Nodes(path),'Color',[1 0.4 0.4])
edges = getedgesbynodeid(h,get(h.Nodes(path),'ID'));
set(edges,'LineColor',[1 0 0])
set(edges,'LineWidth',1.5)

%update travel time on the shortest path selected
DG(21,24)=time(66)*(1+alpha*(1200/capacity(66))^beta);
DG(24,13)=time(74)*(1+alpha*(1200/capacity(74))^beta);
DG(13,12)=time(38)*(1+alpha*(1200/capacity(38))^beta);
DG(12,3)=time(35)*(1+alpha*(1200/capacity(35))^beta);
DG(3,4)=time(6)*(1+alpha*(1200/capacity(6))^beta);
DG(4,5)=time(9)*(1+alpha*(1200/capacity(9))^beta);

%update diagram and find the second shortest path y1
h = view(biograph(DG,[],'ShowWeights','on'));
[dist,path,pred] = graphshortestpath(DG,21,5)
set(h.Nodes(path),'Color',[1 0.4 0.4])
edges = getedgesbynodeid(h,get(h.Nodes(path),'ID'));
set(edges,'LineColor',[1 0 0])
set(edges,'LineWidth',1.5)
%set flows to zero and then assign x1 y1 to flow to find the shortest path
```

```
flow=zeros(76,1);
minobj=999999;
mink=0;
for k=0.01:0.01:1
        flow(66)=k*1200;
        flow(74)=k*1200;
        flow(38)=k*1200;
        flow(35)=k*1200;
        flow(6)=k*1200;
        flow(9)=k*1200;
        flow(64)=(1-k)*1200;
        flow(60)=(1-k)*1200;
        flow(54)=(1-k)*1200;
        flow(17)=(1-k)*1200;
        flow(19)=(1-k)*1200;
        flow(15)=(1-k)*1200;
        fun=@(x,c,t) t*(1+alpha*(x/c).^beta);
        obj=integral(@(x)fun(x,capacity(66),time(66)),0,flow(66))+...
            integral(@(x)fun(x,capacity(74),time(74)),0,flow(74))+...
            integral(@(x)fun(x,capacity(38),time(38)),0,flow(38))+...
            integral(@(x)fun(x,capacity(35),time(35)),0,flow(35))+...
            integral(@(x)fun(x,capacity(6),time(6)),0,flow(6))+...
            integral(@(x)fun(x,capacity(9),time(9)),0,flow(9))+...
```

```
    integral(@(x)fun(x,capacity(64),time(64)),0,flow(64))+...
    integral(@(x)fun(x,capacity(60),time(60)),0,flow(60))+...
    integral(@(x)fun(x,capacity(54),time(54)),0,flow(54))+...
    integral(@(x)fun(x,capacity(17),time(17)),0,flow(17))+...
    integral(@(x)fun(x,capacity(19),time(19)),0,flow(19))+...
    integral(@(x)fun(x,capacity(15),time(15)),0,flow(15));
if obj<minobj
    minobj=obj;
    mink=k;
    flow(66)=k*1200;
    flow(74)=k*1200;
    flow(38)=k*1200;
    flow(35)=k*1200;
    flow(6)=k*1200;
    flow(9)=k*1200;
    flow(64)=(1-k)*1200;
    flow(60)=(1-k)*1200;
    flow(54)=(1-k)*1200;
    flow(17)=(1-k)*1200;
    flow(19)=(1-k)*1200;
    flow(15)=(1-k)*1200;
    flow1=flow;
end
```

```
end
minobj
mink

%check if convergence criterion is met
diff=sum((flow7-flow5).^2)
cri_k=sqrt(diff)/sum(flow5)

%calculate the total travel time on all relevant arcs
totaltravel=DG(21,24)*flow9(66)+DG(24,13)*flow9(74)+DG(13,12)*flow9(38)+...
   DG(12,3)*flow9(35)+DG(3,4)*flow9(6)+DG(21,20)*flow9(64)+...
   DG(20,18)*flow9(60)+DG(18,7)*flow9(54)+DG(7,8)*flow9(17)+...
   DG(8,6)*flow9(19)+DG(6,5)*flow9(15)+DG(24,23)*flow9(76)+...
   DG(23,14)*flow9(71)+DG(14,11)*flow9(40)+DG(11,4)*flow9(31)+...
   DG(4,5)*flow9(9)
```

3. Work zone divided into two segments

Code fragments for road closure from node 21 to node 10:

```
%Set construction paths lengths to 9999 to represent full road closure
len(65)=9999;
len(67)=9999;
len(43)=9999;
%time is the free flow travel time depending on length
time=len';
```

```
%create sparse matrix of the network
DG=sparse(initial, terminal, time);
h = view(biograph(DG,[],'ShowWeights','on'));
%use typical alpha and beta
alpha=0.15; beta=4;

%find the first shortest path x1 and find and display shortest graph
h = view(biograph(DG,[],'ShowWeights','on'));
[dist,path,pred] = graphshortestpath(DG,21,5)
set(h.Nodes(path),'Color',[1 0.4 0.4])
edges = getedgesbynodeid(h,get(h.Nodes(path),'ID'));
set(edges,'LineColor',[1 0 0])
set(edges,'LineWidth',1.5)
flow1=zeros(76,1);
flow1(66)=1200;
flow1(74)=1200;
flow1(38)=1200;
flow1(35)=1200;
flow1(6)=1200;
flow1(9)=1200;

%Update travel time on arcs
DG(21,24)=time(66)*(1+alpha*(flow1(66)/capacity(66)).^beta);
```

```
DG(24,13)=time(74)*(1+alpha*(flow1(74)/capacity(74)).^beta);
DG(13,12)=time(38)*(1+alpha*(flow1(38)/capacity(38)).^beta);
DG(12,3)=time(35)*(1+alpha*(flow1(35)/capacity(35)).^beta);
DG(3,4)=time(6)*(1+alpha*(flow1(6)/capacity(6)).^beta);
DG(4,5)=time(9)*(1+alpha*(flow1(9)/capacity(9)).^beta);

h = view(biograph(DG,[],'ShowWeights','on'));
[dist,path,pred] = graphshortestpath(DG,21,5)
set(h.Nodes(path),'Color',[1 0.4 0.4])
edges = getedgesbynodeid(h,get(h.Nodes(path),'ID'));
set(edges,'LineColor',[1 0 0])
set(edges,'LineWidth',1.5)
%assign all traffic to flow2
flow2=zeros(76,1);
flow2(64)=1200;
flow2(60)=1200;
flow2(54)=1200;
flow2(17)=1200;
flow2(19)=1200;
flow2(15)=1200;
%find the optimal k for minimizing the objective function of UE
minobj=99999;
mink=0;
```

```
for k=0.01:0.01:1
        flow3=flow1*k+flow2*(1-k);
        fun=@(x,c,t) t*(1+alpha*(x/c).^beta);
        obj=integral(@(x)fun(x,capacity(64),time(64)),0,flow3(64))+...
           integral(@(x)fun(x,capacity(60),time(60)),0,flow3(60))+...
           integral(@(x)fun(x,capacity(54),time(54)),0,flow3(54))+...
           integral(@(x)fun(x,capacity(17),time(17)),0,flow3(17))+...
           integral(@(x)fun(x,capacity(19),time(19)),0,flow3(19))+...
           integral(@(x)fun(x,capacity(15),time(15)),0,flow3(15))+...
           integral(@(x)fun(x,capacity(66),time(66)),0,flow3(66))+...
           integral(@(x)fun(x,capacity(74),time(74)),0,flow3(74))+...
           integral(@(x)fun(x,capacity(38),time(38)),0,flow3(38))+...
           integral(@(x)fun(x,capacity(35),time(35)),0,flow3(35))+...
           integral(@(x)fun(x,capacity(6),time(6)),0,flow3(6))+...
           integral(@(x)fun(x,capacity(9),time(9)),0,flow3(9));
        if obj<minobj
                minobj=obj;
                mink=k;
        end
        flow3=flow1*mink+flow2*(1-mink);
end

%from 21 to 5 total time on all arcs
```

```
time1=DG(21,20)*flow5(64)+DG(20,18)*flow5(60)+DG(18,7)*flow5(54)+...
   DG(7,8)*flow5(17)+DG(8,6)*flow5(19)+DG(6,5)*flow5(15)+...
   +DG(21,24)*flow5(66)+DG(24,13)*flow5(74)+DG(13,12)*flow5(38)+...
   DG(12,3)*flow5(35)+DG(3,4)*flow5(6)+DG(4,5)*flow5(9)+...
   DG(24,23)*flow5(76)+DG(23,14)*flow5(71)+DG(14,11)*flow5(40)+...
   DG(11,4)*flow5(31)
```

Code fragments for road closure from node 10 to node 5:

```
%Set construction paths lengths to 9999 to represent full road closure
len(26)=9999;
len(23)=9999;

%time is the free flow travel time depending on length
time=len';
%create sparse matrix of the network; those in construction are set to 9999
DG=sparse(initial, terminal, time);
h = view(biograph(DG,[],'ShowWeights','on'));
%use typical alpha and beta
alpha=0.15; beta=4;

%find the first shortest path x1
h = view(biograph(DG,[],'ShowWeights','on'));
[dist,path,pred] = graphshortestpath(DG,21,5)
set(h.Nodes(path),'Color',[1 0.4 0.4])
```

```
edges = getedgesbynodeid(h,get(h.Nodes(path),'ID'));
set(edges,'LineColor',[1 0 0])
set(edges,'LineWidth',1.5)

flow1=zeros(76,1);
flow1(66)=1200;
flow1(74)=1200;
flow1(38)=1200;
flow1(35)=1200;
flow1(6)=1200;
flow1(9)=1200;

DG(21,24)=time(66)*(1+alpha*(flow1(66)/capacity(66)).^beta);
DG(24,13)=time(74)*(1+alpha*(flow1(74)/capacity(74)).^beta);
DG(13,12)=time(38)*(1+alpha*(flow1(38)/capacity(38)).^beta);
DG(12,3)=time(35)*(1+alpha*(flow1(35)/capacity(35)).^beta);
DG(3,4)=time(6)*(1+alpha*(flow1(6)/capacity(6)).^beta);
DG(4,5)=time(9)*(1+alpha*(flow1(9)/capacity(9)).^beta);

h = view(biograph(DG,[],'ShowWeights','on'));
[dist,path,pred] = graphshortestpath(DG,21,5)
set(h.Nodes(path),'Color',[1 0.4 0.4])
edges = getedgesbynodeid(h,get(h.Nodes(path),'ID'));
```

```
set(edges,'LineColor',[1 0 0])
set(edges,'LineWidth',1.5)
flow2=zeros(76,1);
flow2(64)=1200;
flow2(60)=1200;
flow2(54)=1200;
flow2(17)=1200;
flow2(19)=1200;
flow2(15)=1200;

minobj=99999;
mink=0;
for k=0.01:0.01:1
    flow3=flow1*k+flow2*(1-k);
    fun=@(x,c,t) t*(1+alpha*(x/c).^beta);
    obj=integral(@(x)fun(x,capacity(64),time(64)),0,flow3(64))+...
        integral(@(x)fun(x,capacity(60),time(60)),0,flow3(60))+...
        integral(@(x)fun(x,capacity(54),time(54)),0,flow3(54))+...
        integral(@(x)fun(x,capacity(17),time(17)),0,flow3(17))+...
        integral(@(x)fun(x,capacity(19),time(19)),0,flow3(19))+...
        integral(@(x)fun(x,capacity(15),time(15)),0,flow3(15))+...
        integral(@(x)fun(x,capacity(66),time(66)),0,flow3(66))+...
        integral(@(x)fun(x,capacity(74),time(74)),0,flow3(74))+...
```

```
        integral(@(x)fun(x,capacity(38),time(38)),0,flow3(38))+...
        integral(@(x)fun(x,capacity(35),time(35)),0,flow3(35))+...
        integral(@(x)fun(x,capacity(6),time(6)),0,flow3(6))+...
        integral(@(x)fun(x,capacity(9),time(9)),0,flow3(9));
    if obj<minobj
        minobj=obj;
        mink=k;
    end
    flow3=flow1*mink+flow2*(1-mink);
end
minobj
mink

DG(21,20)=time(64)*(1+alpha*(flow3(64)/capacity(64)).^beta);
DG(20,18)=time(60)*(1+alpha*(flow3(60)/capacity(60)).^beta);
DG(18,7)=time(54)*(1+alpha*(flow3(54)/capacity(54)).^beta);
DG(7,8)=time(17)*(1+alpha*(flow3(17)/capacity(7)).^beta);
DG(8,6)=time(19)*(1+alpha*(flow3(19)/capacity(19)).^beta);
DG(6,5)=time(15)*(1+alpha*(flow3(15)/capacity(15)).^beta);
DG(21,24)=time(66)*(1+alpha*(flow3(66)/capacity(66)).^beta);
DG(24,13)=time(74)*(1+alpha*(flow3(74)/capacity(74)).^beta);
DG(13,12)=time(38)*(1+alpha*(flow3(38)/capacity(38)).^beta);
DG(12,3)=time(35)*(1+alpha*(flow3(35)/capacity(35)).^beta);
```

```
DG(3,4)=time(6)*(1+alpha*(flow3(6)/capacity(6)).^beta);
DG(4,5)=time(9)*(1+alpha*(flow3(9)/capacity(9)).^beta);

h = view(biograph(DG,[],'ShowWeights','on'));
[dist,path,pred] = graphshortestpath(DG,21,5)
set(h.Nodes(path),'Color',[1 0.4 0.4])
edges = getedgesbynodeid(h,get(h.Nodes(path),'ID'));
set(edges,'LineColor',[1 0 0])
set(edges,'LineWidth',1.5)

flow4=zeros(76,1);
flow4(65)=1200;
flow4(67)=1200;
flow4(44)=1200;
flow4(40)=1200;
flow4(31)=1200;
flow4(9)=1200;

minobj=99999;
mink=0;
for k=0.01:0.01:1
    flow=flow3*k+flow4*(1-k);
    fun=@(x,c,t) t*(1+alpha*(x/c).^beta);
```

```
obj=integral(@(x)fun(x,capacity(64),time(64)),0,flow(64))+...
    integral(@(x)fun(x,capacity(60),time(60)),0,flow(60))+...
    integral(@(x)fun(x,capacity(54),time(54)),0,flow(54))+...
    integral(@(x)fun(x,capacity(17),time(17)),0,flow(17))+...
    integral(@(x)fun(x,capacity(19),time(19)),0,flow(19))+...
    integral(@(x)fun(x,capacity(15),time(15)),0,flow(15))+...
    integral(@(x)fun(x,capacity(66),time(66)),0,flow(66))+...
    integral(@(x)fun(x,capacity(74),time(74)),0,flow(74))+...
    integral(@(x)fun(x,capacity(38),time(38)),0,flow(38))+...
    integral(@(x)fun(x,capacity(35),time(35)),0,flow(35))+...
    integral(@(x)fun(x,capacity(6),time(6)),0,flow(6))+...
    integral(@(x)fun(x,capacity(9),time(9)),0,flow(9))+...
    integral(@(x)fun(x,capacity(65),time(65)),0,flow(65))+...
    integral(@(x)fun(x,capacity(67),time(67)),0,flow(67))+...
    integral(@(x)fun(x,capacity(44),time(44)),0,flow(44))+...
    integral(@(x)fun(x,capacity(40),time(40)),0,flow(40))+...
    integral(@(x)fun(x,capacity(31),time(31)),0,flow(31))+...
    integral(@(x)fun(x,capacity(9),time(9)),0,flow(9));
if obj<minobj
    minobj=obj;
    mink=k;
    flow5=flow3*mink+flow4*(1-mink);
end
```

```
end
minobj
mink

DG(21,20)=time(64)*(1+alpha*(flow5(64)/capacity(64)).^beta);
DG(20,18)=time(60)*(1+alpha*(flow5(60)/capacity(60)).^beta);
DG(18,7)=time(54)*(1+alpha*(flow5(54)/capacity(54)).^beta);
DG(7,8)=time(17)*(1+alpha*(flow5(17)/capacity(7)).^beta);
DG(8,6)=time(19)*(1+alpha*(flow5(19)/capacity(19)).^beta);
DG(6,5)=time(15)*(1+alpha*(flow5(15)/capacity(15)).^beta);
DG(21,24)=time(66)*(1+alpha*(flow5(66)/capacity(66)).^beta);
DG(24,13)=time(74)*(1+alpha*(flow5(74)/capacity(74)).^beta);
DG(13,12)=time(38)*(1+alpha*(flow5(38)/capacity(38)).^beta);
DG(12,3)=time(35)*(1+alpha*(flow5(35)/capacity(35)).^beta);
DG(3,4)=time(6)*(1+alpha*(flow5(6)/capacity(6)).^beta);
DG(4,5)=time(9)*(1+alpha*(flow5(9)/capacity(9)).^beta);
DG(21,22)=time(65)*(1+alpha*(flow5(65)/capacity(65)).^beta);
DG(22,15)=time(67)*(1+alpha*(flow5(67)/capacity(67)).^beta);
DG(15,14)=time(41)*(1+alpha*(flow5(41)/capacity(41)).^beta);
DG(14,11)=time(40)*(1+alpha*(flow5(40)/capacity(40)).^beta);
DG(11,4)=time(31)*(1+alpha*(flow5(31)/capacity(31)).^beta);

%from 21 to 5 total time
```

```
time1=DG(21,20)*flow5(64)+DG(20,18)*flow5(60)+DG(18,7)*flow5(54)+...
   DG(7,8)*flow5(17)+DG(8,6)*flow5(19)+DG(6,5)*flow5(15)+...
   +DG(21,24)*flow5(66)+DG(24,13)*flow5(74)+DG(13,12)*flow5(38)+...
   DG(12,3)*flow5(35)+DG(3,4)*flow5(6)+DG(4,5)*flow5(9)+...
   DG(21,22)*flow5(65)+DG(22,15)*flow5(67)+DG(15,14)*flow5(44)+...
   DG(14,11)*flow5(40)+DG(11,4)*flow5(31)
```

4. Work zone divided into three segments

Code fragments for road closure from node 21 to node 15:

```
len(65)=9999;
len(67)=9999;

%time is the free flow travel time depending on length
time=len';

%create sparse matrix of the network; those in construction are set to 9999
DG=sparse(initial, terminal, time);
h = view(biograph(DG,[],'ShowWeights','on'));

alpha=0.15; beta=4;
%find the first shortest path x1/ second graph
h = view(biograph(DG,[],'ShowWeights','on'));
[dist,path,pred] = graphshortestpath(DG,21,5)
```

```
set(h.Nodes(path),'Color',[1 0.4 0.4])
edges = getedgesbynodeid(h,get(h.Nodes(path),'ID'));
set(edges,'LineColor',[1 0 0])
set(edges,'LineWidth',1.5)

flow1=zeros(76,1);
flow1(66)=1200;
flow1(74)=1200;
flow1(38)=1200;
flow1(35)=1200;
flow1(6)=1200;
flow1(9)=1200;

DG(21,24)=time(66)*(1+alpha*(flow1(66)/capacity(66)).^beta);
DG(24,13)=time(74)*(1+alpha*(flow1(74)/capacity(74)).^beta);
DG(13,12)=time(38)*(1+alpha*(flow1(38)/capacity(38)).^beta);
DG(12,3)=time(35)*(1+alpha*(flow1(35)/capacity(35)).^beta);
DG(3,4)=time(6)*(1+alpha*(flow1(6)/capacity(6)).^beta);
DG(4,5)=time(9)*(1+alpha*(flow1(9)/capacity(9)).^beta);

%find the first shortest path x1/ second graph
h = view(biograph(DG,[],'ShowWeights','on'));
[dist,path,pred] = graphshortestpath(DG,21,5)
```

```
set(h.Nodes(path),'Color',[1 0.4 0.4])
edges = getedgesbynodeid(h,get(h.Nodes(path),'ID'));
set(edges,'LineColor',[1 0 0])
set(edges,’LineWidth',1.5)

flow2=zeros(76,1);
flow2(64)=1200;
flow2(60)=1200;
flow2(54)=1200;
flow2(17)=1200;
flow2(19)=1200;
flow2(15)=1200;

minobj=99999;
mink=0;
for k=0.01:0.01:1
        flow3=flow1*k+flow2*(1-k);
        fun=@(x,c,t) t*(1+alpha*(x/c).^beta);
        obj=integral(@(x)fun(x,capacity(64),time(64)),0,flow3(64))+...
            integral(@(x)fun(x,capacity(60),time(60)),0,flow3(60))+...
            integral(@(x)fun(x,capacity(54),time(54)),0,flow3(54))+...
            integral(@(x)fun(x,capacity(17),time(17)),0,flow3(17))+...
            integral(@(x)fun(x,capacity(19),time(19)),0,flow3(19))+...
```

```
            integral(@(x)fun(x,capacity(15),time(15)),0,flow3(15))+...
            integral(@(x)fun(x,capacity(66),time(66)),0,flow3(66))+...
            integral(@(x)fun(x,capacity(74),time(74)),0,flow3(74))+...
            integral(@(x)fun(x,capacity(38),time(38)),0,flow3(38))+...
            integral(@(x)fun(x,capacity(35),time(35)),0,flow3(35))+...
            integral(@(x)fun(x,capacity(6),time(6)),0,flow3(6))+...
            integral(@(x)fun(x,capacity(9),time(9)),0,flow3(9));
        if obj<minobj
           minobj=obj;
           mink=k;
        end
        flow3=flow1*mink+flow2*(1-mink);
end
minobj
mink

%from 21 to 5 total time
time1=DG(21,20)*flow5(64)+DG(20,18)*flow5(60)+DG(18,7)*flow5(54)+...
    DG(7,8)*flow5(17)+DG(8,6)*flow5(19)+DG(6,5)*flow5(15)+...
    +DG(21,24)*flow5(66)+DG(24,13)*flow5(74)+DG(13,12)*flow5(38)+...
    DG(12,3)*flow5(35)+DG(3,4)*flow5(6)+DG(4,5)*flow5(9)+...
    DG(24,23)*flow5(76)+DG(23,14)*flow5(71)+DG(14,11)*flow5(40)+...
    DG(11,4)*flow5(31)
```

Code fragments for road closure from node 15 to node 10:

```
len(43)=9999;
%time is the free flow travel time depending on length
time=len';

%create sparse matrix of the network; those in construction are set to 9999
DG=sparse(initial, terminal, time);
h = view(biograph(DG,[],'ShowWeights','on'));
%use typical alpha and beta
alpha=0.15; beta=4;

%find the first shortest path x1/ second graph
h = view(biograph(DG,[],'ShowWeights','on'));
[dist,path,pred] = graphshortestpath(DG,21,5)
set(h.Nodes(path),'Color',[1 0.4 0.4])
edges = getedgesbynodeid(h,get(h.Nodes(path),'ID'));
set(edges,'LineColor',[1 0 0])
set(edges,'LineWidth',1.5)

flow1=zeros(76,1);
flow1(66)=1200;
flow1(74)=1200;
flow1(38)=1200;
```

```
flow1(35)=1200;
flow1(6)=1200;
flow1(9)=1200;

DG(21,24)=time(66)*(1+alpha*(flow1(66)/capacity(66)).^beta);
DG(24,13)=time(74)*(1+alpha*(flow1(74)/capacity(74)).^beta);
DG(13,12)=time(38)*(1+alpha*(flow1(38)/capacity(38)).^beta);
DG(12,3)=time(35)*(1+alpha*(flow1(35)/capacity(35)).^beta);
DG(3,4)=time(6)*(1+alpha*(flow1(6)/capacity(6)).^beta);
DG(4,5)=time(9)*(1+alpha*(flow1(9)/capacity(9)).^beta);

%find the first shortest path x1/ second graph
h = view(biograph(DG,[],'ShowWeights','on'));
[dist,path,pred] = graphshortestpath(DG,21,5)
set(h.Nodes(path),'Color',[1 0.4 0.4])
edges = getedgesbynodeid(h,get(h.Nodes(path),'ID'));
set(edges,'LineColor',[1 0 0])
set(edges,'LineWidth',1.5)

flow2=zeros(76,1);
flow2(64)=1200;
flow2(60)=1200;
flow2(54)=1200;
```

```
flow2(17)=1200;
flow2(19)=1200;
flow2(15)=1200;

minobj=99999;
mink=0;
for k=0.01:0.01:1
    flow3=flow1*k+flow2*(1-k);
    fun=@(x,c,t) t*(1+alpha*(x/c).^beta);
    obj=integral(@(x)fun(x,capacity(64),time(64)),0,flow3(64))+...
        integral(@(x)fun(x,capacity(60),time(60)),0,flow3(60))+...
        integral(@(x)fun(x,capacity(54),time(54)),0,flow3(54))+...
        integral(@(x)fun(x,capacity(17),time(17)),0,flow3(17))+...
        integral(@(x)fun(x,capacity(19),time(19)),0,flow3(19))+...
        integral(@(x)fun(x,capacity(15),time(15)),0,flow3(15))+...
        integral(@(x)fun(x,capacity(66),time(66)),0,flow3(66))+...
        integral(@(x)fun(x,capacity(74),time(74)),0,flow3(74))+...
        integral(@(x)fun(x,capacity(38),time(38)),0,flow3(38))+...
        integral(@(x)fun(x,capacity(35),time(35)),0,flow3(35))+...
        integral(@(x)fun(x,capacity(6),time(6)),0,flow3(6))+...
        integral(@(x)fun(x,capacity(9),time(9)),0,flow3(9));
    if obj<minobj
        minobj=obj;
```

```
        mink=k;
    end
    flow3=flow1*mink+flow2*(1-mink);
end
minobj
mink

DG(21,20)=time(64)*(1+alpha*(flow3(64)/capacity(64)).^beta);
DG(20,18)=time(60)*(1+alpha*(flow3(60)/capacity(60)).^beta);
DG(18,7)=time(54)*(1+alpha*(flow3(54)/capacity(54)).^beta);
DG(7,8)=time(17)*(1+alpha*(flow3(17)/capacity(7)).^beta);
DG(8,6)=time(19)*(1+alpha*(flow3(19)/capacity(19)).^beta);
DG(6,5)=time(15)*(1+alpha*(flow3(15)/capacity(15)).^beta);
DG(21,24)=time(66)*(1+alpha*(flow3(66)/capacity(66)).^beta);
DG(24,13)=time(74)*(1+alpha*(flow3(74)/capacity(74)).^beta);
DG(13,12)=time(38)*(1+alpha*(flow3(38)/capacity(38)).^beta);
DG(12,3)=time(35)*(1+alpha*(flow3(35)/capacity(35)).^beta);
DG(3,4)=time(6)*(1+alpha*(flow3(6)/capacity(6)).^beta);
DG(4,5)=time(9)*(1+alpha*(flow3(9)/capacity(9)).^beta);

h = view(biograph(DG,[],'ShowWeights','on'));
[dist,path,pred] = graphshortestpath(DG,21,5)
set(h.Nodes(path),'Color',[1 0.4 0.4])
```

```
edges = getedgesbynodeid(h,get(h.Nodes(path),'ID'));
set(edges,'LineColor',[1 0 0]);
set(edges,'LineWidth',1.5)

flow4=zeros(76,1);
flow4(65)=1200;
flow4(67)=1200;
flow4(44)=1200;
flow4(40)=1200;
flow4(31)=1200;
flow4(9)=1200;

minobj=99999;
mink=0;
for k=0.01:0.01:1
    flow=flow3*k+flow4*(1-k);
    fun=@(x,c,t) t*(1+alpha*(x/c).^beta);
    obj=integral(@(x)fun(x,capacity(64),time(64)),0,flow(64))+...
        integral(@(x)fun(x,capacity(60),time(60)),0,flow(60))+...
        integral(@(x)fun(x,capacity(54),time(54)),0,flow(54))+...
        integral(@(x)fun(x,capacity(17),time(17)),0,flow(17))+...
        integral(@(x)fun(x,capacity(19),time(19)),0,flow(19))+...
        integral(@(x)fun(x,capacity(15),time(15)),0,flow(15))+...
```

```
        integral(@(x)fun(x,capacity(66),time(66)),0,flow(66))+...
        integral(@(x)fun(x,capacity(74),time(74)),0,flow(74))+...
        integral(@(x)fun(x,capacity(38),time(38)),0,flow(38))+...
        integral(@(x)fun(x,capacity(35),time(35)),0,flow(35))+...
        integral(@(x)fun(x,capacity(6),time(6)),0,flow(6))+...
        integral(@(x)fun(x,capacity(9),time(9)),0,flow(9))+...
        integral(@(x)fun(x,capacity(65),time(65)),0,flow(65))+...
        integral(@(x)fun(x,capacity(67),time(67)),0,flow(67))+...
        integral(@(x)fun(x,capacity(44),time(44)),0,flow(44))+...
        integral(@(x)fun(x,capacity(40),time(40)),0,flow(40))+...
        integral(@(x)fun(x,capacity(31),time(31)),0,flow(31));
    if obj<minobj
        minobj=obj;
        mink=k;
        flow5=flow3*mink+flow4*(1-mink);
    end

end
minobj
mink

%from 21 to 5 total time
time1=DG(21,20)*flow5(64)+DG(20,18)*flow5(60)+DG(18,7)*flow5(54)+...
```

```
DG(7,8)*flow5(17)+DG(8,6)*flow5(19)+DG(6,5)*flow5(15)+...
+DG(21,24)*flow5(66)+DG(24,13)*flow5(74)+DG(13,12)*flow5(38)+...
DG(12,3)*flow5(35)+DG(3,4)*flow5(6)+DG(4,5)*flow5(9)+...
DG(21,22)*flow5(65)+DG(22,15)*flow5(67)+DG(15,14)*flow5(41)+...
DG(14,11)*flow5(40)+DG(11,4)*flow5(31)
```

Code fragments for road closure from node 15 to node 10:

```
len(26)=9999;
len(23)=9999;

%time is the free flow travel time depending on length
time=len';

%create sparse matrix of the network; those in construction are set to 9999
DG=sparse(initial, terminal, time);
h = view(biograph(DG,[],'ShowWeights','on'));
%use typical alpha and beta
alpha=0.15; beta=4;

%find the first shortest path x1/ second graph
h = view(biograph(DG,[],'ShowWeights','on'));
[dist,path,pred] = graphshortestpath(DG,21,5)
set(h.Nodes(path),'Color',[1 0.4 0.4])
```

```
edges = getedgesbynodeid(h,get(h.Nodes(path),'ID'));
set(edges,'LineColor',[1 0 0])
set(edges,'LineWidth',1.5)

flow1=zeros(76,1);
flow1(66)=1200;
flow1(74)=1200;
flow1(38)=1200;
flow1(35)=1200;
flow1(6)=1200;
flow1(9)=1200;

DG(21,24)=time(66)*(1+alpha*(flow1(66)/capacity(66)).^beta);
DG(24,13)=time(74)*(1+alpha*(flow1(74)/capacity(74)).^beta);
DG(13,12)=time(38)*(1+alpha*(flow1(38)/capacity(38)).^beta);
DG(12,3)=time(35)*(1+alpha*(flow1(35)/capacity(35)).^beta);
DG(3,4)=time(6)*(1+alpha*(flow1(6)/capacity(6)).^beta);
DG(4,5)=time(9)*(1+alpha*(flow1(9)/capacity(9)).^beta);

%find the first shortest path x1/ second graph
h = view(biograph(DG,[],'ShowWeights','on'));
[dist,path,pred] = graphshortestpath(DG,21,5)
set(h.Nodes(path),'Color',[1 0.4 0.4])
```

```
edges = getedgesbynodeid(h,get(h.Nodes(path),'ID'));
set(edges,'LineColor',[1 0 0])
set(edges,'LineWidth',1.5)

flow2=zeros(76,1);
flow2(64)=1200;
flow2(60)=1200;
flow2(54)=1200;
flow2(17)=1200;
flow2(19)=1200;
flow2(15)=1200;

minobj=99999;
mink=0;
for k=0.01:0.01:1
    flow3=flow1*k+flow2*(1-k);
    fun=@(x,c,t) t*(1+alpha*(x/c).^beta);
    obj=integral(@(x)fun(x,capacity(64),time(64)),0,flow3(64))+...
        integral(@(x)fun(x,capacity(60),time(60)),0,flow3(60))+...
        integral(@(x)fun(x,capacity(54),time(54)),0,flow3(54))+...
        integral(@(x)fun(x,capacity(17),time(17)),0,flow3(17))+...
        integral(@(x)fun(x,capacity(19),time(19)),0,flow3(19))+...
        integral(@(x)fun(x,capacity(15),time(15)),0,flow3(15))+...
```

```
        integral(@(x)fun(x,capacity(66),time(66)),0,flow3(66))+...
        integral(@(x)fun(x,capacity(74),time(74)),0,flow3(74))+...
        integral(@(x)fun(x,capacity(38),time(38)),0,flow3(38))+...
        integral(@(x)fun(x,capacity(35),time(35)),0,flow3(35))+...
        integral(@(x)fun(x,capacity(6),time(6)),0,flow3(6))+...
        integral(@(x)fun(x,capacity(9),time(9)),0,flow3(9));
    if obj<minobj
        minobj=obj;
        mink=k;
    end
    flow3=flow1*mink+flow2*(1-mink);
end
minobj
mink

DG(21,20)=time(64)*(1+alpha*(flow3(64)/capacity(64)).^beta);
DG(20,18)=time(60)*(1+alpha*(flow3(60)/capacity(60)).^beta);
DG(18,7)=time(54)*(1+alpha*(flow3(54)/capacity(54)).^beta);
DG(7,8)=time(17)*(1+alpha*(flow3(17)/capacity(7)).^beta);
DG(8,6)=time(19)*(1+alpha*(flow3(19)/capacity(19)).^beta);
DG(6,5)=time(15)*(1+alpha*(flow3(15)/capacity(15)).^beta);
DG(21,24)=time(66)*(1+alpha*(flow3(66)/capacity(66)).^beta);
DG(24,13)=time(74)*(1+alpha*(flow3(74)/capacity(74)).^beta);
```

```
DG(13,12)=time(38)*(1+alpha*(flow3(38)/capacity(38)).^beta);

DG(12,3)=time(35)*(1+alpha*(flow3(35)/capacity(35)).^beta);

DG(3,4)=time(6)*(1+alpha*(flow3(6)/capacity(6)).^beta);

DG(4,5)=time(9)*(1+alpha*(flow3(9)/capacity(9)).^beta);


h = view(biograph(DG,[],'ShowWeights','on'));

[dist,path,pred] = graphshortestpath(DG,21,5)

set(h.Nodes(path),'Color',[1 0.4 0.4])

edges = getedgesbynodeid(h,get(h.Nodes(path),'ID'));

set(edges,'LineColor',[1 0 0]);

set(edges,'LineWidth',1.5)


flow4=zeros(76,1);

flow4(65)=1200;

flow4(67)=1200;

flow4(44)=1200;

flow4(40)=1200;

flow4(31)=1200;

flow4(9)=1200;


minobj=99999;

mink=0;

for k=0.01:0.01:1
```

```
flow=flow3*k+flow4*(1-k);
fun=@(x,c,t) t*(1+alpha*(x/c).^beta);
obj=integral(@(x)fun(x,capacity(64),time(64)),0,flow(64))+...
    integral(@(x)fun(x,capacity(60),time(60)),0,flow(60))+...
    integral(@(x)fun(x,capacity(54),time(54)),0,flow(54))+...
    integral(@(x)fun(x,capacity(17),time(17)),0,flow(17))+...
    integral(@(x)fun(x,capacity(19),time(19)),0,flow(19))+...
    integral(@(x)fun(x,capacity(15),time(15)),0,flow(15))+...
    integral(@(x)fun(x,capacity(66),time(66)),0,flow(66))+...
    integral(@(x)fun(x,capacity(74),time(74)),0,flow(74))+...
    integral(@(x)fun(x,capacity(38),time(38)),0,flow(38))+...
    integral(@(x)fun(x,capacity(35),time(35)),0,flow(35))+...
    integral(@(x)fun(x,capacity(6),time(6)),0,flow(6))+...
    integral(@(x)fun(x,capacity(9),time(9)),0,flow(9))+...
    integral(@(x)fun(x,capacity(65),time(65)),0,flow(65))+...
    integral(@(x)fun(x,capacity(67),time(67)),0,flow(67))+...
    integral(@(x)fun(x,capacity(44),time(44)),0,flow(44))+...
    integral(@(x)fun(x,capacity(40),time(40)),0,flow(40))+...
    integral(@(x)fun(x,capacity(31),time(31)),0,flow(31));
if obj<minobj
    minobj=obj;
    mink=k;
    flow5=flow3*mink+flow4*(1-mink);
```

```
    end
end
minobj
mink

%from 21 to 5 total time
time1=DG(21,20)*flow5(64)+DG(20,18)*flow5(60)+DG(18,7)*flow5(54)+...
    DG(7,8)*flow5(17)+DG(8,6)*flow5(19)+DG(6,5)*flow5(15)+...
    +DG(21,24)*flow5(66)+DG(24,13)*flow5(74)+DG(13,12)*flow5(38)+...
    DG(12,3)*flow5(35)+DG(3,4)*flow5(6)+DG(4,5)*flow5(9)+...
    DG(21,22)*flow5(65)+DG(22,15)*flow5(67)+DG(15,14)*flow5(41)+...
    DG(14,11)*flow5(40)+DG(11,4)*flow5(31)
```

Output Examples

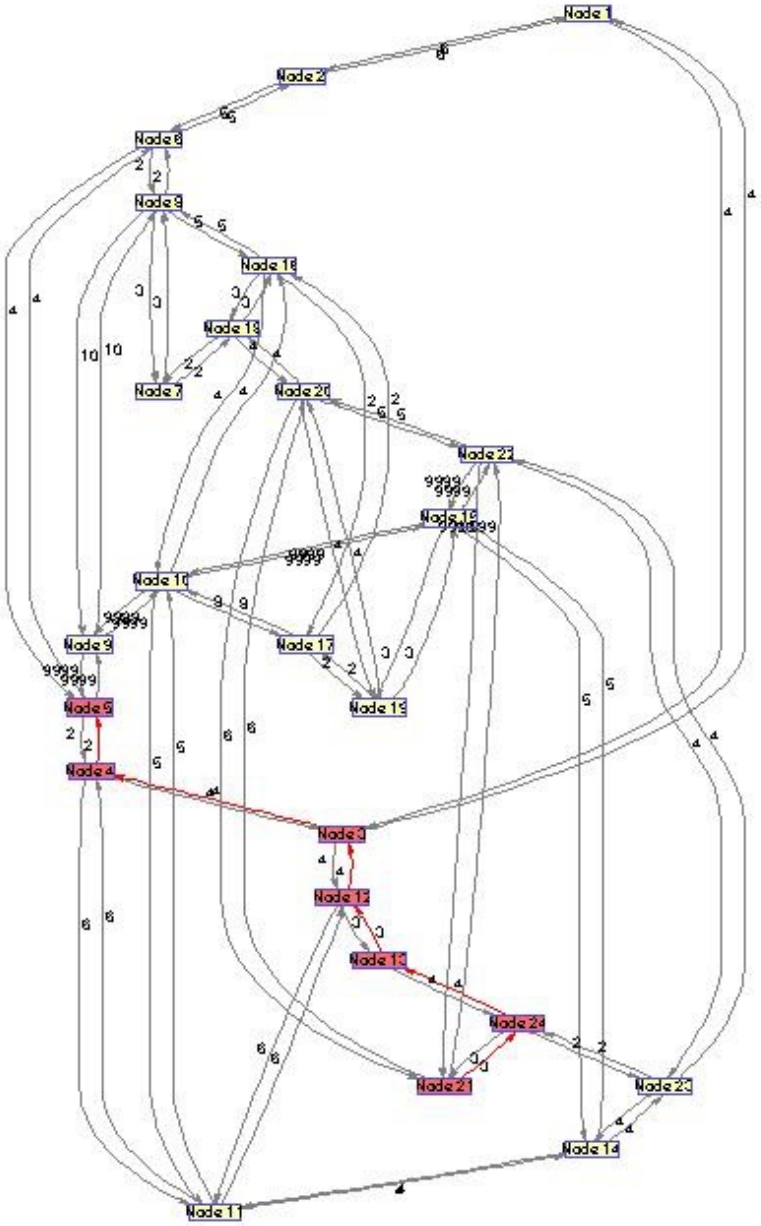


FIGURE 9 OUTPUT EXAMPLE 1

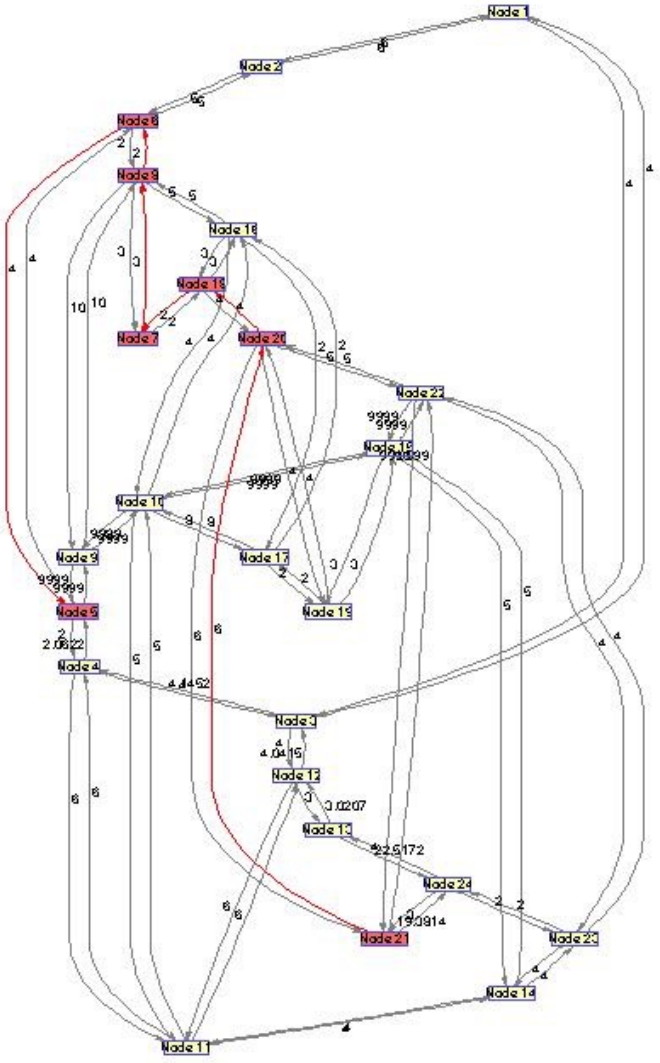


FIGURE 10 OUTPUT EXAMPLE 2

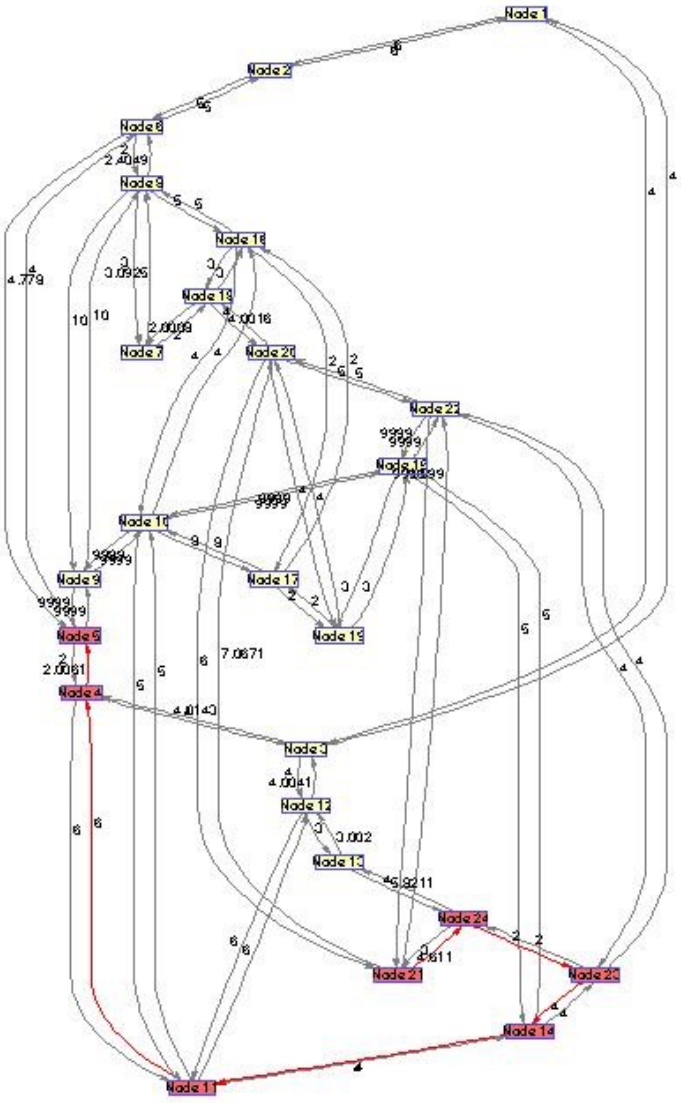


FIGURE 11 OUTPUT EXAMPLE 3

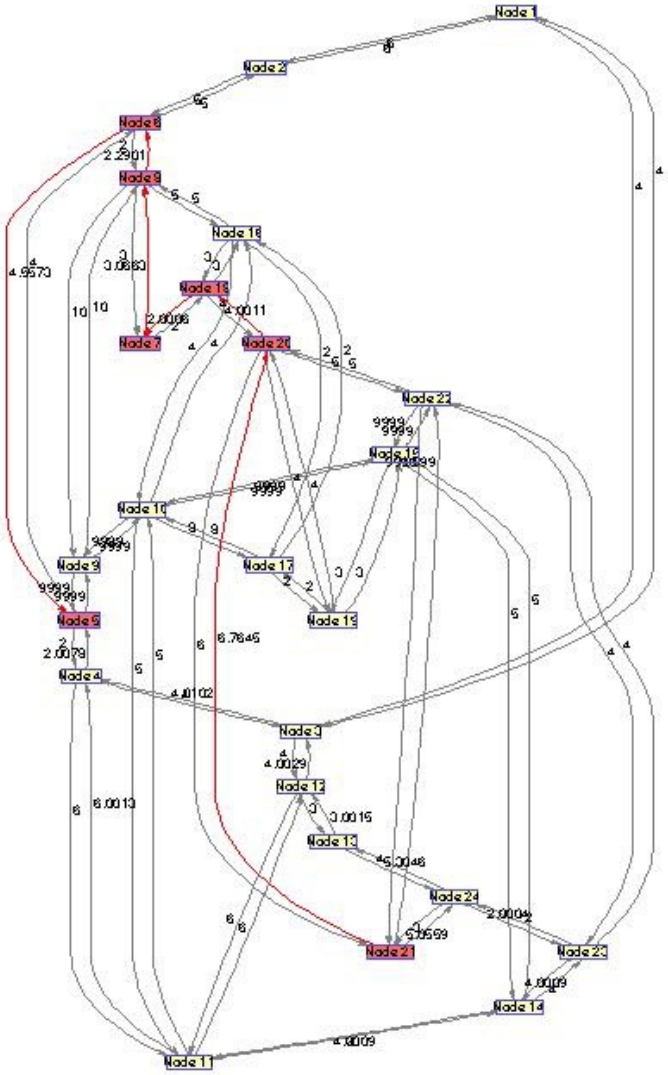


FIGURE 12 OUTPUT EXAMPLE 4

References

[1] Transportation Research Board (2010), Highway Capacity Manual 2010

[2] Yossi Sheffi (1984), Urban Transportation Networks: Equilibrium Analysis with Mathematical Programming Methods

[3] USDOT Federal Highway Administration (2011), Work Zone Road User Costs Concepts and Applications

[4] Chien and Schonfeld (2001), Optimal work zone lengths for four-lane highways. Journal of Transportation Engineering

[5] USDOT, Valuation of Travel Time in Economic Analysis-Revised Departmental Guidance, Memorandum, Office of the Secretary of Transportation, U.S. Department of Transportation, Washington, DC. 2003.

[6]LeBlanc et al (1975), An efficient approach to solving the road network equilibrium traffic assignment problem, Transportation Research

[7] U.S. Census Bureau (2013), Current Population Survey, 2011 -2013 Annual Social and Economic Supplements.