TU Delft

Delft University of Technology

Graph theory algorithms for real time control of a sewer network

van Nooijen, Ronald; Kolechkina, Alla

DOI
10.29007/6d72

Publication date
2018

Document Version
Final published version

Published in
EPiC Series in Engineering

Citation (APA)
van Nooijen, R., & Kolechkina, A. (2018). Graph theory algorithms for real time control of a sewer network. In G. La Loggia , G. Freni, V. Puleo, & M. De Marchis (Eds.), *EPiC Series in Engineering: HIC 2018. 13th International Conference on Hydroinformatics* (pp. 2127-2135). (EPiC Series in Engineering; Vol. 3). EasyChair. https://doi.org/10.29007/6d72

Important note
To cite this publication, please use the final published version (if applicable).
Please check the document version above.

Copyright
Other than for strictly personal use, it is not permitted to download, forward or distribute the text or part of it, without the consent of the author(s) and/or copyright holder(s), unless the work is under an open content license such as Creative Commons.

Takedown policy
Please contact us and provide details if you believe this document breaches copyrights.
We will remove access to the work immediately and investigate your claim.

This work is downloaded from Delft University of Technology.
For technical reasons the number of authors shown on this cover page is limited to a maximum of 10.

EPiC Series in Engineering
Volume 3, 2018, Pages 2127–2135
HIC 2018. 13th International Conference on Hydroinformatics

Graph theory algorithms for real time control of a sewer network

Ronald R.P. van Nooijen[1*] and Alla G. Kolechkina[1]
[1]Faculty of Civil Engineering and Geosciences, Delft University of Technology, Delft, Netherlands
r.r.p.vannooyen@tudelft.nl, a.g.kolechkina@tudelft.nl

Abstract

Many Dutch sewer networks are combined sewer systems, they carry both storm water and foul water. They consist of multiple sub-networks, linked by pumps into a tree structure with the Waste Water Treatment Plant as its root. Within sub-networks sewage transport is by gravity driven flow. Usually the original design assumed local control. Later changes, additions and extensions sometimes reduced the effectiveness of the original design. In these cases central control can improve the performance of the system without costly new construction. We apply two algorithms from graph theory, one is based on stable flows in time, the other on quickest evacuation flows. Results on local control are included to provide a lower bound on performance. A linear programming problem based of a perfect forecast of the whole event provides an upper bound on performance.

1 Introduction

Combined sewer systems form an important and vulnerable part of the urban drainage system. In deltas, such as the coastal provinces of the Netherlands, pump stations are essential components of the system, because the flat terrain and the high groundwater level leave little room for gravity driven flow. In these systems a pipe may do double duty: it collects waste water and run-off, and it transports sewage from other parts of the network onwards in the direction of the Waste Water Treatment Plant (WWTP) [1, 2]. We will write "system" for the network of sewer pipes and pump stations and "district" for a part of the pipe network where flow is gravity driven. Traditionally the pumps in these systems are under local control. A Real Time Control (RTC) scheme based on a central controller can make more efficient use of the storage capacity and the pump capacity in the system. This can lower the risk of streets being flooded and it can reduce the total volume of Combined Sewer Overflow (CSO). The districts that compose a typical Dutch system are relatively small. The area connected to

* Corresponding author

one WWTP, which processes sewage from several different systems, tends to be much larger; this means that high spatial and temporal accuracy would be needed to profitably use weather forecasts. In this paper we consider two graph theory algorithms that can operate without a forecast. One of the algorithms is based on stable flows in time, the other on quickest evacuation flows. We compare the results with a solution obtained by solving a Linear Programming (LP) problem that uses a perfect prediction of the precipitation. Another application of an algorithm from graph theory to sewer system control can be found in [3].

2 Problem statement

There are still many combined sewer systems in use all over the world. These have in general grown in size together with the communities they serve. Consequently these systems are composed of sub-systems developed at different times and with different design methods. This growth has introduced inconsistencies in local storage capacity and transport capacity. Moreover it has resulted in systems where spatial inhomogeneity in precipitation can lead to a load imbalance. When combined with the effects of climate change an increase in CSO volume is often the result. This leads to water quality problems in the receiving water and which in turn creates a need for improvements in both the physical system and the control scheme. If we knew in advance when and where the water entered the system, then we could determine whether or not the network could transport the total inflow and if it could, how it could do so. If we do not know the inflow in advance then the best we can do is to move the water in the system out of the system as rapidly as possible. Preferably whilst also preventing CSO in vulnerable locations. In this paper we present two algorithms, one based on stable flows in time, and one based on quickest evacuation flows, that should be able to do this. The flows calculated by these network algorithms will contain the information needed to control the pumps in the sewer system. A standard Supervisory control And Data Acquisition (SCADA) system can be used to gather input data on stored volumes for the algorithms and transmit the resulting commands to the pumps. We refer to such an approach as “central control” as opposed to “local control” where a local level measurement controls each pump. Algorithm performance is tested on a simplified computer model of a sewer system. As precipitation forecasts of sufficient accuracy are hard to come by, the tests are run without such predictions, but both algorithms can use precipitation or inflow predictions. We considered the sewer system “KW” with five districts. Details are given in Table 1, more information can be found in [2, 4].

quantity	unit	KW-1	KW-2	KW-3	KW-8	KW-1
Storage	m^3	1657	218	270	84	303
Pump cap.	m^3/s	0.0375	0.0236	0.0638	0.0038	0.0319
Area	$10^4 m^2$	17.62	3.07	2.18	2.63	2.42
Pumps to	-	KW-4	KW-4	WWTP	KW-2	KW-4

Table 1: The KW network

The corresponding directed graph is shown in Figure 1. A similar larger system is described in [1].

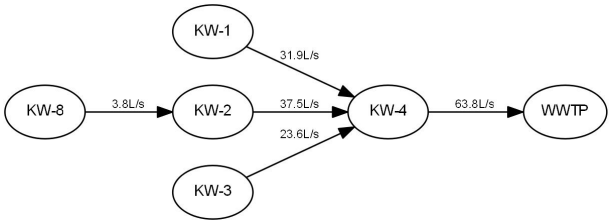

Figure 1: Directed graph with districts as vertices and pump stations as edges

3 Some graph and network theory

Some basic definitions from graph and network theory are needed to describe the algorithms in use. A directed graph consists of a set of vertices V and a set of directed arcs A. Each arc starts in one vertex (called the tail) and ends in another vertex (called the head) [5-7]. A network is a directed graph, where there are two disjoint subsets of vertices, the sources S, and the sinks T. Moreover, there is a function b that assigns a transport capacity to each arc [6, 7]. A flow f on a network is a function that assigns a transport flow to each arc in such a way that the transport capacity is not exceeded, and that the transported quantity is conserved in all vertices that are not sources or sinks. Sources can supply unlimited flow and sinks can absorb unlimited flow. In the case of sewer systems it is desirable to add a time dimension. Usually the time is taken as discrete. To do so a transit time is assigned to each arc. A flow over time is then defined as a function that assigns flows to each arc at each point in time such that all capacity constraints are satisfied at all times, and flow conservation holds in all vertices that are not sources or sinks. For a finite time extent such a problem can be translated back into a network flow problem by creating the "time expanded graph" with copies of the original graph vertices at all points in time. The vertices in the expanded graph are labelled with ordered pairs (v,t) of the vertex v in the original graph and the point in time t. Next arcs are added in such a way that there is an arc a' from (v_1,t_1) to (v_2,t_2) with capacity b' if and only if the original graph contained an arc a rom v_1 to v_2 with travel time t_2-t_1 and capacity $b(a) = b'$. Such a time expanded network can include storage by adding arcs from a vertex to itself at the next time step with a capacity equal to the available storage capacity at that vertex.

3.1 Quickest evacuation flows

There is a class of problems in graph and network theory called "quickest evacuation flows" or "earliest arrival flows". These are studied in the context of evacuations of buildings. One way to look at the problem of setting pump flow rates is to consider it as an "quickest evacuation" problem: in other words, how can we remove the sewage currently present in the system as quickly as possible. There is considerable literature on this topic [7-16]. We construct an algorithm that empties the districts as quickly as possible in a given order. To do so we make use of properties of maximal flows on networks described in [10]. For this we first need to construct a network on which the desired solution is a maximal flow. To avoid problems with floating point values we internally use 64-bit integers for the calculation of flows and storages. If we express flows in litres per time step and volumes in litres, then we still get a reasonably accurate solution. The network is constructed as follows. We start with one source vertex for each district, then add one terminal vertex. For each time step we add a distinct vertex corresponding to each district and a vertex corresponding to the WWTP. We then add an arc from the WWTP to the sink with a capacity equal to the volume in litres, that can be processed by the WWTP in one time step. Next we add an arc running from each district vertex to the corresponding vertex at the next time step with a capacity equal to the local storage in litres to

represent water stored at the node. If an inflow prediction is available, then we also add an arc entering the vertex from the source with a capacity equal to the predicted inflow volume over the time step. Finally, we add arcs corresponding to the pump stations between districts of different time steps. The algorithm proposed in [11] will now derive a flow that at each time step has the largest possible total cumulative outflow from the vertex with the highest priority. For the vertex with the second highest priority it has the largest possible total cumulative outflow that can be realized with the remaining unused capacity. This pattern continues on for the other vertices.

3.2 Stable flows

Setting pump flow rates can be based on the concept of "stable flows", this is an extension of the classical "stable marriage problem". Stable flows over time are introduced in [17]. If we have an order in which the districts need to be emptied, then we can derive a "stable flow over time" problem formulation. This algorithm could in principle work using only communication with neighbouring districts, but this would entail considerable communication overhead during algorithm execution. The concept of a stable flow over time is based on the concept of "stable flow", which will be described first. A stable flow problem has the following characteristics. We start with a network. For all vertices v that are not sources or sinks we have a list $L_{\text{in}}(v)$ of incoming arcs in order of preference from most to least preferred, and a list $L_{\text{out}}(v)$ of outgoing arcs in order of preference, from most to least preferred. A stable flow is characterised by the property that there are no two vertices that could take action to increase the flow between
them in such a way that both would be acting more in line with their preferences. This is formalized in the definition of a blocking walk.

A *blocking walk* for a given flow f on a network is a directed walk for which all the following statements are true:

- each arc in the walk is carrying less flow than its maximum capacity;
- the directed walk starts in the set S on the network or the first arc in the walk is in use while the first vertex still has unused capacity on a more preferred outgoing arc;
- the directed walk ends in the set T of the network or the last arc in the walk is in use while the last vertex still has unused capacity on a more preferred incoming arc.

Given the definition of a blocking walk, we can now define a *stable flow* by the absence of blocking walks.

In sewer systems we need the time dimension. In [17] the stable flow problem is extended to flows over time by constructing the time expanded graph and considering stable flows on that graph. We will construct a network such that a stable flow over time on that network corresponds to pump settings that transport sewage from each district to the WWTP, while taking into account preferences for accepting inflow and dispatching outflow for the individual districts. We could, for example, order acceptance from upstream districts according to the vulnerability of those districts to a local CSO and do the same for dispatch to downstream districts.

To create such a network and get a working algorithm some auxiliary vertices are needed. We will describe the time expanded network as this allows us to accommodate the case where the travel time for the purposes of mass conservation is zero. As a basis we take a network with a vertex v_j for each district j, a vertex w for the WWTP, and an arc a_k for each pump station k.

We have one source vertex σ and one terminal vertex τ. We will have N time steps in our expanded network. For each time step we have a vertex (w,t) with an outgoing arc to τ with capacity equal to the WWTP plant capacity. For each district j and each time step t we have a vertex labelled (j,t), an auxiliary vertex $v_{j,t}^{\text{spill}}$, and an auxiliary vertex $v_{j,t}^{\text{store}}$. The vertex $v_{j,t}^{\text{spill}}$ has one incoming arc from σ with a capacity equal to the precipitation volume for the next time step and two outgoing arcs with

infinite capacity: one to (j,t) and one to τ. The one to (j,t) is preferred. The vertex (j,t) has the following incoming arcs, given in order of preference:

- an arc from $v_{j,t}^{\text{spill}}$ with infinite capacity;
- an arc from $v_{j,t}^{\text{store}}$ with capacity equal to the storage capacity;
- arcs from upstream districts ordered from most vulnerable to CSO to least vulnerable to CSO with capacity equal to pump station capacity.

It has the following outgoing arcs, again given in order of preference:

- to the WWTP if there is an arc to the WWTP in the original network with capacity equal to the arc in the original network;
- to downstream districts ordered from least vulnerable to CSO to most vulnerable to CSO with capacity equal to pump station capacity;
- to $v_{j,t}^{\text{store}}$ with capacity equal to district storage capacity.

The vertex $v_{j,t}^{\text{store}}$ has two incoming arcs, given in order of preference:

- from (j,t) with capacity equal to the district storage capacity;
- from (j,t) with capacity equal to district storage capacity.

It also has two outgoing arcs, given in order of preference:

- (j,t) with capacity equal to district storage capacity;
- $v_{j,t+1}^{\text{store}}$ with capacity equal to district storage capacity.

The algorithm can be seen as a multiple round negotiation between vertices. Vertices initially push the maximum available flow along their most preferred outgoing arc. Next they accept or reject (part of) the flow on their incoming arcs. This continues until a stable flow has been reached. With the network and priorities given above, this will result in as large a flow as possible to the WWTP while respecting pump capacities and WWTP capacity. CSO may occur, but will occur first in the less vulnerable districts. A flow can be found using the algorithm described in [17].

4 Materials and methods

To evaluate the results of the proposed algorithms we compare them to two reference cases. We use the traditional local control method as a lower bound on performance and to the results obtained with linear programming using full knowledge of the precipitation as an upper bound on performance. The graph theoretical algorithms were run without inflow predictions.

4.1 Inflow generation

We use selected events from a time series of precipitation data from “de Bilt” in the Netherlands with a 15 minute time step according to the method from [18]. The data was obtained from the “Koninklijk Nederlands Meteorologisch Instituut” (KNMI). The precipitation data was split into events by looking for dry periods long enough to empty the system. Events where the peak intensity results in inflow that the sustained throughput capacity of the system are disregarded. Precipitation is translated directly into run-off for each district by multiplying by the connected area.

4.2 The model used for the sewer networks

For the purpose of this paper we take the simplest possible model of a sewer system. The model has the structure of a directed graph. The districts with gravity driven flow are modelled as reservoirs [1]. This type of model goes back to [19, 20]. These will be the vertices in our graph together with the WWTP. The pumping stations will be the arcs. This corresponds best to the behaviour of Dutch

combined sewer systems, where there is a large amount of excess storage in the pipes of the sewer system itself.

4.3 Traditional local control

Local control was implemented switching the pump "on", when a volume corresponding to five minutes or more worth of pumping at full capacity is present, and "off", when the volume reaches zero. The results are included for two reasons. Firstly, they provide a lower performance bound as a reference. The performance of any central scheme should on average be better [18]. Secondly, a system that is specifically designed for local control will, in the case of uniform precipitation over the whole system, perform as well as any central system [21]. Always provided that only the total CSO volume is relevant, in other words, that the distribution CSO in space and time plays no role. In traditional local control the pumps are either "on" or "off".

4.4 Linear programming used as "best case" control solution

To establish a reference for each event LP was applied to each event to see if there was a feasible solution without CSO. If there was no solution then a second LP problem was solved to find the solution corresponding to minimum total CSO volume for that event. The solution obtained is "best case" provided that only the total CSO volume is relevant. The purely local solution and the LP solution provide reference bounds on the total CSO volume associated with a given event for other control algorithms. In our LP solution we allow the pumps to run at any flow rate between zero and full capacity, provided there is a sufficient supply of sewage.

4.5 Setup of computer experiments

Several computer experiments were carried out. The node ordering KW-8, KW-2, KW-4, KW-3, KW-1 was used for the quickest evacuation flow (QE) algorithm. For the stable flow (SF) algorithm the preference order for the inflows into KW-4 was KW-1, KW-2, KW-3. The number of time steps used in the expanded network was chosen based on preliminary investigation of the relative improvement when using additional time steps.

5 Results

We split the events with non-zero CSO volume into three groups. Results for events with total CSO volume up to 150m^3 are shown in Figure 2, events with total CSO volume from150 m^3 to 400 m^3 are shown in Figure 3, and events with CSO volume exceeding 400 m^3 are shown in 4. The SF algorithm was run with 10 time step deep network and the QE algorithm with a 20 time step deep network. In the legends LC refers to local control.

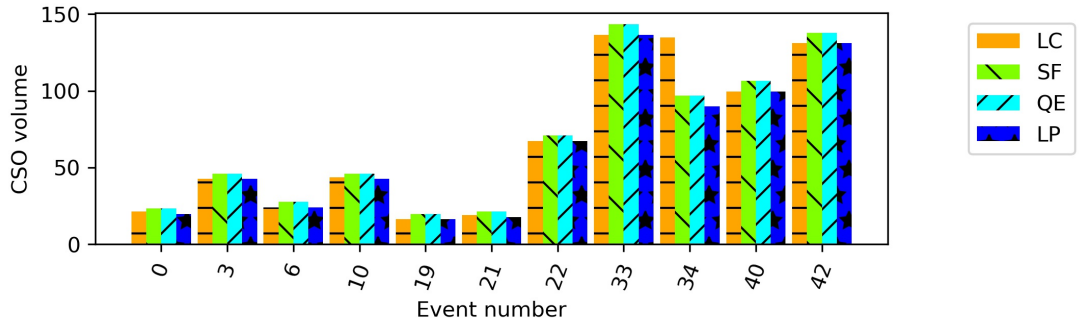

Figure 2: Events with CSO volume up to 150 m^3

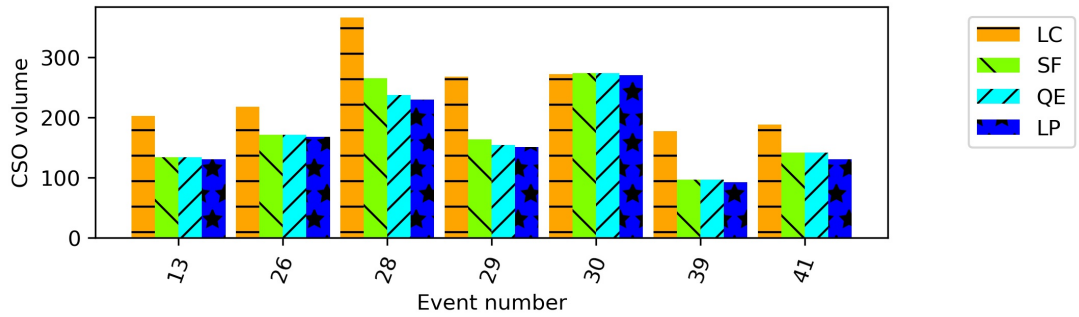

Figure 3: Events with CSO volume from 150 m^3 to 400 m^3

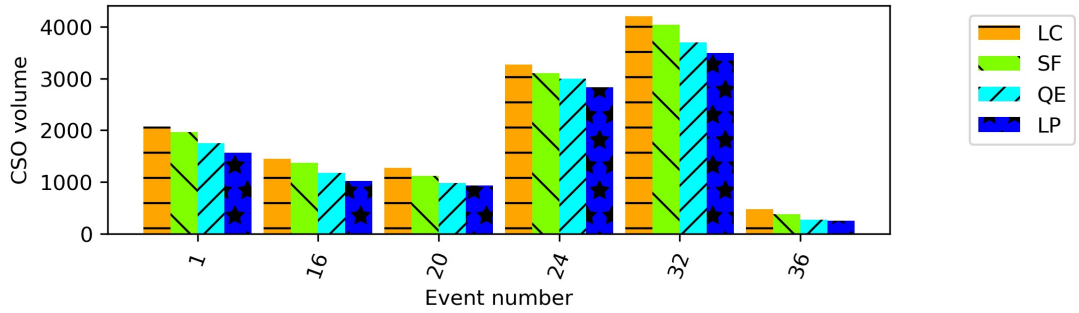

Figure 4: Events with CSO volume exceeding 400 m^3

The code for both controllers was written for transparency, not speed. Nevertheless we provide the following timing information. On a 3.5GHz Xeon processor one execution of SF took approximately two seconds and one execution of QE took approximately half a second.

6 Discussion

Clearly more research is needed, but the results obtained so far are encouraging. While for small CSO volumes SF and QE perform slightly worse than local control, comparison of local control and

LP shows that in these cases there is almost no room for improvement. The biggest improvements are visible for events that cause CSO volumes between 150 m^3 and 400 m^3, where QE without an inflow forecast is close to LP with a perfect forecast. For large CSO volumes SF and QE also perform somewhat better than local control. In all cases QE performs as well or better than SF. In the expanded network form both algorithms are relatively easy to explain to non-experts. Further study into ways to dynamically assign priorities is planned. Extensions of the QE algorithm are available for time varying networks and for networks with varying time delays per arc. This would allow for application to models of sewer systems where there are time delays associated with transport. Further investigations are needed to determine the effect of the choice of priorities, the size of the system, and the number of time steps in the extended network for both methods.

References

[1] R.R. van Nooijen and A. Kolechkina, Speed of discrete optimization solvers for real time sewer control, Urban Water Journal, 10:5 (2013) 354-363.
[2] R.R.P. van Nooijen, et al., Tuning of a central controller for a sewer network using multiple simplified models, in: Proceedings of the 9th International Conference on Urban Drainage Modelling (UDM 2012), Faculty of Civil Engineering, University of Belgrade, Belgrade, Serbia, 2012, pp. 1-9.
[3] K. Neugebauer, W. Schilling, and J. Weiss, A Network Algorithm for the Optimum Operation of Urban Drainage Systems, Water Science & Technology, 24:6 (1991) 209-216.
[4] R.R.P. van Nooijen, et al., Implementation of central control for multiple sewer systems, in: Proceedings of the 12th IWA/IAHR International Conference on Urban Drainage, Porto Alegre, Brazil, 2011, pp. 10-15.
[5] S. Arora and B. Barak, Computational complexity, Cambridge University Press, Cambridge, 2009.
[6] D. Jungnickel, Graphs, Networks and Algorithms, Springer, 2013.
[7] L.R. Ford, Jr. and D.R. Fulkerson, Flows in networks, Princeton University Press, Princeton, N.J., 1962.
[8] L.R. Ford and D.R. Fulkerson, Constructing Maximal Dynamic Flows from Static Flows, Operations Research, 6:3 (1958) 419-433.
[9] D. Gale, Transient flows in networks, Michigan Math. J., 6 (1959) 59-63.
[10] W.L. Wilkinson, An algorithm for universal maximal dynamic flows in a network, Operations Res., 19 (1971) 1602-1612.
[11] E. Minieka, Maximal, lexicographic, and dynamic network flows, Operations Res., 21 (1973) 517-527.
[12] R.G. Ogier, Minimum-delay routing in continuous-time dynamic networks with piecewise-constant capacities, Networks, 18:4 (1988) 303-318.
[13] J.E. Aronson, A survey of dynamic network flows, Ann. Oper. Res., 20:1-4 (1989) 1-66.
[14] M. Skutella, An introduction to network flows over time, in: Research trends in combinatorial optimization, Springer, Berlin, 2009, pp. 451-482.
[15] M. Schmidt and M. Skutella, Earliest arrival flows in networks with multiple sinks, Discrete Applied Mathematics, 164 (2014) 320-327.
[16] S. Göttlich, et al., Evacuation modeling: a case study on linear and nonlinear network flow models, EURO Journal on Computational Optimization, 4:3-4 (2015) 219-239.
[17] Á. Cseh, J. Matuschke, and M. Skutella, Stable Flows over Time, Algorithms, 6:3 (2013) 532-545.

[18] R. van Nooijen, A. Kolechkina, and E. van Leeuwen, Input Selection for Sewer Network Flow Controller Evaluation, in: New Aspects of Automatic Control, Modelling & Simulation, 12th WSEAS Int. Conf. on Automatic Control, Modelling \& Simulation, EUROPMENT Press, Sofia, Bulgaria, 2010, pp. 53-58.

[19] M. Papageorgiou, Automatic Control Strategies for Combined Sewer Systems, Journal of Environmental Engineering, 109:6 (1983) 1385-1402.

[20] J.W. Labadie, N.S. Grigg, and P.D. Trotta, Minimization of combined sewer overflows by large-scale mathematical programming, Computers & Operations Research, 1:3-4 (1974) 421-435.

[21] R.R.P. van Nooijen and A.G. Kolechkina, Room for automatic control in combined sewer systems, in: Proceedings of the 19th IFAC World Congress, Cape Town, South Africa, 2014, pp. 5315-5320.