An ASP Framework for Efficient Urban Traffic Optimization

An ASP Framework for Efficient Urban Traffic Optimization

Matteo Cardellini
DIBRIS, Università degli Studi di Genova
Genova, Italy
Politecnico di Torino*
Torino, Italy
matteo.cardellini@polito.it

Avoiding congestion and controlling traffic in urban scenarios is becoming nowadays of paramount importance due to the rapid growth of our cities' population and vehicles. The effective control of urban traffic as a means to mitigate congestion can be beneficial in an economic, environmental and health way. In this paper, a framework which allows to efficiently simulate and optimize traffic flow in a large roads' network with hundreds of vehicles is presented. The framework leverages on an Answer Set Programming (ASP) encoding to formally describe the movements of vehicles inside a network. Taking advantage of the ability to specify optimization constraints in ASP and the off-the-shelf solver CLINGO, it is then possible to optimize the routes of vehicles inside the network to reduce a range of relevant metrics (e.g., travel times or emissions). Finally, an analysis on real-world traffic data is performed, utilizing the state-of-the-art Urban Mobility Simulator (SUMO) to keep track of the state of the network, test the correctness of the solution and to prove the efficiency and capabilities of the presented solution.

1 Introduction

Problem Description. At the end of the 21st century, the world population is expected to increase to 10.9 Billion, adding more than 3 Billion people to the current population [19]. This huge growth, which will directly translate in a higher number of vehicles roaming the streets of our cities, demands improvements in the transport infrastructure and a better utilization of our roads for the purpose of avoiding congesting the network. Traffic jams have a negative impact on safety and fuel consumption, which directly translates to a higher cost for drivers and health issues for residents near highly trafficked roads, caused by bad air quality and noise pollution [26].

Overview of existing literature. One of the most common methods in the literature to optimize traffic flow in road networks is to schedule traffic light switching phases, or *signal phase plans (SPPs)*, with the aim of minimizing delay and avoiding wasting time at intersections [22, 12]. The well-known real time adaptive traffic control system SCOOT [4], for example, makes use of this methodology and is now used extensively throughout the United Kingdom. Unfortunately, managing only the switching phases of traffic lights has some limitations: the only metric which is controllable and optimizable is the waiting time at intersections (which has a direct impact on the total travel time of vehicles) but is not straightforwardly expandable to consider other metrics (i.e., pollution, risk, fuel consumption, etc). Moreover, the (*macroscopic*) point of view of traffic lights, which manages the flow of traffic modelling incoming and outgoing lanes as queues of vehicles, does not allow for a more detailed (*microscopic*) consideration of the single vehicles and their routes inside the network. Microscopic simulation models

*The author is a PhD Student at the Italian National PhD Programme in Artificial Intelligence https://www.phd-ai.it/

Y. Lierler, J.F. Morales, C. Dodaro, V. Dahl, M. Gebser, T. Tekle (Eds.):
International Conference on Logic Programming 2022 (ICLP'22).
EPTCS 364, 2022, pp. 217–227, doi:10.4204/EPTCS.364.37

have been largely discarded in the literature due to their high complexity and low scalability. In this paper, we leverage state-of-the-art Artificial Intelligence techniques, coupled with domain-dependent optimizations, to model the traffic flow of hundreds of vehicles inside large European cities from a *microscopic* point of view.

The use of Artificial Intelligence techniques in road transportation has already been found to be efficient in optimizing traffic flow [1, 20]. For instance, [25] introduced a *mixed discrete-continuous planning* [14] approach for reducing congestion through a *macroscopic* point of view. In [9] instead, a *temporal planning* approach was used for managing traffic, now through a *microscopic* prospective, with the intention of reducing air pollution and respect air quality limitations. Other instances of urban traffic problems solved with automated planning technologies can be found in [8]. Even if automated planning has been beneficial in efficiently solving several real-world problems in transportation [6, 23], the main point of failure is the ability to scale in the presence of large number of vehicles.

Goal of the research. Arguably, the purpose of optimizing the flow of traffic inside a road network lies not in finding the best possible route for every vehicle, but instead in finding the best combination (schedule) of routes for all the vehicles in the network. For this reason, the approach presented in this paper relies mainly on an Answer Set Programming (ASP) [17, 21, 3] encoding of the problem. ASP has already proved to be an effective tool for solving several practical scheduling problems [11, 10, 7]. In particular, in [13], an ASP based solution was presented for modelling an abstract *mesoscopic* flow model (a middle-ground approach between *macroscopic* and *microscopic*) and a strategy for generating traffic lights SPPs. In this paper, the ASP encoding will have the goal to find the best route (according to some metric) for every new vehicle that enters the network using a relaxed version of the network and traffic rules. After computing the best route in the relaxed system, the real metrics and performance are computed using the microscopic traffic simulation tool SUMO [18] which will instead consider all the difficulties of the real system. Our proposed framework will behave as a *Centralized Urban Traffic Controller* (CUTC): we will suppose that the centralized controller knows the position of every vehicle inside the network and is tasked to find the best possible route (according to some metric) for new vehicles which enter the network.

Structure of the paper. Section 2 discusses the architecture of the proposed framework, the domain specific optimizations which allow the system to scale to manage a high number of vehicles, and the ASP encoding which is used under the hood to schedule the routes inside the network. Section 3 shows how the proposed framework compares with other approaches and how the system scales with respect to the number of vehicles which are inside the network. Section 4 closes the paper by discussing open issues of the framework and how these can be tackled in future work.

2 Current status of the research

2.1 Architecture

In the last decade, driven by the concept of *smart cities*, several communications systems between vehicles and traffic components (i.e., traffic lights, traffic controllers, etc) have been developed [24]. As previously stated, the proposed framework purpose is to act as a CUTC, which has knowledge of the position of vehicles inside the network and is tasked to find an optimal route (according to some metric)

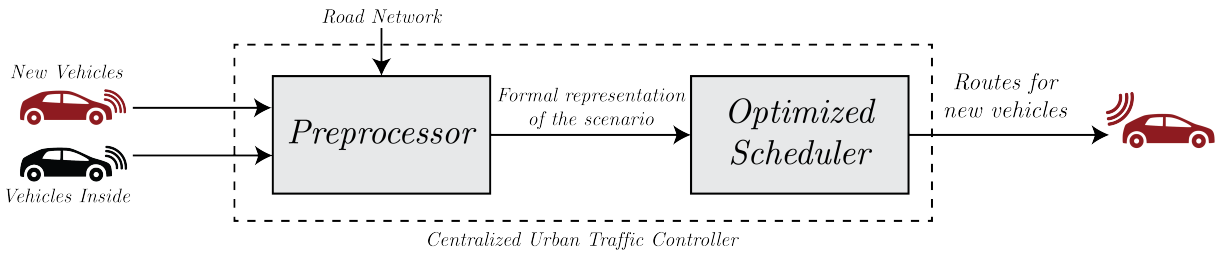

Figure 1: Architecture of the proposed framework

for new vehicles which approach the network. In the proposed framework, the system is able to communicate with vehicles through *Vehicular Ad Hoc Networks* (VANETs) and fetch data about vehicles (i.e., their position and route inside the network). Figure 1 shows the architecture for the proposed framework, which is composed of the following components:

- a *Preprocessor* is dedicated to build an internal model of the road network, abstracting part of it (i.e., joining small streets together and easing intersections in roundabouts) and to compute preliminary results (i.e., list of possible routes for every vehicle, street's enter and exit times ranges for every new vehicle) in order to guide the computation and avoiding having the *Optimized Scheduler* searching solutions which are already known to be unpromising. This component takes as input the network structure, the list of incoming new vehicles and the position of all the vehicles which have already a route inside the network (through VANETs). The preprocessor outputs then a simplified formal representation describing the network, the possible routes of new vehicles and the status of the vehicles inside the network.
- an *Optimized Scheduler* receives a formal representation of the scenario, which has been constructed by the preprocessor, and finds an optimal route for each vehicle which is approaching the network. These optimized routes are then communicated to the approaching vehicles through the VANETs.

More in depth, in the proposed approach the formal representation of the traffic scenario is specified using an Answer Set Programming encoding and the *Optimized Scheduler* is a wrapper of the general purpose solver CLINGO [15].

2.2 Domain Specific Optimizations

Before describing the ASP Formulation of the encoding, it is important to elaborate on how the *Preprocessor* reasons upon the network and the vehicles that occupy it as a means of simplifying the solving process, excluding solutions which are known beforehand to be infeasible, or not optimal:

- In the proposed architecture, the origin and destination of each vehicle which approaches the network is known a priori. For this reason, it is possible to compute beforehand, for all the entering vehicles, all the possible (acyclic) paths in the graph network which connects their source and target streets. In large and complicate maps, this would result in hundreds of possible routes. For this reason, the *preprocessor* groups similar routes together and then takes a couple from the shortest of each group. This will allow the scheduler to have the most different (shortest) routes in order to deal with traffic.
- Since the scheduling has to be time-dependent, it is important to discretize the time in steps which are sufficiently small to capture real traffic nuances but are large enough to still be able to schedule in a reasonable time. In our experiment, the discretization step was chosen to be of 5s.

- A single roundabout is composed of several very small streets which connect all the intersections of incoming and outgoing streets. The small streets result in a high penalty for vehicles that need to cross the roundabout, since every small street of the roundabout must be run in the discretization step ($5s$). For this reason, the preprocessor joins together all the streets which connects all incoming and outgoing streets of the roundabout, creating longer streets which do not penalize crossing the roundabout so heavily. Since streets are now grouped together, a single street is accounted multiple times in the network, and it is up to the scheduler to respect the total capacity of the roundabout.
- Knowing all the routes of vehicles inside, and their expected position at every point in time (obtained from the scheduler when the vehicles were approaching the network), it is possible, for every route that a new vehicle could run, to compute beforehand a range on the timings of entrance and exit in every street of the route. Intuitively, the minimum time in which a vehicle enters and exits a street in a route can be computed as if all traffic were removed, meaning the vehicle is not slowed down and the streets are run at maximum speed. A maximum exit time, instead, can be computed by analysing how many vehicles would be present in the street at its minimum entry time: (i) if the vehicles inside are less than the street capacity then we can consider the speed inside the street to be inversely proportional to the number of vehicles in the street, (ii) if the vehicles congest the street than we can compute a maximum exit time by considering how much it will take for the queue to move to the controlled vehicle.

2.3 ASP Formulation

The main core of the proposed architecture lies in the ASP formulation of the traffic scenario. Here, the ASP encoding is presented, based on the input language of CLINGO. In the following, we assume the reader is familiar with syntax and semantics of ASP; for details about syntax and semantics of ASP programs, the reader is referred to [5]. Firstly, the network-specific optimized data model produced by the preprocessor is introduced, afterwords the static encoding which models the behaviour of traffic flows inside the map will be discussed.

Data Model. The data model is composed of the following atoms:

- `streetOnRoute(S,R,MIN,MAX)` which models the fact that street `S` composes route `R`. `MIN` and `MAX` represent the minimum and maximum time a vehicle, which starts to run through route `R` at $t = 0$, is expected to enter street `S`.
- `link(S1,S2)` specifies that it is possible for a vehicle to move from street `S1` to `S2`,
- `vehicle(V,T)` defines the presence in the map of a vehicle `V` of type `T`, which can be `1` if the vehicle is *controlled*, meaning that it is a new vehicle for which a route has yet to be found, or `0` if the vehicle is *simulated*, meaning that a route is already been set and the system need only to keep track of their presence in the map.
- `origin(V,FROM)` designates `FROM` as the street the vehicle `V` is at the time of the planning. If a vehicle is *controlled*, the origin coincides with the first street the vehicle will step into when entering the map. Similarly, `destination(V,TO)` specifies that `TO` is the final street which will bring the vehicle `V` outside the map.
- `possibleRouteOfVehicle(V,R)` signals that a vehicle `V` in order to move from its origin to its destination can follow a route `R`. If the vehicle is *simulated* then it will only have one possible route

```
1  1 {solutionRoute(V, R): possibleRouteOfVehicle(V, R)} 1 :- vehicle(V, 1).
2  solutionRoute(V, R) :- possibleRouteOfVehicle(V, R), vehicle(V, 0).
3  solutionStreet(V, S) :- solutionRoute(V,R), streetOnRoute(S, R,_,_).
4  1 {enter(V,S,T) : time(T), T >= MIN, T <= MAX} 1 :- vehicle(V, 1), solutionStreet(V, S), solutionRoute(V,
       R), streetOnRoute(S, R, MIN, MAX), not origin(V,S).
5  enter(V,S,0) :- origin(V,S).
6  1 {exit(V,S,T) : time(T), T > IN, T <= IN + MAX} 1 :- vehicle(V, 1), enter(V,S,IN),
       maxTrafficTravelTime(S,MAX).
7  nVehicleOnStreet(S,T,N) :- enter(_,S,T), N = #sum{1,V: enter(V,S,IN), IN <= T; -1,V: exit(V,S,OUT), OUT
       <= T}.
8  travelTime(S,T,X) :- enter(_,S,T), nVehicleOnStreet(S,T,N), heavyTrafficThreshold(S,A,_), N >= A,
       heavyTrafficTravelTime(S,X).
9  travelTime(S,T,X) :- enter(_,S,T), nVehicleOnStreet(S,T,N), mediumTrafficThreshold(S,A,B), N >= A, N < B,
       mediumTrafficTravelTime(S,X).
10 travelTime(S,T,X) :- enter(_,S,T), nVehicleOnStreet(S,T,N), lightTrafficThreshold(S,_,B), N < B,
       lightTrafficTravelTime(S,X).
11 :- vehicle(V,1), exit(V,S,OUT), enter(V,S,IN), travelTime(S,IN,X), OUT < IN + X.
12 :- vehicle(V,1), exit(V,S1,OUT1), enter(V,S2,IN2), link(S1,S2), IN2 != OUT1.
13 :- enter(V,S,T), vehicle(V,1), capacity(S,MAX), nVehicleOnStreet(S,T,N), N > MAX.
14 :- enter(V,SR,T), streetInRoundabout(SR,R), vehicle(V,_), roundabout(R,MAX), #sum{X,S:
       nVehicleOnStreet(S,T,X), streetInRoundabout(S,R)} = N, N > MAX.
15 :~ solutionRoute(V, R), vehicle(V,1), cost(V,R,N). [N@2, V, R]
16 :~ nVehicleOnStreet(S,T,N). [N@2,S,T]
```

Figure 2: ASP Encoding used in the optimized scheduler

available, which is the one found when the vehicle entered the network, conversely if a vehicle is *controlled* this atom will specify a subset of all the possible routes from origin to destination, as discussed in the previous section.

- `time(T)` specifies the time unit of the scheduling, ranging from 0 to the maximum horizon in which all vehicles have left the street. As previously stated, in the proposed approach the time step has been chosen to be of $5s$.
- `capacity(S,N)`, `heavyTrafficTravelTime(S,T)`, `mediumTrafficTravelTime(S,T)`, and `lightTrafficTravelTime(S,T)` indicate respectively the capacity of street `S` and the times needed to run the street in cases of heavy traffic (15 km/h), medium traffic (30 km/h), low traffic (45 km/h) based on the length of the street. `maxTrafficTravelTime(S,T)`, similarly, models the time it would take to free the street in cases of congestion in which the street is at it maximum capacity.
- `heavyTrafficThreshold(S, MIN, MAX)`, `mediumTrafficThreshold(S, MIN, MAX)` and the atom `lightTrafficThreshold(S, MIN, MAX)` specify the range $(\texttt{MIN},\texttt{MAX}]$ of vehicles in street `S` which characterize respectively heavy, medium and low traffic of the previous point.
- `enter(V,S,IN)` and `exit(V,S,OUT)` specify that the *simulated* vehicle `V` will enter and exit street `S` at time `IN` and `OUT`, respectively. These atoms are constructed from the scheduling solution found previously.
- `roundabout(R, C)` specifies the existence of a roundabout `R` with maximum capacity (in all its streets) of `C`. `streetInRoundabout(SS,R)` indicates that the (simplified) street `SS` belongs to the roundabout `R`.
- `cost(R,V,N)` specifies that running on route `R` for vehicle `V` has a cost of `N`.

Rules. Figure 2 lists the ASP encoding used in the optimized scheduler. Rule r_1 defines the atom `solutionRoute`, which, for every *controlled* vehicle, chooses a single route between all the most different shortest routes which move the vehicle from its origin to its destination, as computed by the preprocessor. Rule r_2, instead, imposes the `solutionRoute` for *simulated* vehicles equal to the route they are already running. Rule r_3 defines the atom `solutionStreet(V,S)` which signals that the vehicle `V` will run through street `S` in the solution. Rule r_4 is used to compute an entry time for every *controlled* vehicle in the minimum and maximum range computed in the preprocessor; for the first origin street, the entry time is imposed to zero with Rule r_5. Similarly, rule r_6 defines the exit time of vehicles for every street in their route. Rule r_7 defines the atom `nVehicleOnStreet(S,T,N)` which counts the number of vehicles on street `S` at time `T`. This atom is then used in rules r_8 trough r_{10} to compute the atom `travelTime(S,T,X)` which represents the time (`X`) a vehicle which enters the street `S` at time `T` would take to run the whole length of the street in situations of, respectively, heavy, medium or light traffic. Constraint r_{11} imposes that when a vehicle enters a street, it cannot leave it before the amount of time specified in the atom `travelTime`; this defines only a lower-bound, since the vehicle could remain congested and leave much after. Constraint r_{12} imposes an order between streets in a route. Constraints r_{13} and r_{14} force the vehicle to respect the capacities of streets and roundabouts, respectively. The weak constraint r_{15} optimizes the cost of each route for *controlled* vehicles. If two solution produces the same cost, weak constraint r_{16} breaks the tie by preferring the scheduling which allows vehicles to better spread along the network.

3 Preliminary Results

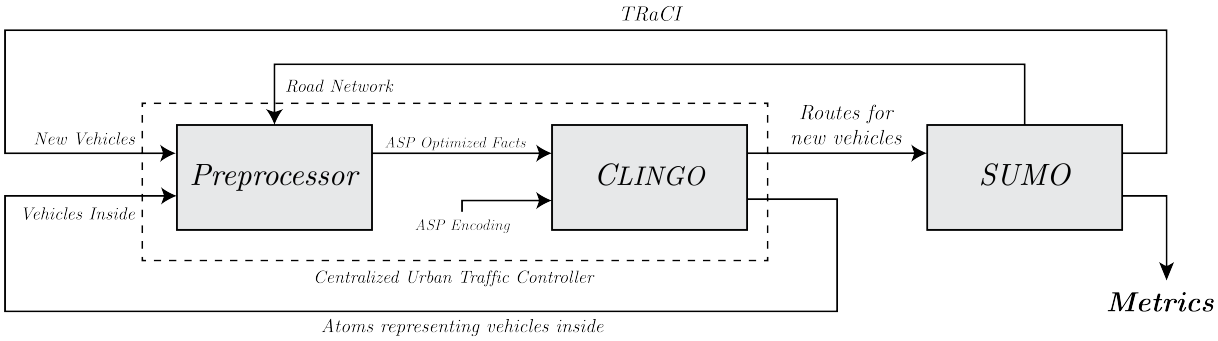

Figure 3: The architecture of the experiments.

Architecture of the experiments. In order to test the correctness and performance of the proposed approach, it is of paramount importance to connect the system to a real time traffic simulator which can capture all the nuances of traffic flow which are not covered in the modelling (i.e., traffic lights, rights of way, overtakes, etc). Figure 3 shows how the CUTC was wired to the traffic simulator SUMO. The traffic control interface TRaCi [27] connects SUMO with the Preprocessor to fetch a representation of the road network and to get notified when new vehicles approach the network and optimized routes needs to be computed; the CLINGO wrapper, after computing the optimal routes of approaching vehicles, returns to the preprocessor a list of atoms representing the expected positions of vehicles inside the network,

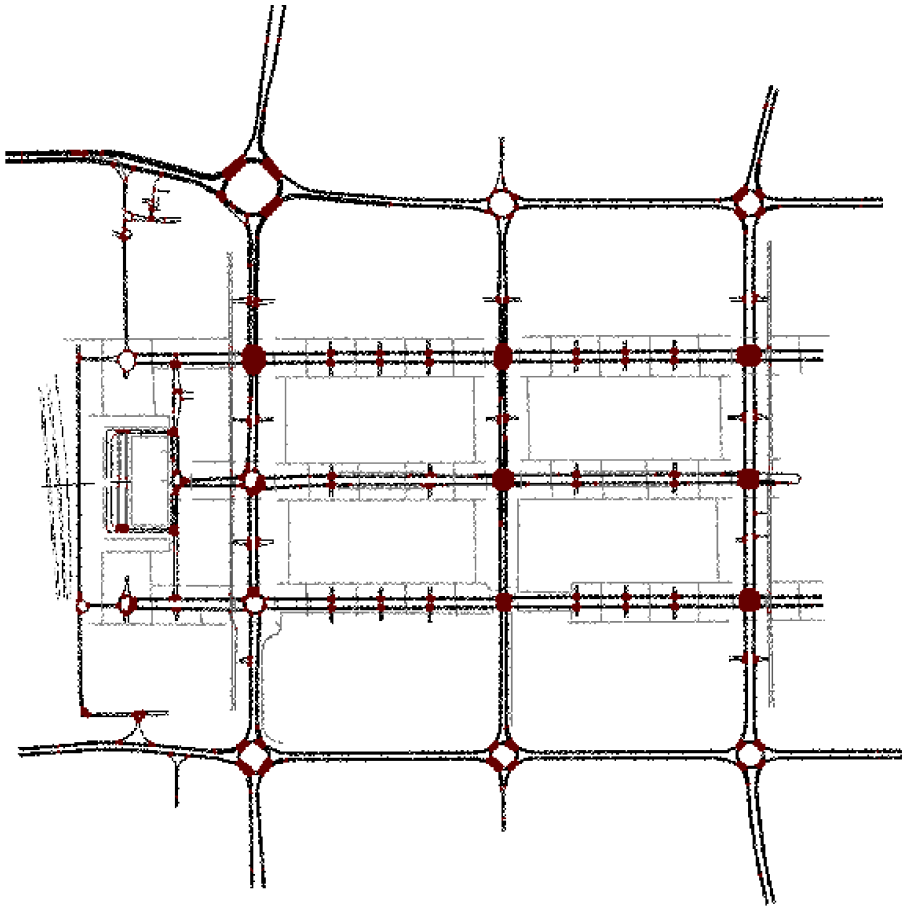

Figure 4: A map of the centre of Milton Keynes (UK), the network in which the experiments are performed

so that, when computing the routes of new approaching vehicles the position of vehicles inside can be accounted for.

The network of Milton Keynes, which is used in our experiments, is shown in Figure 4. Milton Keynes is a town in the United Kingdom, located about 80 kilometres north-west of London, with a population of approximately $230,000$. The model covers an area of approximately 2.9 square kilometres, and includes more than 25 junctions and more than 50 links.

The experiment presented in this section follow an iterative approach. At time $t = 0$, the network is empty and new vehicles approach the network, for these vehicles the CUTC will compute both the route and the expected positions of vehicles in the street (in terms of `enter` and `exit` atoms). At the following time step, those vehicles will become *simulated* and the CUTC will then find routes of new approaching vehicles, keeping in consideration the position of *simulated* vehicles found in the previous iterations.

Comparison with real traffic. For the purpose of evaluating the performance of the proposed framework, in this paragraph, we will compare real traffic data of the Milton Keynes urban area with a simulation in which the same vehicles are routed using our proposed approach. In order to be able to compare the real data with our framework, historical traffic data provided by the Milton Keynes Council and

	Real	*ASP*
Total Duration [s]	15,729	5,065
Avg. Route Length [m]	2,465	2,107
Avg. Speed [m/s]	2.49	5.28
Avg. Duration [s]	3,718.95	515.82
Avg. Waiting Time [s]	3,132.36	259.39
Avg. Depart Delay [s]	791.78	55.69

Table 1: Performance of real traffic data coming from the Milton Keynes urban area and the same vehicles routed using the proposed approach

gathered by sensors distributed in the region between December 2015 and December 2016, has been transformed into a SUMO simulation model. The model simulates the morning rush hour (between 8am and 9am on non-holiday weekdays), in which 1900 vehicles move inside the network. The model has been calibrated and validated. In order to obtain the performance for the ASP-based framework, the architecture shown in Figure 3 has been put in place. Table 1 shows a comparison between the two approaches in terms of *Total Duration* (i.e. the time the last vehicles exits the network), *Average Route Length, Speed, Duration, Waiting Time* (i.e. the time vehicles spend in queues) and *Depart Delay* (i.e. time vehicles spent waiting for the road to free in order to enter the network). As it can be seen by the comparison, the proposed approach is able to greatly increase the over-all performance of the network, spreading traffic and reducing congestion, increasing the average speed of vehicles and allowing the network to free faster.

Scalability. In the proposed framework, the ASP encoding has to keep track of the position of hundreds of vehicles inside the network in order to route the traffic of new approaching vehicles in an optimal way. In the experiment presented in the previous paragraph, for example, ten minutes after the start of the simulation, up to 200 vehicles are inside the network. Even with this large number of vehicles, the CLINGO solver is able to find an *optimal* route in less than 5*s* in most of the cases. This is possible due to the preliminary work performed by the preprocessor which leaves to the optimizer to explore a pruned search space containing only viable solutions, of which the best must be found. To further improve performance, CLINGO is executed with the option `--parallel-mode=2` in order to parallelize two optimization algorithms: Branch and Bound with Restart on Model [16] and the Unsatisfiable Shrinking Core [2] which has already proven to be effective in [7]. Experiments were run on a MacBook Pro with a 2.5 GHz Intel Core i7 quad-core, with 16 GB of RAM. The fast resolution of the problems gives the proposed framework the possibility to be implemented in real time scenarios.

4 Conclusions and Future Work

In this paper, we presented an ASP-based framework which allows to efficiently simulate the flow of traffic in large real-life networks. We showed how this framework can be used to optimize the cost of routes and the time it takes for a vehicle to complete them. A comparison with real data showed that the proposed framework is able to reduce congestion and guide the vehicles in reaching their destination faster. We have also shown how the proposed framework is able to scale in order to be able to deal with instances with hundreds of vehicles inside. Even if the results proved the feasibility of our approach and

showed good performances, the proposed framework can still be improved with the aim of increasing its modelling capacities and consequentially the optimization of the traffic flow.

Future work. In the following years of his PhD, the author plans to tackle several, more complicated, aspects of Urban Traffic, which would help to improve the modelling capacity of the framework and reduce congestion in the network even further:

- Implement other metrics which do not depend directly on the route chosen by the vehicles. In the proposed approach, indeed, only costs associated with the route of the vehicles were considered; this can be used to model a monetary cost of the street (for example tolls) and also to penalize certain routes for certain types of vehicles (we could suppose that trucks or heavy vehicles should avoid entering the city centre). On the other hand, pollution is more correlated with the number of vehicles, with their speed and with the number of *"start and stops"* the vehicles take.
- Deal with re-routing of vehicles inside the network. In our proposed approach, vehicle routing is only possible when the vehicle enters the network. While this approach is easier to implement and requires fewer communications between vehicles and the CUTC can lead to suboptimal solutions. In fact, in some scenarios, it would also be beneficial to reroute vehicles inside the network with the aim to adjust traffic and allow a more rapid flow of vehicles.
- Model traffic lights and keep track of their switch phases in order to route traffic accordingly. The presence of traffic lights could be modelled in two ways: (i) by simply adding a penalizing time (i.e., the average time in which the traffic light is red) for routes which includes traffic lights, or (ii) by modelling precisely the time dependent switching of green phases of traffic lights in the ASP encoding. These two methods differ enormously in terms of complexity, but are both worth exploring in order to investigate their benefits.
- Model larger and more complicated urban areas. We discussed how the proposed approach is able to rapidly find optimal solutions in scenarios with up to 200 vehicles. Even if the proposed approach can successfully manage traffic in a small network like Milton Keynes, it would be interesting to investigate the solution on larger urban areas with thousands of vehicles.

References

[1] Rusul Abduljabbar, Hussein Dia, Sohani Liyanage & Saeed Asadi Bagloee (2019): *Applications of artificial intelligence in transport: An overview*. Sustainability 11(1), p. 189, doi:10.3390/SU11010189.

[2] Mario Alviano & Carmine Dodaro (2020): *Unsatisfiable Core Analysis and Aggregates for Optimum Stable Model Search*. Fundam. Informaticae 176(3-4), pp. 271–297, doi:10.3233/FI-2020-1974.

[3] Chitta Baral (2010): *Knowledge Representation, Reasoning and Declarative Problem Solving*. Cambridge University Press. Available at `http://www.cambridge.org/de/academic/subjects/computer-science/artificial-intelligence-and-natural-language-processing/knowledge-representation-reasoning-and-declarative-problem-solving`.

[4] R.D. Bretherton (1990): *SCOOT urban traffic control system — Philosophy and evaluation*, pp. 237–239. doi:10.1016/B978-0-08-037025-5.50040-2.

[5] Francesco Calimeri, Wolfgang Faber, Martin Gebser, Giovambattista Ianni, Roland Kaminski, Thomas Krennwallner, Nicola Leone, Marco Maratea, Francesco Ricca & Torsten Schaub (2020): *ASP-Core-2 Input Language Format*. Theory Pract. Log. Program. 20(2), pp. 294–309, doi:10.1017/S1471068419000450.

[6] Matteo Cardellini, Marco Maratea, Mauro Vallati, Gianluca Boleto & Luca Oneto (2021): *In-Station Train Dispatching: A PDDL+ Planning Approach*. In Susanne Biundo, Minh Do, Robert Goldman, Michael Katz, Qiang Yang & Hankz Hankui Zhuo, editors: *Proceedings of the Thirty-First International Conference on Automated Planning and Scheduling, ICAPS 2021, Guangzhou, China (virtual), August 2-13, 2021*, AAAI Press, pp. 450–458. Available at `https://ojs.aaai.org/index.php/ICAPS/article/view/15991`.

[7] Matteo Cardellini, Paolo De Nardi, Carmine Dodaro, Giuseppe Galatà, Anna Giardini, Marco Maratea & Ivan Porro (2021): *A Two-Phase ASP Encoding for Solving Rehabilitation Scheduling*. In Sotiris Moschoyiannis, Rafael Peñaloza, Jan Vanthienen, Ahmet Soylu & Dumitru Roman, editors: *Rules and Reasoning - 5th International Joint Conference, RuleML+RR 2021, Leuven, Belgium, September 13-15, 2021, Proceedings*, *Lecture Notes in Computer Science* 12851, Springer, pp. 111–125, doi:10.1007/978-3-030-91167-6_8.

[8] Isabel Cenamor, Lukáš Chrpa, Falilat Jimoh, Thomas L McCluskey & Mauro Vallati (2014): *Planning & scheduling applications in urban traffic management*. In: *Proceedings of PlanSIG 2014, Annual Workshop of UK Planning & Scheduling Special Interest Group, 2014.*

[9] Lukás Chrpa, Daniele Magazzeni, Keith McCabe, Thomas Leo McCluskey & Mauro Vallati (2016): *Automated planning for Urban traffic control: Strategic vehicle routing to respect air quality limitations*. *Intelligenza Artificiale* 10(2), pp. 113–128, doi:10.3233/IA-160099.

[10] Carmine Dodaro, Giuseppe Galatà, Andrea Grioni, Marco Maratea, Marco Mochi & Ivan Porro (2021): *An ASP-based Solution to the Chemotherapy Treatment Scheduling problem*. *Theory Pract. Log. Program.* 21(6), pp. 835–851, doi:10.1017/S1471068421000363.

[11] Carmine Dodaro & Marco Maratea (2017): *Nurse Scheduling via Answer Set Programming*. In Marcello Balduccini & Tomi Janhunen, editors: *Logic Programming and Nonmonotonic Reasoning - 14th International Conference, LPNMR 2017, Espoo, Finland, July 3-6, 2017, Proceedings*, *Lecture Notes in Computer Science* 10377, Springer, pp. 301–307, doi:10.1007/978-3-319-61660-5_27.

[12] Mariagrazia Dotoli, Maria Pia Fanti & Carlo Meloni (2006): *A signal timing plan formulation for urban traffic control*. *Control Engineering Practice* 14(11), pp. 1297–1311, doi:10.1016/j.conengprac.2005.06.013.

[13] Thomas Eiter, Andreas A. Falkner, Patrik Schneider & Peter Schüller (2020): *ASP-Based Signal Plan Adjustments for Traffic Flow Optimization*. In Giuseppe De Giacomo, Alejandro Catalá, Bistra Dilkina, Michela Milano, Senén Barro, Alberto Bugarín & Jérôme Lang, editors: *ECAI 2020 - 24th European Conference on Artificial Intelligence, 29 August-8 September 2020, Santiago de Compostela, Spain, August 29 - September 8, 2020 - Including 10th Conference on Prestigious Applications of Artificial Intelligence (PAIS 2020)*, *Frontiers in Artificial Intelligence and Applications* 325, IOS Press, pp. 3026–3033, doi:10.3233/FAIA200478.

[14] Maria Fox & Derek Long (2006): *Modelling Mixed Discrete-Continuous Domains for Planning*. *J. Artif. Intell. Res.* 27, pp. 235–297, doi:10.1613/jair.2044.

[15] Martin Gebser, Roland Kaminski, Benjamin Kaufmann, Max Ostrowski, Torsten Schaub & Philipp Wanko (2016): *Theory Solving Made Easy with Clingo 5*. In Manuel Carro, Andy King, Neda Saeedloei & Marina De Vos, editors: *Technical Communications of the 32nd International Conference on Logic Programming, ICLP 2016 TCs, October 16-21, 2016, New York City, USA*, *OASIcs* 52, Schloss Dagstuhl - Leibniz-Zentrum für Informatik, pp. 2:1–2:15, doi:10.4230/OASIcs.ICLP.2016.2.

[16] Martin Gebser, Roland Kaminski, Benjamin Kaufmann, Javier Romero & Torsten Schaub (2015): *Progress in clasp Series 3*. In Francesco Calimeri, Giovambattista Ianni & Miroslaw Truszczynski, editors: *Logic Programming and Nonmonotonic Reasoning - 13th International Conference, LPNMR 2015, Lexington, KY, USA, September 27-30, 2015. Proceedings*, *Lecture Notes in Computer Science* 9345, Springer, pp. 368–383, doi:10.1007/978-3-319-23264-5_31.

[17] Michael Gelfond & Vladimir Lifschitz (1991): *Classical Negation in Logic Programs and Disjunctive Databases*. *New Gener. Comput.* 9(3/4), pp. 365–386, doi:10.1007/BF03037169.

[18] Pablo Álvarez López, Michael Behrisch, Laura Bieker-Walz, Jakob Erdmann, Yun-Pang Flötteröd, Robert Hilbrich, Leonhard Lücken, Johannes Rummel, Peter Wagner & Evamarie WieBner (2018): *Microscopic Traffic Simulation using SUMO*. In Wei-Bin Zhang, Alexandre M. Bayen, Javier J. Sánchez Medina &

Matthew J. Barth, editors: *21st International Conference on Intelligent Transportation Systems, ITSC 2018, Maui, HI, USA, November 4-7, 2018*, IEEE, pp. 2575–2582, doi:10.1109/ITSC.2018.8569938.

[19] Hannah Ritchie Max Roser & Esteban Ortiz-Ospina (2013): *World Population Growth*. Our World in Data. Available at `https://ourworldindata.org/world-population-growth`.

[20] JC Miles & Andrew J Walker (2006): *The potential application of artificial intelligence in transport*. In: *IEE Proceedings-Intelligent Transport Systems*, 153, IET, pp. 183–198.

[21] Ilkka Niemelä (1999): *Logic Programs with Stable Model Semantics as a Constraint Programming Paradigm*. *Ann. Math. Artif. Intell.* 25(3-4), pp. 241–273, doi:10.1023/A:1018930122475.

[22] Markos Papageorgiou, Christina Diakaki, Vaya Dinopoulou, Apostolos Kotsialos & Yibing Wang (2003): *Review of road traffic control strategies*. *Proc. IEEE* 91(12), pp. 2043–2067, doi:10.1109/JPROC.2003.819610.

[23] Miquel Ramírez, Michael Papasimeon, Nir Lipovetzky, Lyndon Benke, Tim Miller, Adrian R. Pearce, Enrico Scala & Mohammad Zamani (2018): *Integrated Hybrid Planning and Programmed Control for Real Time UAV Maneuvering*. In Elisabeth André, Sven Koenig, Mehdi Dastani & Gita Sukthankar, editors: *Proceedings of the 17th International Conference on Autonomous Agents and MultiAgent Systems, AAMAS 2018, Stockholm, Sweden, July 10-15, 2018*, International Foundation for Autonomous Agents and Multiagent Systems Richland, SC, USA / ACM, pp. 1318–1326. Available at `http://dl.acm.org/citation.cfm?id=3237896`.

[24] Tom Schaffnit (2010): *Automotive Standardization of Vehicle Networks*. John Wiley & Sons, Ltd, doi:10.1002/9780470661314.ch7.

[25] Mauro Vallati, Daniele Magazzeni, Bart De Schutter, Lukás Chrpa & Thomas Leo McCluskey (2016): *Efficient Macroscopic Urban Traffic Models for Reducing Congestion: A PDDL+ Planning Approach*. In Dale Schuurmans & Michael P. Wellman, editors: *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence, February 12-17, 2016, Phoenix, Arizona, USA*, AAAI Press, pp. 3188–3194. Available at `http://www.aaai.org/ocs/index.php/AAAI/AAAI16/paper/view/11985`.

[26] Joeri Van Mierlo, G Maggetto, E Burgwal & R Gense (2004): *Driving style and traffic measures - Influence on vehicle emissions and fuel consumption*. *Proceedings of the Institution of Mechanical Engineers Part D Journal of Automobile Engineering* 218, pp. 43–50, doi:10.1243/095440704322829155.

[27] Axel Wegener, Michal Piorkowski, Maxim Raya, Horst Hellbrück, Stefan Fischer & Jean-Pierre Hubaux (2008): *TraCI: An Interface for Coupling Road Traffic and Network Simulators*. doi:10.1145/1400713.1400740.