Probability in the Engineering and Informational Sciences, **22**, 2008, 587–602. Printed in the U.S.A.
doi:10.1017/S026996480800034X

AN MDP DECOMPOSITION APPROACH FOR TRAFFIC CONTROL AT ISOLATED SIGNALIZED INTERSECTIONS

RENÉ HAIJEMA AND JAN VAN DER WAL
Department of Quantitative Economics
Faculty of Economics and Business
University of Amsterdam
1018WB Amsterdam, The Netherlands
E-mail: r.haijema@uva.nl; jan.v.d.wal@tue.nl

This article presents a novel approach for the *dynamic* control of a signalized intersection. At the intersection, there is a number of arrival flows of cars, each having a single queue (lane). The set of all flows is partitioned into disjoint combinations of nonconflicting flows that will receive green together. The dynamic control of the traffic lights is based on the numbers of cars waiting in the queues. The problem concerning when to switch (and which combination to serve next) is modeled as a Markovian decision process in discrete time. For large intersections (i.e., intersections with a large number of flows), the number of states becomes tremendously large, prohibiting straightforward optimization using value iteration or policy iteration. Starting from an optimal (or nearly optimal) fixed-cycle strategy, a one-step policy improvement is proposed that is easy to compute and is shown to give a close to optimal strategy for the dynamic problem.

1. INTRODUCTION

In this article we focus on the dynamic control at a signalized intersection in isolation. Although usually traffic lights are part of a network, it is often realistic to consider them as if they are in isolation. We will present a novel approach that allows extensions to network cases.

A control mechanism should prescribe when a green signal should change into yellow and which (group of) flows gets the right of way after an "all red" phase. For psychological and safety reasons, most control strategies found in practice are *cyclic*: the order in which the (groups of) flows are served is fixed (e.g., 1, 2, 3, 1, 2, 3, 1, etc.).

 0269-9648/08 $25.00

Therefore, we focus on the decision *when to switch* from green to yellow (although our approach can deal with acyclic control as well). This decision is to be based on the current state of the traffic lights and on the number of cars waiting at each of the queues. These numbers are assumed to be known from magnetic-loop detectors or cameras. In principle, this is possible for a number of years already; see, for example, [4].

Our approach also allows for the use of information on near-future arrivals next to statistical properties of the arrival process, but this is beyond the scope of this article.

Two basic control strategies are *fixed-cycle* control (FC) and *exhaustive* control (XH). Under FC, not only is the order fixed but also the durations of the green periods. XH keeps signals green until all queues that have the right of way are "exhausted" (empty) and then the signals are changed into yellow. The cyclic variant is abbreviated by XHC. An alternative is anticipative exhaustive control, which anticipates departures during one or two yellow slots: XHC(1) and XHC(2), respectively.

In practice, one finds various existing control systems such as TRANSYT, SCOOT, and many others (for an overview, see. [13]). The most common systems are based on FC or a mixture of FC and XH: Each combination receives green during a minimum green time and its green period ends when a maximum green time is reached, or when its queues have been empty for some time.

Some studies report on systems that control the lights using more detailed information on the queue lengths. In MOVA, the time gains or losses of switching to yellow are computed dynamically for all flows under simplifying assumptions; see [19]. Some systems take decisions based on a Markovian decision problem (MDP) formulation, but solving a dynamic programming is numerically involved. Examples for a single intersection are PRODYN and RHODOS, which employs dynamic programming over a rolling planning horizon; see [3] and [16], respectively.

Because of the computational complexity of solving a dynamic program, computer scientists apply learning algorithms to smooth the traffic flow. An interesting example is that of Wiering, van Veenen, Vreeken, and Koopman [21]: A learning algorithm is applied to heuristically minimize the overall average delay per car by dynamically adjusting the lights and routes traveled by cars having specific destinations in a network. The prediction of the delay is made in a microscopic simulation environment through reinforcement learning, so no prior known distributions of car arrivals are used.

In this article we present a novel approach based on a discrete-time MDP formulation. Starting from a (nearly) optimal fixed-cycle strategy (FC), we execute one *policy improvement* step that leads to a dynamic control strategy, which turns out to be quite good. This strategy tells when to lengthen, reduce, or end the green period using detailed information on the queue lengths. The key idea is that for FC, the multidimensional system decomposes, such that it can be evaluated flow by flow. This decomposition reduces the computational burden significantly and allows one to apply the approach to intersections of virtually any size.

As mentioned in Tijms [17], the idea of applying a one-step policy improvement after decomposition of the multidimensional MDP goes back to Norman [12]. Krishnan and Ott [5,6] and Bhulai [1] successfully applied the approach to the routing

of telephone calls. Sassen, Tijms, and Nobel [15] used it for the routing of jobs to parallel queuses. Wijngaard [22] used the decomposition approach for the control of production and inventory problems. To our knowledge, the present article is the first to apply the approach to the dynamic control of traffic lights.

This article is organized as follows. Section 2 discusses some modeling aspects. Section 3 studies the MDP formulation. The one-step improvement approach is presented in Section 4. Section 5 gives some numerical results for two examples of intersections under various traffic conditions. Section 6 closes the article with some conculsions and remarks.

2. MODELING ASSUMPTIONS AND SOME NOTATIONS

The (automated) controller of the traffic lights periodically makes decisions about how to adjust the lights. When switching from green for one set of flows to green for another set, a fixed switching time is to be acknowledged. This suggests modeling the problem in *dicrete time*, which calls for some simplifications of the processes.

2.1. Discrete Time

The time unit or slot is taken to be the time a car needs to pass the intersection when the light is green. (One might think of this length as being 2 seconds. Two seconds also corresponds to a safe driving distance between cars at different speeds: about 4 m at 15 km/h and 23 m at 50 km/h.)

2.2. Grouping of Flows

The F is partitioned into S disjoint subsets $C_1, \ldots, C_S$. We call a subset of flows a *combination of flows*, or simply a *combination*. The combinations are fixed and not part of our optimization problem. Flows in the same combination will always receive green, yellow, and red together. If one combination has green or yellow, all other lights show red.

2.3. Signal Process

Changing from green for one combination to green for another combination of flows takes three slots (6 s): two slots of yellow and one slot in which all lights show red to clear the intersection. This *switching time* is independent of the two sets of flows involved. (In our model, the yellow and red slots have the same duration as the green slots. It is, however, straightforward to extend the approach to different slot lengths.)

2.4. Arrival Process

Arrivals in different flows and in different slots are independent. Per flow, the number of car arrivals in a slot is either 0 or 1. The probability of an arrival in flow f is denoted

https://doi.org/10.1017/S026996480800034X Published online by Cambridge University Press

by q_f. (One could also assume Poisson arrivals, but having more than one car arrival on a lane within a slot is not very realistic.)

2.5. Departure Process

When the signal shows green or yellow and cars are present, in every slot exactly one car passes the stopping line. From a driver's point of view it might be more realistic to say that during the first yellow slot, the probability that a car passes is still 1, but during the second yellow slot, it is less than 1. The model allows for this, as it allows for any Markovian stochastic departure process, but notations would become more involved. Assuming deterministic departure processes with an interdeparture time of one slot (2s) is quite common in the practice of traffic engineering [14].

2.6. Queuing Process

A car arriving at an empty queue that has the right of way passes the stopping line without delay. Therefore, we assume new arrivals to take place at the beginning of the slot (after the observation of the state of the process) and to join the tail of the queue. Departures from the queue takes place at the end of the slot prior to the observation of the new state.

3. FORMULATION AS A MARKOV DECISION PROBLEM

To describe the MDP model in detail, the states, the decisions, the transitions and the costs are specified.

3.1. States

The arrival and departure processes are assumed to be Markovian (i.e., memoryless), so the state of the traffic is fully described by the vector $\underline{k} = (k_1, \ldots, k_F)$, with k_f the number of cars in flow f present at the beginning of a slot. Further, x denotes the state of the light with $x = (l, i)$, if C_l is having green ($i = 0$), first yellow ($i = 1$), second yellow ($i = 2$), or red ($i = 3$). (Note that because the cyclic order is to be kept, the "all red" state ($i = 3$) also marks which combination (l) had green. If one has no constraint on the order, one "all red" state suffices.)

The states are thus denoted by the vector $(\underline{k}, x) = (\underline{k}, (l, i))$.

3.2. Decisions

The decisions one might take depend primarilly on the traffic light state but also on the lengths of the queues.

- If all lights are red, the possible decisions are to keep all lights red (if all queues are empty, this might be optimal because switching always costs two yellow

https://doi.org/10.1017/S026996480800034X Published online by Cambridge University Press

slots) or to give green to one of the combinations. If we want to maintain the cyclic order, a combination can only be skipped if all queues for that combination are empty.

- If the lights are green for one subset, there are two decisions: Keep the lights as they are or change from green to "first yellow".
- At the end of a first yellow slot, there is only one decision: Continue to the second yellow slot.
- After the second yellow, the only decision is to change into red for all flows.

When no car is waiting at any queue, a special action is taken: the state of the lights (if not yellow) is frozen, meaning green lights stay green and red lights stay red during the coming slot.

The decision space, denoted by $\mathcal{A}(\underline{k}, (l, i))$ is thus

$$\mathcal{A}(\underline{k}, (l,i)) = \begin{cases} \{(l,0),(l,1)\} & \text{if } i = 0 \text{ (green)} \\ (l, i+1) & \text{if } i = 1,2 \text{ (yellow)} \\ \{(l,3),(l',0)\} & \text{if } i = 3 \text{ (all red), with } l' \text{ the next} \\ & \quad \text{nonempty combination.} \end{cases}$$

Decisions are taken at the beginning of a slot and executed instantaneously. Thus, if a decision grants green to a combination, cars of that combination can leave in the very same slot.

3.3. Transition Probabilities

Action a implies an instantaneous change of the lights from state x into state a. Hence, the transition probability from state $(\underline{k}, x)$ to state $(\underline{k}', x')$ is zero unless $x' = a$. The transition probabilities, denoted by $p(\underline{k}, x; \underline{k}', a)$, are best described by considering each flow separately. Let $p_f(k_f, a, k_f')$ denote the transition probability for the number of cars in flow f when action a is taken. Since the flows are independent, the simultaneous transition probabilities are simply the product

$$p(\underline{k}, x; \underline{k}', a) = \prod_{f=1}^{F} p_f(k_f, a, k_f'). \tag{1}$$

Further, if, by action a, flow f receives green or yellow during the coming slot, the transition probabilities per queue are given by (with $y^+ = \max\{y, 0\}$)

$$p_f(k_f, a, (k_f - 1)^+) = 1 - q_f, \quad p_f(k_f, a, k_f) = q_f, \tag{2}$$

and if action α implies red to flow f, then

$$p_f(k_f, a, k_f) = 1 - q_f, \quad p_f(k_f, a, k_f + 1) = q_f. \tag{3}$$

https://doi.org/10.1017/S026996480800034X Published online by Cambridge University Press

3.4. Criterion and Costs

The criterion we use is the overall average waiting time per car. According to Little's law, this is equivalent to minimizing the average number of cars waiting for all queues together. To achieve this, we use a linear cost structure with one unit of costs for every car present at the beginning of a slot. Let $c(\underline{k}, x)$ denote the one slot costs in state $c(\underline{k}, x)$, then

$$c(\underline{k}, x) = \sum_{f=1}^{F} k_f. \tag{4}$$

3.5. Reduction to a Finite State Space

With the above definition of the states and transitions, the state space will be countable. In order to reduce the state space to a finite one, we limit in each flow the number of cars that can be present. For this, we have to reject arrivals when the "buffer of a flow is full" (i.e., when the maximum number of cars for that flow has been reached). If an arriving car is "lost" because the buffer is full, we penalize this by a cost term based on the concept of externality cost. This means that these costs are not only the costs for the time the car is expected to spend in the system but also contain the expected delay the car would have caused on cars arriving later in the same flow.

3.6. Successive Approximations

In order to obtain the average number of waiting cars, we use successive approximations. Let $V_n(\underline{k}, x)$ denote the expected minimal costs over an horizon of n slots when starting in state $(\underline{k}, x)$. Then, starting with $V_0(\underline{k}, x) = 0$, one recursively computes $V_n(\underline{k}, x)$:

$$V_{n+1}(\underline{k}, x) = c(\underline{k}, x) + \min_{a \in \mathcal{A}(\underline{k}, x)} \sum_{\underline{k}'} p(\underline{k}, x; \underline{k}', a) \cdot V_n(\underline{k}', a) \tag{5}$$

until the span (difference between the maximal and minimal component) of $V_{n+1} - V_n$ is sufficiently small. (One can show that convergence is guaranteed as the system empties every now and then, so that all states communicate.) Then the average of the maximal and minimal component of $V_{n+1} - V_n$ is the approximation for the minimal average number of cars in the system, and a (nearly) optimal stationary strategy π is obtained from

$$\pi(\underline{k}, x) = \arg \min_{a \in \mathcal{A}(\underline{k}, x)} \sum_{\underline{k}'} p(\underline{k}, x; \underline{k}', a) V_n(\underline{k}', a). \tag{6}$$

4. THE ONE-STEP IMPROVEMENT APPROACH

Even after the state space reduction discussed in Section 3.5, the multidimensionality of the system (one dimension per flow) usually leads to tremendous state spaces so

https://doi.org/10.1017/S026996480800034X Published online by Cambridge University Press

that, in general, a straightforward dynamic programming approach is not feasible. Therefore, we will use a decomposition approach combined with a one-step policy improvement technique. As outlined in Section 1, over the years this technique has been applied successfully to a number of multidimensional decision problems (see [1,5,6,12,15,22]). The decomposition requires a special strategy, which is, in our case, FC. In our implementation of the approach, four steps are distinguished:

1. Find a good FC for which the process decomposes into F periodic but independent processes, one for each flow.
2. Compute for each flow and each number of cars the relative values corresponding to the slot within the cycle.
3. Use these relative values to execute a policy improvement step that leads to a (hopefully very good) strategy for the dynamic control of the traffic light.
4. Evaluate the new strategy by simulation, and compare it with a number of traffic light control strategies seen in practice. For small problems, comparison with the optimal strategy is possible by solving the MDP.

The first three steps are discussed in the next three sub sections. The results obtained in the last steps are reported in Section 5.

4.1. Fixed-Cycle System

The reason for starting with FC is that, in this case, the flows are completely independent of each other. So the analysis can be done "flow by flow". In the next subsection, we quickly discuss the numerical solution of the Markov chain corresponding to one flow. In particular, we are interested in obtaining the relative values (or bias terms) of the states. (Since the late 1950s and early 1960s, the evaluation of FC control has received considerable attention. To mention only a few references, see Webster [20], Miller [8,9], Darroch [2], Newell [10,11], McNeill [7], and Van Leeuwaarden [18].)

At this point we are just interested in setting a (nearly) optimal FC. We therefore use a simple local search algorithm. The search starts with a feasible cycle of minimum length (all queues are just stable). Next, successively one increases the length of the cycle by one slot according to the best increase of the green period for one of the combinations. This process of incremental search is stopped when the last so many increases did not yield a better cycle. The best cycle so far is considered the optimal cycle. (Our approach, appears to be fairly insensitive to minor changes in the duration of the FC from which we start.)

4.2. Relative Values per Flow

FC under, each flow perceives a periodic green, yellow, red cycle, independent of the queue lengths of the various flows. Let D be the cycle duration. Then, for flow f, there is a number d_f of green or yellow departure slots followed by $r_f = D - d_f$ red slots. So, for flow f, the FC results in a periodic Markov chain with d_f departure slots and

https://doi.org/10.1017/S026996480800034X Published online by Cambridge University Press

r_f red slots. The state of the chain can be described as a pair (k, t), with k the number of cars present at the beginning of a slot and t the number of the slot within the cycle, $t = 1, \ldots, D$. (For FC, the traffic light state description is somewhat simpler than in the general dynamic situation.) The analysis of such a periodic Markov chain, and in particular the computation of the relative values, can simply be done numerically by dynamic programming. As the state space for one flow is small, one can make the buffer so large that the probability on rejection of arrivals plays no role. (The equilibrium distribution, and thus the average costs per time unit, can also be computed using a generating function approach; see Darroch [2] and Van Leeuwaarden [18].)

The simple approach is the following. Note that the immediate cost in state (k, t) is just the number of cars present, thus equal to k. Let $v_n^f(k, t)$be the total expected cost (i.e., the total expected number of cars) over an horizon of n slots for flow f only when starting in state (k, t).

1. Define $v_0^f(k, t) = 0$ for all k and t.
2. For all k and t, recursively compute $v_n^f(k, t)$, for $n = 1, 2, \ldots$, as follows:
   - If t is a departure slot for f,
   $$v_{n+1}^f(k, t) = k + q_f \cdot v_n^f(k, t + 1) + (1 - q_f) \cdot v_n^f((k - 1)^+, t + 1); \quad \textbf{(7)}$$
   - if t is a red slot,
   $$v_{n+1}^f(k, t) = k + q_f \cdot v_n^f(k + 1, t + 1) + (1 - q_f) \cdot v_n^f(k, t + 1), \quad \textbf{(8)}$$

   where $v_n^f(\cdot, D + 1)$ should be read as $v_n^f(\cdot, 1)$.
   For all k and t, the cycle average $(v_{n+D}^f(k, t) - v_n^f(k, t))/D$ converges exponentially fast to the average costs per slot, as the k-step Markov chains are all irreducible.
3. A way to obtain the relative values is by choosing a reference state for example $(0, D)$ and then computing for sufficiently large n, the differences $v_n^f(k, t) - v_n^f(0, D)$. For periodic chains, one has to be careful, as these differences change periodically with n. Then a useful definition for the relative value vector v_{rel}^f (ignoring the dependence on n, or assuming n is sufficiently large) is
$$v_{\text{rel}}^f := \frac{v_n^f + v_{n+1}^f + \cdots + v_{n+D-1}^f}{D} - \frac{v_n^f(0, D) + v_{n+1}^f(0, D) + \cdots + v_{n+D-1}^f(0, D)}{D} \cdot e, \quad \textbf{(9)}$$
   with $e = (1, 1, \ldots, 1)^T$. Given the reference state, the relative values are unique.

Example: Consider as an example an intersection where four flows are served in two combinations: flows 1 and 3 form C_1, and 2 and 4 compose C_2. (For an illustration of

https://doi.org/10.1017/S026996480800034X Published online by Cambridge University Press

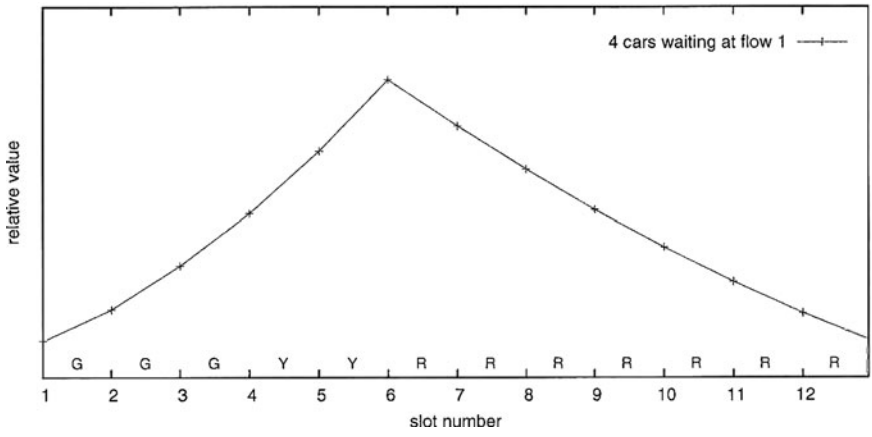


FIGURE 1. Relative value curve when four cars are waiting at flow 1.

the infrastructure, we refer to intersection F4C2 in Section 5.) Both combinations get green during three slots. Switching takes another β slots, so the duration of a cycle is 12 slots. A typical relative value curve of flow 1 is depicted in Figure 1 when four cars are waiting at its queue. Slots 1–5 are departure slots for flow 1; for the next seven slots, the signals of flow 1 show red. The worst position in the cycle is the start of slot 6, since all four cars has to wait for at least seven slots. The best position is the start of slot 1, because then the remaining green period last longest. The differences heavily depend on k, the number of waiting cars. Similar curves can be draw for all numbers of cars and all flows.

4.3. Fixed-Cycle Improvement

Now, let us return to the system as a whole. Under the FC control, the systems state is described by the pair $(\underline{k}, t)$, with $\underline{k} = (k_1, \ldots, k_F)$ the numbers of cars in each flow and t the slot within the cycle. In slot t, at most one subset has green or yellow; all other subsets have red. The decisions one might make concern the traffic light and only indirectly the number of cars and can be seen as time jumps within the cycle. Not all jumps are possible. From a green slot, only jumps to other green slots of the same green interval and the first yellow slot are possible. During the two yellow slots, there is no freedom. After an all-red slot, one might choose to jump to any slot that is green for some subset or one might keep the all-red position. (Since we want to maintain the cyclic order, we only allow to skip combinations for which all flows are empty.)

Consider state $(\underline{k}, t)$. Let $\mathcal{T}(\underline{k}, t)$ be the set of possible time jumps in slot t. Then, for every $s \in \mathcal{T}(\underline{k}, t)$, we compute $\sum_{f=1}^{F} v_{\text{rel}}^{f}(k_f, s)$. The one-step optimal decision (time jump) in state $(\underline{k}, t)$, thus assuming that after this jump the system returns to the

https://doi.org/10.1017/S026996480800034X Published online by Cambridge University Press

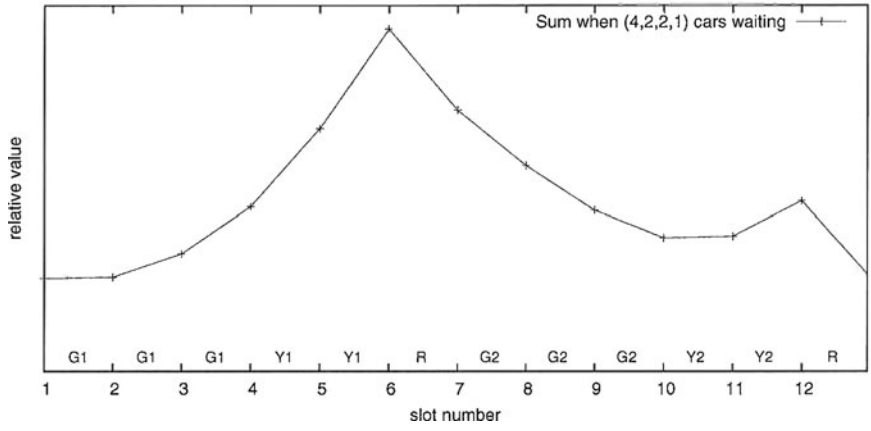


FIGURE 2. Relative value curve when (4,2,2,1) cars are waiting at flows 1–4.

FC regime, follows from

$$\sigma(\underline{k}, t) = \arg \min_{s \in T(\underline{k},t)} \sum_{f=1}^{F} v_{\text{rel}}^{f}(k_f, s). \tag{10}$$

These $\sigma(\underline{k}, t)$ together constitute the improved strategy σ that is expected to be close to optimal.

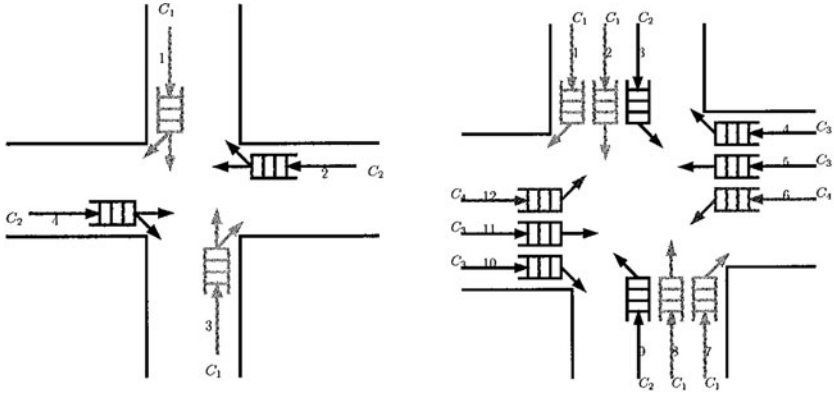
The implementation of the strategy σ is done as follows. In every slot, the state is observed as a pair $(\underline{k}, t)$, where t is the result of taking one time step from the previous decision. Then σ prescribes the next time jump and so forth.

Example: Revisit the example given in the previous subsection. To illustrate the concept, we show in Figure 2 the sum of the relative value curves over all flows with $\underline{k} = (4, 2, 2, 1)$ cars are waiting at flows 1–4. The best position in the cycle (yielding the lowest sum of relative values) is slot 1: the start of green to C_1. This best point is, however, not reachable from all points in the cycle. When C_2 has green, the best decision is to change to yellow, since the best feasible jump is to slot 10, where the curve has a local minimum.

5. EVALUATION AND COMPARISON

In order to evaluate the improved strategy σ, one has to be able to analyze the resulting Markov chain. If the problem is small, this can be done numerically. If the number of flows is large, the number of states is usually too large to analyze the chain. However, simulating the chain is quite simple, and the convergence is, in general, very fast.

https://doi.org/10.1017/S026996480800034X Published online by Cambridge University Press



(a) "F4C2" serve 4 flows in 2 symmetric combinations

(b) "F12C2" serve 12 flows in 4 asymmetric combinations

FIGURE 3. Two typical infrastructures.

In order to judge the quality of the improved strategy σ, we compare it by simulation with a number of other strategies. (In the simulations, all cars are allowed to enter. If the number of cars in a flow would be larger than the largest number considered in the Markov chain model for that flow, then the corresponding relative values are obtained from an extrapolation of the computed relative value function.) The first strategy to compare σ with is the FC strategy. Another interesting strategy is XH: The subsets receive green in some (fixed) order and stay green until the buffers of all flows in the subset are empty; then the next subset of nonempty queues is served. In addition, results are provided for anticipative exhaustive control, XHC(l) and XHC(2), where a queue is called exhausted when less than one (respectively two) cars are present.

5.1. Test Cases

5.1.1. Infrastructures: F4C2 and F12C4

Figure 3 presents two infrastructures that are used as test cases. The flows at each intersection are numbered clockwise. The first is a simple symmetric intersection, called F4C2, where four flows are served in two combinations. Flows 1 and 3 compose C_1, and flows 2 and 4 constitute C_2. (In this simple case, cars are not allowed to turn left.)

The second infrastructure, called F12C4, is much more complex: 12 flows are grouped in 4 combinations (C_1 to C_4). C_1 and C_3 are composed of four flows each, whereas C_2 and C_4 have only two flows:

$$C_1 = \{1, 2, 7, 8\}, \quad C_2 = \{3, 9\}, \quad C_3 = \{4, 5, 10, 11\}, \quad C_4 = \{6, 12\}.$$

https://doi.org/10.1017/S026996480800034X Published online by Cambridge University Press

The combinations and the order in which the combinations are visited (1, 2, 3, 4) are as given. Their optimization is beyond the scope of this article. We will vary the arrival intensities to test the new approach against the other strategies.

5.1.2. Workload and Arrival Rates.

The mean number of cars arriving within a slot at flow f equals q_f. Thus, ignoring the time required for switching between combinations, flow f brings a workload of q_f to the system. If several flows are served in the same combination, for the workload the "heaviest" one determines the contribution of that combination to the overall workload, further denoted as ρ. So,

$$\rho = \sum_{i=1}^{S} \max_{f \in C_i} q_f. \tag{11}$$

If $\rho < 1$, then the system will be stable for exhaustive service. It will also be stable if the fixed cycle is made long enough and each combination gets a sufficient number of departure slots.

In practice, the experienced load will be higher, since the "all red" switching phases consume part of the capacity. For example, consider the symmetric F4C2 case, where at each flow during a slot with probability 0.3 a car arrives. From (11) it follows that $\rho = 0.3 + 0.3 = 0.6$; thus, the workload is said to be 60%. The optimal FC gives green durations of 3 slots for both combinations, resulting in a cycle length of 12 slots. At each flow, on average $0.3 \times 12 = 3.6$ cars arrive within a cycle, whereas at most $3 + 2 = 5$ cars can leave. The true workload, based on the number of green and yellow slots, equals $3.6/5 = 0.72$. For the symmetric case with $\rho = 0.8$, the optimal duration of a green period equals 8 slots and the true workload under the optimal FC of 22 slots then is $0.4 \times 22/(8 + 22) = 0.88$. Thus, $\rho = 0.8$ is already close to saturation. For $\rho = 0.4$, the true workload under the optimal FC of eight slots is 0.53.

In the next subsection we report results on the two infrastructures under consideration at different loads and for both symmetric and asymmetric cases.

Table 1. Overall Mean Waiting Time (in s) for the Symmetric F4C2

Rule	$\rho = 0.40$		$\rho = 0.60$		$\rho = 0.80$	
RVC	5.06		7.01		14.2	
FC	5.43	+7%	8.27	+18%	17.0	+20%
XHC	5.76	+14%	8.82	+26%	19.9	+40%
XHC(1)	5.03	−1%	7.21	+3%	15.5	+9%
XHC(2)	5.09	+1%	7.31	+4%	14.2	+0%
MDP (cyclic)	4.89	−3%	6.95	−1%	13.5	−5%
FC cycle length (in s) and departure times per combination	16 (6, 6)		24 (10, 10)		44 (20, 20)	

https://doi.org/10.1017/S026996480800034X Published online by Cambridge University Press

5.2. A Simple Example (F4C2)

5.2.1. Symmetric Arrival Rates (F4C2).

Table 1 presents the results for varying workloads at a symmetric F4C2 intersection (in the cases $\rho = 0.4, 0.6$, and 0.8, all flows have identical arrival probabilities per slot of 0.2, 0.3, and 0.4, respectively). Thus, our relative value heuristic obtained from the one-step policy improvement, further denoted as RVC (Relative Value Cyclic), is close to the optimal cyclic MDP strategy. The difference is on average only 3%. The gain compared to the FC and XHC is, on average, 15% and 27%, respectively. On average, the heuristic is a little better than the anticipating exhaustive variants, but the difference is small for this simple intersection.

5.2.2. Asymmetric Arrival Rates (F4C2).

Two interesting cases of nonidentical arrival rates for F4C2 are considered:

1. The flows within the same combination are identical, but combination 2 is three times as busy as combination 1,
2. One flow (say flow 1) has a load that is only a third of the load of the other flows.

The results are reported in Table 2 for a load of 0.6. The average waiting time for flow f is denoted as $E(W_f)$. As we see, flows that are part of a busier combination experience a lower waiting time. This might seem counterintuitive, but apparently a heavier load leads to a preferential treatment. If a light loaded flow and a more heavily loaded are together in one combination, then the light loaded flow benefits but the heavily loaded flow suffers a bit.

TABLE 2. Mean Waiting Times (in s) for Two Asymmetric F4C2 Cases at $\rho = 0.6$

Rule	$E(W)$ overall		$E(W_1)$	$E(W_2)$	$E(W_3)$	$E(W_4)$
$q = (0.15, 0.45, 0.15, 0.45)$						
RVC	5.9		10.5	4.4	10.4	4.4
FC departure times 6, 14 seconds	6.9	+17%	11.2	5.4	11.2	5.4
XHC	7.5	+27%	11.0	6.3	11.0	6.3
XHC(1)	6.6	+12%	8.6	5.9	8.6	5.9
XHC(2)	7.3	+24%	7.7	7.2	7.7	7.2
MDF (cyclic)	5.9	−0%	10.4	4.4	10.4	4.4
$q = (0.1, 0.3, 0.3, 0.3)$						
RVC	6.5		6.1	5.6	8.3	5.7
FC departure times 10, 10 seconds	8.0	+23%	5.2	8.3	8.3	8.3
XHC	7.7	+18%	6.8	7.4	8.7	7.4
XHC(1)	6.5	+0%	5.4	6.2	7.3	6.2
XHC(2)	6.7	+3%	4.7	6.7	7.6	6.7
MDF (cyclic)	6.3	−3%	5.2	5.9	7.5	5.9

https://doi.org/10.1017/S026996480800034X Published online by Cambridge University Press

5.3. A More Complex Intersection (F12C4)

For the large intersection with 12 flows, computation of the optimal MDP strategy is practically impossible. Further, note that the combinations in this case no longer have an equal number of flows. This affect the results, since normally it is more profitable to serve combinations of more flows, because doing so maximizes the number of cars that depart in a slot (of course, only when cars are present at the queues).

5.3.1. Symmetric Arrival Rates (F12C4).

Table 3 gives the results for varying workloads at a F12C4 intersection in the case that all flows have identical arrival intensities. The RVC heuristic outperforms all other strategies. At a load of 0.8, the difference with the best anticipative exhaustive control is already quite large (28%).

Although the arrival rates per flow are identical (0.1, 0.15, and 0.2 for a load of 0.4, 0.6, and 0.8, respectively), the mean waiting times differ per combination. Combinations C_1 and C_3 are "thicker" than combinations C_2 and C_4, since the latter two combinations have only two flows each, whereas C_1 and C_3 consists of four flows each. Therefore, C_1 and C_3 experience a lower waiting time than combinations C_2 and C_4. The detailed results are reported in Table 4 for the case of $\rho = 0.8$.

5.3.2. Asymmetric Arrival Rates (F12C4).

An interesting asymmetric case for F12C4 is where the flows in C_2 have a third of the load of the other flows. So, for the flows in C_2, the arrival probability is 0.08; for

Table 3. Overall Mean Waiting Time (in s) at Different Loads for symmetric F12C4

Rule	$\rho = 0.4$		$\rho = 0.6$		$\rho = 0.8$	
RVC	13.5		19.3		41.8	
FC	15.0	+11%	23.7	+23%	50.5	+21%
XHC	19.2	+42%	33.4	+73%	89.8	+115%
XHC(1)	14.9	+10%	25.1	+30%	70.1	+68%
XHC(2)	13.5	+0%	19.6	+2%	53.3	+28%
FC cycle length (in s)	32		40		88	
FC departure times (in s)	(6, 6, 6, 6)		(8, 8, 8, 8)		(20, 20, 20, 20)	

Table 4. Mean Waiting Times (in s) for Symmetric F12C4 at $\rho = 0.8$

Rule	$E(W)$ Overall		$E(W)$ C_1, C_3	$E(W)$ C_2, C_4
RVC	41.8		37.4	50.6
FC departure times 20, 20, 20, 20 s	50.5	+21%	50.5	50.4
XHC	89.8	+115%	88.5	92.4
XHC(1)	70.1	+68%	68.9	72.4
XHC(2)	53.3	+28%	52.1	55.8

https://doi.org/10.1017/S026996480800034X Published online by Cambridge University Press

TABLE 5. Mean Waiting Times (in s) for an Asymmetric F12C4 Case at $\rho = 0.8$

Rule	$E(W)$ Overall		$E(W)$ C_1, C_3	$E(W)$ C_2	$E(W)$ C_4
RVC	39.4		34.9	66.6	48.7
FC departure times 22, 8, 22, 22 s	47.1	+20%	45.6	69.4	45.6
XHC	85.1	+116%	82.8	107.8	86.9
XHC(1)	66.6	+69%	64.6	84.8	68.3
XHC(2)	50.5	+28%	48.8	65.0	52.6

the other flows, it is 0.24. Then the overall load is again 0.8. The results are reported m. Table 5. Intuitively, one expects the flows in C_2 to suffer from high waiting times, since its load is low and it consists of two flows only. However, as we see, the waiting time for C_2 is (considerably) lower than under FC, XHC, and XHC(l).

6. CONCLUSION

We have presented a new approach for the dynamic control of the traffic lights at an isolated intersection based on an MDP decomposition technique for multidimensional decision problems. The results are quite good even in situations close to saturation. Particularly for complicated intersections (i.e., with a larger numbers of combined flows), the heuristic outperforms existing strategies such as FC control and XH rules.

As will be shown in forthcoming articles, the decomposition technique is able to deal with additional arrival information. In addition, the technique can be used to dynamically control nearby intersections.

Acknowledgement

We thank Sandjai Bhulai for the fruitful discussions on the decomposition of multidimensional Markov decision processes.

References

1. Bhulai, S. (2004). Dynamic routing policies for multi-skill call centers. Technical report WS2004-11, Vrije Universiteit Amsterdam.
2. Darroch, J.N. (1964). On the traffic light queue. *Annals of Mathematical Statistics* 35(1): 380–388.
3. Farges, J.-L., Henry, J.-J., & Tufal, J. (1983). The PRODYN real-time traffic algorithm. In *Proceedings of the 4th IFAC Symposium on Transportation Systems*, pp. 307–312.
4. Higashikubo, M., Hinenoya, T., & Takeuchi, K. (1997). Traffic queue length measurement using an image processing sensor. *Sumitomo Electric Technical Review* 43: 64–68.
5. Krishnan, K.R. & Ott, T.J. (1986). State-dependent routing for telephone traffic: Theory and results. In *Proceedings of the 25th IEEE conference on decision and control, Athens, Greece*. New York: IEEE, pp. 2124–2128.
6. Krishan, K.R. & Ott, T.J. (1987). Joining the right queue: A Markov decision rule. In *Proceedings of the 26th IEEE conference on decision and control, Los Angeles*. New York: IEEE, pp. 1863–1868.
7. McNeill, D.R. (1965). A solution to the fixed cycle traffic light problem for compound poisson arrivals. *Journal of Applied Probability* 5: 624–635.

8. Miller, A.J. (1963). Settings for fixed-cycle traffic signals. *Operations Research* 14(4): 373–386.
9. Miller, A.J. (1963). A computer control system for traffic networks. In *Proceedings of the International Symposium on Traffic Theory*, pp. 200–220.
10. Newell, G.F. (1960). Queues for a fixed-cycle traffic light. *Annals of Mathematical Statistics* 31(3): 589–597.
11. Newell, G.F. (1965). Approximation methods for queues with applications to the fixed-cycle traffic light. *SIAM Review* 7(2): 223–240.
12. Norman, J.M. (1972). *Heuristic procedures in dynamic programming*. Manchester, UK: Manchester University Press.
13. Papageorgiou, M., Diakaki, C., Dinopoulou, V., Kotsialos, A., & Wang, Y. (2003). Review of road traffic control strategies. In *Proc. IEEE* 91: 2043–2067.
14. Prinsen, L.H.A. (2006). Personal communication.
15. Sassen, S.A.E., Tijms, H.C., & Nobel, R.D. (1997). A heuristic rule for routing customers to parallel queues. *Statistica Neerlandica* 51(1): 107–121.
16. Sen, S. & Head, L. (1997). Controlled optimization of phases at an intersection. *Transportation Science* 31: 5–17.
17. Tijms, H.C. (2003). *A first course in stochastic models*. New York: Wiley.
18. van Leeuwaarden, J.S.H. (2006). Delay analysis for the fixed cycle traffic light queue. *Transportation Science* 40(2): 189–199.
19. Vincent, R.A. & Young, C.P. (1986). Self-optimizing traffic signal control using microprocessor: The TRRL "MOVA" strategy for isolated intersections. *Traffic Engineering and Control* 27: 385–387.
20. Webster, F.V. (1958). Traffic signal settings. Road Research technical paper 39. Road Research Laboratory, London.
21. Wiering, M., van Veenen, J., Vreeken, J., & Koopman, A. (2004). Intelligent traffic light control. Technical report UU-CS-2004-029, Institute of Information and Computing Sciences, Utrecht University.
22. Wijngaard, J. (1979). Decomposition for dynamic programming in production and inventory control. *Engineering and Process Economics* 4: 385–388.