Linear programming formulations for queueing control problems with action decomposability

Ariel Waserhole, Jean-Philippe Gayon, Vincent Jost

Linear programming formulations for queueing control problems with action decomposability

Ariel Waserhole [1,2] Jean-Philippe Gayon[1] Vincent Jost[2]

[1] G-SCOP, Grenoble-INP [2] LIX CNRS, École Polytechnique Palaiseau

April 26, 2013

Abstract

We consider a special class of continuous-time Markov decision processes (CTMDP) that are action decomposable. An action-Decomposed CTMDP (D-CTMPD) typically models queueing control problems with several types of events. A sub-action and cost is associated to each type of event. The action space is then the Cartesian product of sub-action spaces. We first propose a new and natural Quadratic Programming (QP) formulation for CTMDPs and relate it to more classic Dynamic Programming (DP) and Linear Programming (LP) formulations. Then we focus on D-CTMDPs and introduce the class of decomposed randomized policies that will be shown to be dominant in the class of deterministic policies by a polyhedral argument. With this new class of policies, we are able formulate decomposed QP and LP with a number of variables linear in the number of types of events whereas in its classic version the number of variables grows exponentially. We then show how the decomposed LP formulation can solve a wider class of CTMDP that are quasi decomposable. Indeed it is possible to forbid any combination of sub-actions by adding (possibly many) constraints in the decomposed LP. We prove that, given a set of linear constraints added to the LP, determining whether there exists a deterministic policy solution is NP-complete. We also exhibit simple constraints that allow to forbid some specific combinations of sub-actions. Finally, a numerical study compares computation times of decomposed and non-decomposed formulations for both LP and DP algorithms.

Keywords: Continuous-time Markov decision process; Queueing control; Event-based dynamic programming; Linear programming.

1 Introduction

Different approaches exist to solve numerically a Continuous-Time Markov Decision Problem (CTMDP) that are based on optimality equations (or Bellman equations). The most popular method is the value iteration algorithm which is essentially a backward Dynamic Programming (DP). Another well known approach is to model a CTMDP as a Linear Programming (LP). LP based algorithms are slower than DP based algorithms. However, LP formulations offer the possibility to add very easily linear constraints on steady state probabilities, which is not the case of DP formulations.

Good introductions to CTMDPs can be found in the books of Puterman (1994), Bertsekas (2005) and Guo and Hernández-Lerma (2009).

In this paper, we consider a special class of CTMDPs that we call action Decomposed CTMDPs (D-CTMPDs). D-CTMDP typically model queueing control problems with several types of events (demand arrival, service end, failure, etc) and where a sub-action (admission, routing, repairing, etc) and also a cost is associated to each type of event. The class of D-CTMPD is related to the concept of event-based DP, first introduced by Koole (1998). Event-based DP is a systematic approach for deriving monotonicity results of optimal policies for various queueing and resource sharing models. Citing Koole: "Event-based DP deals with event operators, which can be seen as building blocks of the value function. Typically it associates an operator with each basic event in the system, such as an arrival at a queue, a service completion, etc. Event-based DP focuses on the underlying properties of the value and cost functions, and allows us to study many models at the same time." The event-based DP framework is strongly related to older works (see e.g. Lippman (1975); Weber and Stidham (1987)).

Apart from the ability to prove structural properties of the optimal policy, the event-based DP framework is also a very natural way to model many queueing control problems. In addition, it allows to reduce drastically the number of actions to be evaluated in the value iteration algorithm. The following example will be used throughout the paper to illustrate our approach and results and will be referred to as the dynamic pricing problem.

Example – Dynamic pricing in a multi-class M/M/1 queue. *Consider a single server with n different classes of clients that are price sensitive (see Figure 1). There is a finite buffer of size C for each client class. Clients of class $i \in I = \{1, \ldots, n\}$ arrive according to an independent Poisson*

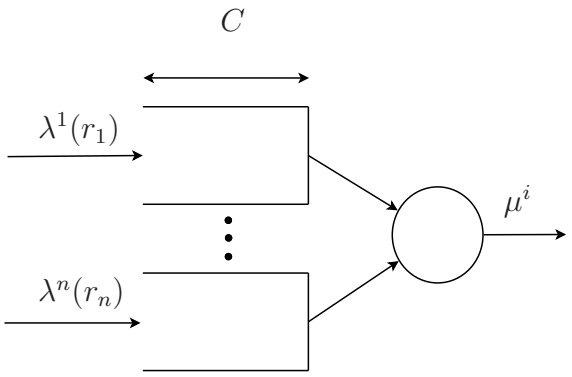

Figure 1: The multi-class M/M/1 queue with dynamic pricing.

process with rate $\lambda^i(r_i)$ where r_i is a price dynamically chosen in a finite set P of k prices. For clients of class i, the waiting cost per unit of time is b^i and the processing time is exponentially distributed with rate μ^i. At any time the decision maker has to set the entrance price for each class of clients and to decide which class of clients to serve with the objective to maximize the average reward, in the class of preemptive dynamic policies. This problem has been studied, among other works, by Maglaras (2006), Çil et al. (2011), and Li and Neely (2012).

For this example, the state and action spaces have respectively a cardinality of $C^n + 1$ and nk^n. However, the action selection process in the value iteration algorithm does not require to evaluate the nk^n actions. It is sufficient to evaluate at each iteration only $n(k+1)$ actions (k possibilities for

each class of customer and n possibilities for the class to be served). This property has been used intensively in the literature since the seminal paper of Lippman (1975). In the classic LP formulation of the dynamic pricing problem, that one can find in (Puterman, 1994) for instance, the number of variables grows exponentially with the number of possible prices. In this paper, we will show that the LP can be reformulated in a way such that the number of variables grows linearly with the number of possible prices.

Our contributions can be summarized as follows. We first propose a new and natural Quadratic Programming (QP) formulation for CTMDP and relate it to more classic Dynamic Programming (DP) and Linear Programming (LP) formulations. Then, we introduce the class of D-CTMPDs which is probably the largest class of CTMDPs for which the event-based DP approach can be used. We also introduce the class of decomposed randomized policies that will be shown to be dominant among randomized policies. With these new policies, we are able to reformulate the QP and the LP with a number of variables growing linearly with the number of event types. With respect to the decomposed DP, this LP formulation is really simple to write and does not need the uniformization process necessary for the DP formulation which is sometimes source of errors and waste of time. Moreover, it allows to use generic LP related techniques such that sensitivity analysis (Filippi, 2011) or approximate linear programming (Dos Santos Eleutrio, 2009).

Another contribution of the paper is to show how to forbid some actions while preserving the structure of the decomposed LP. If some actions (combinations of sub-actions) are forbidden, the DP cannot be decomposed anymore. In the dynamic pricing example, imagine that we want a low price to be selected for at least one class of customer. In the (non-decomposed) DP formulation, it is easy to add this constraint by removing all actions that does not contain a low price. However, it is not possible to decompose anymore the DP, in our opinion. In the decomposed LP formulation, we show how it is possible to remove this action and other combinations of actions by adding simple linear constraints. We also discuss the generic problem of reducing arbitrarily the action space by adding a set of linear constraints in the decomposed LP. Not surprisingly, this problem is difficult and is not appropriate if many actions are removed arbitrarily. When new constraints are added in the decomposed LP, it is also not clear whether deterministic policies remain dominant or not. We even prove that, given a set of linear constraints added to the LP, determining whether there exists a deterministic policy solution is NP-complete. However, for some simple action reductions, we show that deterministic policies remain dominant. We finally present numerical results comparing LP and DP formulations (decomposed or not).

Before presenting the organization of the paper, we mention briefly some related literature addressing MDP with large state space (Bertsekas and Castañon, 1989; Tsitsiklis and Van Roy, 1996) which tries to fight the curse of dimensionality, *i.e.* the exponential growth of the state space size with some parameter of the problem. Another issue, less tackled, appears when the state space is relatively small but the action space is very large. Hu et al. (2007) proposes a randomized search method for solving infinite horizon discounted cost discrete-time MDP for uncountable action spaces.

The rest of the paper is organized as follows. We first address the average cost problem. Section 2 reminds the definition of a CTMDP and formulates it as a QP. We also link the QP formulation with classic LP and DP formulations. In Section 3, we define properly D-CTMDP and show how the DP and LP can be decomposed for this class of problems. Section 4 discusses the problem of reducing the action space by adding valid constraints in the decomposed LP. Section 5 compares numerically computation times of decomposed and non-decomposed formulations for both LP and DP algorithms,

for the dynamic pricing problem. Finally, Section 6 explains how our results can be adapted to a discounted cost criterion.

2 Continuous-Time Markov Decision Processes

In this section, we remind some classic results on CTMDPs that will be useful to present our contributions.

2.1 Definition

We slightly adapt the definition of a CTMDP given by Guo and Hernández-Lerma (2009). A CTMDP is a stochastic control problem defined by a 5-tuple

$$\left\{S,\ A,\ \lambda_{s,t}(a),\ h_s(a), r_{s,t}(a)\right\}$$

with the following components:

- S is a finite set of states;
- A is a finite set of actions, $A(s)$ are the actions available from state $s \in S$;
- $\lambda_{s,t}(a)$ is the transition rate to go from state s to state t with action $a \in A(s)$;
- $h_s(a)$ is the reward rate while staying in state s with action $a \in A(s)$;
- $r_{s,t}(a)$ is the instant reward to go from state s to state t with action $a \in A(s)$.

Instant rewards $r_{s,t}(a)$ can be included in the reward rates $h_s(a)$ by an easy transformation and reciprocally. Therefore for ease of presentation, we will use the aggregated reward rate $\tilde{h}_s(a) := h_s(a) + \sum_{t \in S} \lambda_{s,t}(a) r_{s,t}(a)$.

We first consider the objective of maximizing the average reward over an infinite horizon, the discounted case will be discussed in the Section 6. We restrict our attention to stationary policies which are dominant for this problem (Bertsekas, 2005) and define the following policy classes:

- A (*randomized* stationary) *policy* p sets for each state $s \in S$ the probability $p_s(a)$ to select action $a \in A(s)$ with $\sum_{a \in A(s)} p_s(a) = 1$.
- A *deterministic policy* p sets for each state s one action to select: $\forall s \in S$, $\exists a \in A(s)$ such that $p_s(a) = 1$ and $\forall b \in A(s) \setminus \{a\},\ p_s(b) = 0$.
- A *strictly randomized policy* has at least one state where the action is chosen randomly: $\exists s \in S$, $\exists a \in A(s)$ such that $0 < p_s(a) < 1$.

The best average reward policy p^* with gain g^* is solution of the following Quadratic Program (QP) together with a vector ϖ^* (called "variant pi" or "pomega"). ϖ_s is to be interpreted as the stationary distribution of state $s \in S$ under policy p.

QP (1)

$$g^* = \max \quad \sum_{s\in S}\sum_{a\in A(s)} \tilde{h}_s(a)p_s(a)\varpi_s \tag{1a}$$

$$\text{s. t.} \quad \sum_{a\in A(s)}\sum_{t\in S}\lambda_{s,t}(a)p_s(a)\varpi_s = \sum_{t\in S}\sum_{a\in A(t)}\lambda_{t,s}(a)p_t(a)\varpi_t, \qquad \forall s\in S, \tag{1b}$$

$$\sum_{s\in S}\varpi_s = 1, \tag{1c}$$

$$\varpi_s \geq 0, \tag{1d}$$

$$\sum_{a\in A(s)} p_s(a) = 1, \qquad \forall s\in S, \tag{1e}$$

$$p_s(a) \geq 0. \tag{1f}$$

QP (1) is a natural way to formulate a stationary MDP. It is easy to see that this formulation solves the best average stationary reward policy. First Equations (1e) and (1f) define the space of admissible stationnary randomized policies p. Secondly for a given policy p, Equations (1b)-(1d) compute the stationary distribution ϖ of the induced Continuous-Time Markov Chain with gain expressed by Equation (1a).

As we will show in the Section 2.3, the usual LP formulation (4) can be derived directly from QP (1) by simple substitutions of variables. However, in the literature it is classically derived from the linearization of dynamic programming optimality equations given in the following subsection.

2.2 Optimality equations

A CTMDP can be transformed into a discrete-time MDP through a uniformization process (Lippman, 1975). In state s, we uniformize the CTMDP with rate $\Lambda_s := \sum_{t\in S}\Lambda_{s,t}$ where $\Lambda_{s,t} := \max_{a\in A(s)}\lambda_{s,t}(a)$. This uniformization rate simplifies greatly the optimality equations and linear programs.

In the rest of the paper, under relatively general conditions (typically for unichain models, see *e.g.* Bertsekas (2005)) we assume that the optimal average reward g^* is independent of the initial state and is the unique solution together with an associated differential reward vector v^* that satisfies the Bellman's optimality equations:

$$\frac{g}{\Lambda_s} = T\Big(v(s)\Big) - v(s), \qquad \forall s\in S, \tag{2}$$

with operator T defined as

$$T\Big(v(s)\Big) := \max_{a\in A(s)}\left\{\frac{1}{\Lambda_s}\left(\tilde{h}_s(a) + \sum_{t\in S}\Big[\lambda_{s,t}(a)v(t) + \Big(\Lambda_{s,t} - \lambda_{s,t}(a)\Big)v(s)\Big]\right)\right\}. \tag{3}$$

These optimality equations can be used to compute the best MDP policy. For instance, the value iteration algorithm roughly consists in defining a sequence of value function $v_{n+1} = T(v_n)$ that provides a stationary ϵ-optimal deterministic policy in a number of iterations depending on the desired ϵ.

2.3 Linear programming formulation

Since Manne (1960) we know that it is possible to compute the best average reward policy through a LP formulation. From Equations (2) and (3), one can show that the optimal average reward g^* is the solution of the following program:

$$\begin{aligned} g^* = \min \quad & g \\ \text{s.t.} \quad & \frac{g}{\Lambda_s} \geq T\Big(v(s)\Big) - v(s), && \forall s \in S, \\ & v(s) \in \mathbb{R}, \quad g \in \mathbb{R}. \end{aligned}$$

Linearizing the max function in operator T leads to the following LP formulation and its dual counterpart.

Primal LP

$$\begin{aligned} g^* = \min \quad & g \\ \text{s.t.} \quad & g \geq \tilde{h}_s(a) + \sum_{t \in S} \lambda_{s,t}(a)\Big(v(t) - v(s)\Big), && \forall s \in S,\ \forall a \in A(s), \\ & v(s) \in \mathbb{R}, \quad g \in \mathbb{R}. \end{aligned}$$

Dual LP (4)

$$g^* = \max \quad \sum_{s \in S} \sum_{a \in A} \tilde{h}_s(a)\pi_s(a) \tag{4a}$$

$$\text{s.t.} \quad \sum_{a \in A(s)} \sum_{t \in S} \lambda_{s,t}(a)\pi_s(a) = \sum_{t \in S} \sum_{a \in A(t)} \lambda_{t,s}(a)\pi_t(a), \qquad \forall s \in S, \tag{4b}$$

$$\sum_{s \in S} \sum_{a \in A(s)} \pi_s(a) = 1, \tag{4c}$$

$$\pi_s(a) \geq 0. \tag{4d}$$

The advantage of the dual formulation is to allow a simple interpretation: variable $\pi_s(a)$ is the average proportion of time spent in state s choosing action a.

A simple way to show that the dual LP (4) solves the best average reward policy is to see that it can be obtained from QP (1) by the following substitutions of variables:

$$\pi_s(a) = p_s(a)\varpi_s, \quad \forall s \in S,\ \forall a \in A(s).$$

Indeed, any solution (p, ϖ) of QP (1) can be mapped into a solution π of dual LP (4) with same expected gain thanks to the following mapping:

$$(p, \varpi) \mapsto \pi = \Big(\pi_s(a) = p_s(a)\varpi_s\Big).$$

For the opposite direction, there exists several "equivalent" mappings preserving the gain. Their differences lie in the decisions taken in unreachable states ($\varpi_s = 0$). We exhibit one:

$$\pi \mapsto (p, \varpi) = \left(p_s(a) = \left\{ \begin{array}{ll} \frac{\pi_s(a)}{\varpi_s}, & \text{if } \varpi_s \neq 0 \\ 1, & \text{if } \varpi_s = 0 \text{ and } a = a^1 \\ 0, & \text{otherwise} \end{array} \right\}, \varpi_s = \sum_{a \in A(s)} \pi_s(a) \right).$$

Since any solution of QP (1) can be mapped to a solution of dual LP (4) and conversely, in the sequel we overload the word *policy* as follows:

- We (abusively) call *(randomized) policy* a solution π of the dual LP (4).
- We say that π is a *deterministic policy* if it satisfies $\pi_s(a) \in \{0, \varpi_s = \sum_{a \in A(s)} \pi_s(a)\}$, $\forall s \in S$, $\forall a \in A(s)$.

3 Action Decomposed Continuous-Time Markov Decision Processes

3.1 Definition

An action Decomposed CTMDP (D-CTMDP) is a CTMDP such that:

- In each state $s \in S$, the action space can be written as the Cartesian product of $n_s \geq 1$ sub-action sets: $A(s) = A_1(s) \times \ldots \times A_{n_s}(s)$. An action $a \in A(s)$ is then composed by n_s sub-actions $(a_1, \ldots, a_{n_s})$ where $a_i \in A_i(s)$.
- *Sub-action* a_i increases the transition rate from s to t by $\lambda^i_{s,t}(a_i)$, the reward rate by $h^i_s(a_i)$ and the instant reward rate by $r^i_{s,t}(a_i)$.
  - The resulting transition rate from s to t is then $\lambda_{s,t}(a) = \sum_{i=1}^{n_s} \lambda^i_{s,t}(a_i)$.
  - The resulting aggregated reward rate in state s when action a is taken is then $\tilde{h}_s(a) = \sum_{i=1}^{n_s} \tilde{h}^i_s(a_i)$ with $\tilde{h}^i_s(a_i) = h_s(a_i) + \sum_{t \in S} \lambda^i_{s,t}(a_i) r^i_{s,t}(a_i)$.

D-CTMDPs typically model queueing control problems with several types of events (demand arrival, service end, failure, etc), an action associated to each type of event (admission control, routing, repairing, etc) and also a cost associated to each type of event. Event-based DP, as defined by Koole (1998), is included in the class of D-CTMDPs.

For ease of notation, we assume without loss of generality that each state $s \in S$ has exactly $n_s = n$ independent sub-action sets, with $I = \{1, \cdots, n\}$, and that each sub-action set $A_i(s)$ contains exactly k sub-actions.

We introduce the concept of decomposed policy.

- A *(randomized) decomposed policy* is a vector $\mathring{p} = ((\mathring{p}^1_s, \ldots, \mathring{p}^n_s),\ s \in S)$ such that for each state s there is a probability $\mathring{p}^i_s(a_i)$ to select sub-action $a_i \in A_i(s)$ with $\sum_{a_i \in A_i(s)} \mathring{p}^i_s(a_i) = 1$, $\forall s \in S$, $\forall i \in I$. The probability to choose action $a = (a_1, \cdots, a_n)$ in state s is then $p_s(a) = \prod_{i \in I} \mathring{p}^i_s(a_i)$.

- A decomposed policy $\mathring{p}$ is said *deterministic* if $\forall s \in S,\ \forall i \in I,\ \exists a_i \in A_i(s)$ such that $\mathring{p}^i_s(a_i) = 1$ and $\forall b_i \in A_i(s) \setminus \{a_i\},\ \mathring{p}^i_s(b_i) = 0$. In other words, $\mathring{p}$ selects one sub-action for each state s and each sub-action set A_i.

In the following we will see that decomposed policies are dominant for D-CTMDPs. It is interesting since a decomposed policy $\mathring{p}$ is described in a much more compact way than a classic policy p.

Simply applying the definition, one can check that the best average reward decomposed policy $\mathring{p}^*$ is solution of the following quadratic program where $\mathring{\varpi}_s$ is to be interpreted as the stationary distribution of state $s \in S$.

Decomposed QP (5)

$$g^* = \max \quad \sum_{s\in S}\sum_{i\in I}\sum_{a_i\in A_i(s)} \tilde{h}^i_s(a_i)\mathring{p}^i_s(a_i)\mathring{\varpi}_s \tag{5a}$$

$$\text{s.t.} \quad \sum_{i\in I}\sum_{a_i\in A_i(s)}\sum_{t\in S} \lambda^i_{s,t}(a_i)\mathring{p}^i_s(a_i)\mathring{\varpi}_s = \sum_{t\in S}\sum_{i\in I}\sum_{a_i\in A_i(t)} \lambda^i_{t,s}(a_i)\mathring{p}^i_t(a_i)\mathring{\varpi}_t, \qquad \forall s \in S, \tag{5b}$$

$$\sum_{s\in S} \mathring{\varpi}_s = 1, \tag{5c}$$

$$\mathring{\varpi}_s \geq 0, \tag{5d}$$

$$\sum_{a_i\in A_i(s)} \mathring{p}^i_s(a_i) = 1, \qquad \forall s \in S,\ \forall i \in I, \tag{5e}$$

$$\mathring{p}^i_s(a_i) \geq 0. \tag{5f}$$

Example – Dynamic pricing in a multi-class M/M/1 queue (D-CTMDP formulation)**.** *We continue the example started in the Section 1. This problem can be modeled as a D-CTMDP with state space* $S = \{(s_1, \dots, s_n) \mid s_i \leq C,\ \forall i \in I\}$. *In each state* $s \in S$, *there is* $n_s = (n+1)$ *sub-actions and an action can be written as* $a = (r_1, \dots, r_n, d)$ *with* r_i *the price decided to be offered to client class* i *and* d *the client class to process. The action space is then* $A = P^n \times D$ *with* $D = \{1, \dots, n\}$. *The waiting cost in state* $(s_1, \dots, s_n)$ *is independent of the action selected and is worth* $\sum_i h^i s_i$. *The reward rate incurred by sub-action* r_i *is* $\lambda_i(r_i)r^i$. *Let the function* $\mathbb{1}_b$ *equals 1 if boolean expression* b *is worth true and* 0 *otherwise. The resulting aggregated reward rate in state* $s = (s_1, \dots, s_n)$ *when action* $a = (r_1, \dots, r_n, d)$ *is selected is then* $\tilde{h}_s(a) = \sum_{i=1}^n \tilde{h}^i_s(r_i)$ *with* $\tilde{h}^i_s(r_i) = h^i s_i + \mathbb{1}_{s_i<C}\lambda_i(r_i)r^i$.

For this example, the cardinality of the state space and the action space are respectively $|S| = (C+1)^n$ *and* $|A| = k^n n$.

3.2 Optimality equations

Optimality equations for CTMDPs can be rewritten in the context of a D-CTMDPs to take advantage of decomposition properties. Let $\Lambda^i_{s,t} = \max_{a_i\in A_i(s)} \lambda^i_{s,t}(a_i)$. The uniformization rate is again $\Lambda_s = \sum_{t\in S} \Lambda_{s,t}$ where $\Lambda_{s,t}$, as defined previous section, can be rewritten as follows:

$$\Lambda_{s,t} = \max_{a\in A(s)} \lambda_{s,t}(a) = \max_{\substack{(a_1,\ \dots,\ a_n) \\ \in\ A_1(s)\times\dots\times A_n(s)}} \sum_{i\in I} \lambda^i_{s,t}(a_i) = \sum_{i\in I} \max_{a_i\in A_i(s)} \lambda^i_{s,t}(a_i) = \sum_{i\in I} \Lambda^i_{s,t}.$$

Operator T as defined in Equation (3) can be rewritten as:

$$T\Big(v(s)\Big) = \max_{\substack{(a_1,\ \ldots,\ a_n) \\ \in\ A_1(s)\times\ldots\times A_n(s)}} \left\{ \frac{1}{\Lambda_s} \sum_{i\in I} \left(\tilde{h}^i_s(a_i) + \sum_{t\in S} \left[\lambda^i_{s,t}(a_i) v(t) + \Big(\Lambda^i_{s,t} - \lambda^i_{s,t}(a_i)\Big) v(s) \right] \right) \right\}. \tag{6}$$

That we can decompose as:

$$T\Big(v(s)\Big) = \frac{1}{\Lambda_s} \sum_{i\in I} \left(\max_{a_i\in A_i(s)} \left\{ \tilde{h}^i_s(a_i) + \sum_{t\in S} \left[\lambda^i_{s,t}(a_i) v(t) + \Big(\Lambda^i_{s,t} - \lambda^i_{s,t}(a_i)\Big) v(s) \right] \right\} \right). \tag{7}$$

The value iteration algorithm is much more efficient if T is expressed as in the latter equation. Experimental results presented in Section 5 show it clearly. Indeed computing the maximum requires n^k evaluations in Equation (6) and nk in Equation (7). To the best of our knowledge, this decomposition property of operator T is used in many queueing control problems (see Koole (1998) and related papers) but has not been formalized as generally as in this paper.

Example – Dynamic pricing in a multi-class M/M/1 queue (DP approach)**.** *We can now write down the optimality equations. We use the following uniformization: let* $\Lambda = \sum_{i\in I} \Lambda^i + \Delta$ *with* $\Lambda^i = \max_{r_i\in P}\{\lambda^i(r_i)\}$ *and* $\Delta = \max_{i\in I}\{\mu^i\}$.

For a state $s = (s_1, \ldots, s_n)$ *and with* e_i *the unitvector of the* i^{th} *coordinate, the operator* T *for classic optimality equations can be defined as follows:*

$$T\Big(v(s)\Big) = \max_{(r_1,\ldots,r_n,d)\in A} \left\{ \sum_{i\in I} \Big[\tilde{h}^i(r_i) + \mathbb{1}_{s_i<C} \lambda^i(r_i) v(s+e_i) \Big] \right.$$
$$\left. + \mathbb{1}_{s_d>0} \mu^d v(s-e_d) + \left(\Lambda - \sum_{i\in I} \mathbb{1}_{s_i<C} \lambda^i(r_i) + \Delta - \mathbb{1}_{s_d>0} \mu^d \right) v(s) \right\}.$$

Since we are dealing with a D-CTMDP, operator T *can also be decomposed as:*

$$T\Big(v(s)\Big) = \frac{1}{\Lambda} \left(\sum_{i\in I} \left[\tilde{h}^i(r_i) + \max_{\substack{r_i\in P \\ s_i<C}} \left\{ \lambda^i(r_i) v(s+e_i) + \Big(\Lambda^i - \lambda^i(r_i)\Big) v(s) \right\} \right] \right.$$
$$\left. + \max_{\substack{d\in D \\ s_d>0}} \left\{ \mu^d v(s-e_d) + (\Delta - \mu^d) v(s) \right\} \right).$$

3.3 LP formulation

Let $\mathring{\pi}^i_s(a_i)$ be interpreted as the average proportion of time spent in state s choosing action $a_i \in A_i(s)$ among all sub-actions $A_i(s)$. From decomposed QP (5) we can build the LP (8) formulation with simple substitutions of variable:

$$\mathring{\pi}^i_s(a_i) = \mathring{\pi}^i_s(a_i) \mathring{\varpi}_s, \quad \forall s \in S,\ \forall i \in I,\ \forall a_i \in A_i(s).$$

We obtain that g^* is the solution of the following LP formulation:

Decomposed Dual LP (8)

$$g^* = \max \quad \sum_{s\in S}\sum_{i\in I}\sum_{a_i\in A_i(s)} \tilde{h}^i_s(a_i)\mathring{\pi}^i_s(a_i) \tag{8a}$$

$$\text{s.t.} \quad \sum_{i\in I}\sum_{a_i\in A_i(s)}\sum_{t\in S}\lambda^i_{s,t}(a_i)\mathring{\pi}^i_s(a_i) = \sum_{t\in S}\sum_{i\in I}\sum_{a_i\in A_i(t)}\lambda^i_{t,s}(a_i)\mathring{\pi}^i_t(a_i), \qquad \forall s\in S, \tag{8b}$$

$$\sum_{a_i\in A_i(s)}\mathring{\pi}^i_s(a_i) = \mathring{\varpi}_s, \qquad \forall s\in S,\ \forall i\in I, \tag{8c}$$

$$\sum_{s\in S}\mathring{\varpi}_s = 1, \tag{8d}$$

$$\mathring{\pi}^i_s(a_i) \geq 0, \quad \mathring{\varpi}_s \geq 0. \tag{8e}$$

The decomposed dual LP formulation (8) has $|S|(kn+1)$ variables and $|S|((k+1)n+2)+1$ constraints. It is much less than the classic dual LP formulation (4) that has $|S|k^n$ variables and $|S|(k^n+1)$ constraints.

Lemma 1. *Any solution $(\mathring{p}, \mathring{\varpi})$ of decomposed QP (1) can be mapped into a solution $(\mathring{\pi}, \mathring{\varpi})$ of decomposed dual LP (4) with same expected gain thanks to the following mapping:*

$$(\mathring{p}, \mathring{\varpi}) \to (\mathring{\pi}, \mathring{\varpi}) = \left(\mathring{\pi}^i_s(a_i) = \mathring{\pi}^i_s(a_i)\mathring{\varpi}_s,\ \mathring{\varpi}\right).$$

For the opposite direction, there exists several "equivalent" mappings preserving the gain. Their differences lie in the decisions taken in unreachable states ($\varpi_s = 0$). We exhibit one:

$$(\mathring{\pi}, \mathring{\varpi}) \mapsto (\mathring{p}, \mathring{\varpi}) = \left(\mathring{p}^i_s(a_i) = \left\{\begin{array}{ll} \frac{\mathring{\pi}^i_s(a_i)}{\mathring{\varpi}_s}, & \text{if } \mathring{\varpi}_s \neq 0 \\ 1, & \text{if } \mathring{\varpi}_s = 0 \text{ and } a_i = a_1 \\ 0, & \text{otherwise} \end{array}\right\},\ \mathring{\varpi}\right).$$

□

Since any solution of the decomposed QP (5) can be matched to a solution of decomposed dual LP (8) and conversely (Lemma 1), in the sequel we again overload the word *policy* as follows:

- We (abusively) call *(randomized) decomposed policy* a solution $(\mathring{\varpi}, \mathring{\pi})$ of the decomposed dual LP (8).
- We say that $(\mathring{\varpi}, \mathring{\pi})$ is a *deterministic policy* if it satisfies $\mathring{\pi}^i_s(a_i) \in \{0, \mathring{\varpi}_s\},\ \forall s\in S,\ \forall i\in I,\ \forall a_i \in A_i(s)$.

Dualizing the decomposed dual LP (8), we obtain the following primal version:

Decomposed Primal LP (9)

$$
\begin{aligned}
g^* = \min \quad & g \\
\text{s.t.} \quad & m(s,i) \geq \tilde{h}^i_s(a_i) + \sum_{t\in S} \lambda^i_{s,t}(a_i)\Big(v(t) - v(s)\Big), && \forall s \in S,\ \forall i \in I,\ \forall a_i \in A_i(s), \\
& g \geq \sum_{i\in I} m(s,i), && \forall s \in S, \\
& m(s,i) \in \mathbb{R}, \quad v(s) \in \mathbb{R}, \quad g \in \mathbb{R}.
\end{aligned}
$$

Note that the decomposed primal LP (9) could have also been obtained using the optimality equations (7). Indeed, under some general conditions (Bertsekas, 2005), the optimal average reward g^* is independent from the initial state and together with an associated differential cost vector v^* it satisfies the optimality equations (7). The optimal average reward g^* is hence the solution of the following equations:

$$
\begin{aligned}
g^* = \min \quad & g \\
\text{s.t.} \quad & \frac{g}{\Lambda_s} \geq T\Big(v(s)\Big) - v(s), && \forall s \in S.
\end{aligned}
$$

That can be reformulated using decomposability to have:

$$
\begin{aligned}
g^* &= \max_{s\in S} \Big\{ \Lambda_s \Big(T(v(s)) - v(s) \Big) \Big\} \\
&= \max_{s\in S} \left\{ \sum_{i\in I} \left(\max_{a_i \in A_i(s)} \left\{ \tilde{h}^i_s(a_i) + \sum_{t\in S} \Big[\lambda^i_{s,t}(a_i) v(t) + \Big(\Lambda^i_{s,t} - \lambda^i_{s,t}(a_i) \Big) v(s) \Big] - \Lambda^i_s v(s) \right\} \right) \right\} \\
&= \max_{s\in S} \left\{ \sum_{i\in I} \left(\max_{a_i \in A_i(s)} \left\{ \tilde{h}^i_s(a_i) + \sum_{t\in S} \lambda^i_{s,t}(a_i) \Big(v(t) - v(s) \Big) \right\} \right) \right\}. \qquad (10)
\end{aligned}
$$

The LP (9) can be obtained from Equation (10) using Lemma 3 given in appendix.

Example – Dynamic pricing in a multi-class M/M/1 queue (LP approach)**.** *With $a = (r_1, \ldots, r_n, d) \in A$, we can now formulate its classic dual LP formulation:*

$$
\begin{aligned}
\max \quad & \sum_{s\in S} \sum_{a\in A} \left(\sum_{i=1}^{n} \tilde{h}^i(r_i) \right) \pi_s(a) \\
s.t. \quad & \sum_{a\in A} \left(\mathbb{1}_{s_d > 0} \mu^d + \sum_{i=1}^{n} \mathbb{1}_{s_i < C} \lambda^i(r_i) \right) \pi_s(a) \\
& \quad = \sum_{a\in A} \left(\sum_{i=1}^{n} \mathbb{1}_{s_i > 0} \lambda^i(r_i) \pi_{s-e_i}(a) + \mathbb{1}_{s_d < C} \mu^d \pi_{s+e_d}(a) \right), && \forall s \in S, \\
& \sum_{s\in S} \sum_{a\in A} \pi_s(a) = 1, \\
& \mathring{\pi}_s(a) \geq 0.
\end{aligned}
$$

And its decomposed Dual LP formulation:

$$
\begin{aligned}
\max \quad & \sum_{s\in S}\sum_{i=1}^{n} \tilde{h}^i(r_i)\mathring{\pi}^i_s(r_i) \\
s.t. \quad & = \sum_{i\in I}\sum_{r_i\in P} \mathbb{1}_{s_i>0}\lambda^i(r_i)\mathring{\pi}^i_{s-e_i}(r_i) + \sum_{d\in D} \mathbb{1}_{s_d<C}\mu^d\mathring{\pi}_{s+e_d}(d), && \forall s\in S, \\
& \sum_{r_i\in P} \mathring{\pi}^i_s(r_i) = \mathring{\varpi}_s, && \forall s\in S,\ \forall i\in I, \\
& \sum_{d\in D} \mathring{\pi}_s(d) = \mathring{\varpi}_s, && \forall s\in S, \\
& \sum_{s\in S} \mathring{\varpi}_s = 1, \\
& \mathring{\pi}^i_s(r_i)\geq 0, \quad \mathring{\pi}_s(d)\geq 0, \quad \mathring{\varpi}_s\geq 0.
\end{aligned}
$$

3.4 Polyhedral results

Seeing the decomposed dual LP (8) as a reformulation of the decomposed QP (5), see Lemma 1, it is clear that it gives a policy maximizing the average reward. However, it doesn't provide any structure of optimal solutions. For this purpose, Lemma 2 gives two mappings linking classic and decomposed policies that are used in Theorem 1 to prove polyhedral results showing the dominance of deterministic policies. This means that the simplex algorithm on the decomposed dual LP (8) will return the best average reward deterministic policy.

Lemma 2. *Let $a(i)$ be the i^{th} coordinate of vector a. The following policy mappings preserve the strictly randomized and deterministic properties:*

$$
D:\ p\mapsto \mathring{p}=\left(\mathring{p}^i_s(a_i)=\sum_{a\in A(s)/a(i)=a_i} p_s(a)\right);\qquad D^{-1}:\ \mathring{p}\mapsto p=\left(p_s(a_1,\ldots,a_n)=\prod_{i\in I}\mathring{p}^i_s(a_i)\right).
$$

Moreover:
(a) D is linear.
(b) The following policy transformations preserve moreover the excepted gain:

1. $(p,\varpi)\mapsto(\mathring{p},\mathring{\varpi})=(D(p),\varpi)$;
2. $\pi\mapsto(\mathring{\pi},\mathring{\varpi})=(D(\pi),\mathring{\varpi}_s=\sum_{a\in A(s)}\pi_s(a))$;
3. $(\mathring{p},\mathring{\varpi})\mapsto(p,\varpi)=(D^{-1}(\mathring{p}),\mathring{\varpi})$;
4. $(\mathring{\pi},\mathring{\varpi})\mapsto(\pi,\varpi)=(D^{-1}(\mathring{\pi}),\mathring{\varpi})$.

□

Theorem 1. *The best decomposed CTMDP average reward policy is solution of the decomposed dual LP (8). Equations (8b)-(8e) describe the convex hull of deterministic policies.*

Proof: We call P the polytope defined by constraints (8b)-(8e). From Lemma 1 we know that all policies are in P. To prove that vertices of P are deterministic policies we use the characterization that a vertex of a polytope is the unique optimal solution for some objective.

Assume that $(\mathring{\pi}, \mathring{\varpi})$ is a strictly randomized decomposed policy, optimal solution with gain $\mathring{g}$ of the decomposed dual LP (8) for some objective $\tilde{h}_o$. From Lemma 2 we know that there exists a strictly randomized non decomposed policy (π, ϖ) with same expected gain. Deterministic policies are dominant in non decomposed models, therefore there exists a deterministic policy (π^*, ϖ^*) with gain $g^* \geq \mathring{g}$. From Lemma 2 we can convert (π^*, ϖ^*) into a deterministic decomposed policy $(\mathring{\pi}^*, \mathring{\varpi}^*)$ with same expected gain $\mathring{g}^* = g^* \geq \mathring{g}$. Since $(\mathring{\pi}, \mathring{\varpi})$ is optimal we have then $\mathring{g}^* = g^*$ which means that $(\mathring{\pi}, \mathring{\varpi})$ is not the unique optimal solution for objective $\tilde{h}_o$. Therefore a strictly decomposed randomized policy can't be a vertex of the decomposed LP (8) and P is the convex hull of deterministic policies. □

3.5 Benefits of decomposed LP

First, recall that with the use of action decomposability, decomposed LP (8) allows to have a complexity polynomial in the number of independent sub-action sets: $|S|(kn+1)$ variables and $|S|((k+1)n+2)$ constraints for the dual whereas in the classic it grows exponentially: $|S|k^n$ variables and $|S|(k^n+1)$ constraints. In the Section 5 we will see that it has a substantial impact on the computation time.

Secondly, even if the LP (8) is slower to solve than DP (7), as shown experimentally in the Section 5, this mathematical programming approach offers some advantages. First, LP formulations can help to characterize the polyhedral structure of discrete optimization problems, see Büyüktahtakin (2011). Secondly, there is in the LP literature generic methods directly applicable such as sensitive analysis, see Filippi (2011), or approximate linear programming techniques, see Dos Santos Eleutrio (2009). Another interesting advantage is that the dual LP (8) is really simple to write and does not need the uniformization necessary to the DP (7) which is sometimes source of waste of time and errors.

Finally, a big benefit of the LP formulation is the ability to add extra constraints that are not known possible to consider in the DP. A classic constraint that is known possible to add only in the LP formulation is to restrain the stationary distribution on a subset $T \subset S$ of states to be greater than a parameter q, for instance to force a quality of service:

$$\sum_{s \in T} \varpi_s \geq q.$$

Nevertheless, we have to be aware that such constraint can enforce strictly randomized policies as optimal solutions. In the next section we discuss constraints that preserve the dominance of deterministic policies.

4 Using decomposed LP to tackle a wider class of large action space MDP

4.1 Constraining action space, preserving decomposability

In decomposed models we can lose hand on certain action incompatibilities we had on classic ones:

Example – Dynamic pricing in a multi-class M/M/1 queue (Adding extra constraints)**.** *Let say we have two prices* high h *and* low l *for the n classes of clients. In a state s we have then the following set of actions: $A(s) = P(s) \times D(s)$ where $P(s) = \prod_{i=1}^{n} P_i(s)$ and $P_i(s) = \{h_i, l_i\}$. Assume that, for some marketing reasons, there needs to be at least one low price selected. In the non decomposed model, it is easily expressible by a new space of action $P'(s) = P(s) \setminus \{(h_1, \dots, h_n)\}$ removing the action where all high prices are selected. However, it is not possible to use decomposition anymore with this new action space, even though there is still some structure in the problem.*

When reducing the action space event-based DP techniques are inapplicable, but we can use some polyhedral properties of the LP formulation to model this action space reduction. In Theorem 2, we will show that it is possible to reduce the action space of any state $s \in S$ to $A'(s) \subseteq A(s)$ and to preserve the action decomposability advantage by adding a set of constraints to the decomposed dual LP (8). However, this theorem is not constructive and only proves the existence of such constraints. Moreover, in general the number of constraints necessary to characterize the admissible decomposed policies might be exponential and it would be finally less efficient than considering the non decomposed model (Proposition 1). Nevertheless, in some particular cases, one might be able to intuitively formulate some valid constraints that a decomposed policy should respect as in the following example.

Example – Dynamic pricing in a multi-class M/M/1 queue (Constraining in average)**.** *In* average *we can forbid solutions with all high prices (sub-action h_i) by selecting at most $n-1$ high prices (Equation (11a)), or at least one low price (sub-action l_i, Equation (11b)):*

$$\sum_{i=1}^{n} \mathring{p}_s^i(h_i) = \sum_{i=1}^{n} \frac{\mathring{\pi}_s^i(h_i)}{\mathring{\varpi}_s} \leq n-1, \tag{11a}$$

or

$$\sum_{i=1}^{n} \mathring{p}_s^i(l_i) = \sum_{i=1}^{n} \frac{\mathring{\pi}_s^i(l_i)}{\mathring{\varpi}_s} \geq 1. \tag{11b}$$

These equations constrain the *average behaviour* of the system. However, together with decomposed dual LP (8) it is not obvious that they provide optimal deterministic policies in general. In Corollary 1, we will provide a *state-policy decomposition criteria* to verify if a set of constraints correctly models an action space reduction. However, to apply it in general as we will prove in Proposition 2, it involves to solve a co-NP complete problem. Moreover, when trying to formulate valid constraints, we will see in Proposition 3 that it is even NP-complete to check whether or not there exists one feasible deterministic policy (not only the optimal).

These results hold in general, in practice Equations (11a) or (11b) correctly models the action space reduction and can be generalized under the general *action reduction constraint* framework. In Corollary 2 we will prove that we can use several action reduction constraints at the same time (under some assumptions) and always obtain deterministic policies as optimum. In other words, adding such constraints to the decomposed dual LP (8) and using the simplex algorithm to solve it will return an optimal deterministic solution. Next example provides another application of action reduction constraint framework.

Example – Dynamic pricing in a multi-class M/M/1 queue (Another extra constraints)**.** *We want now to select exactly $n/2$ high prices, the number of actions to remove from the original action space is exponential in n: $A'(s) = \{(a_1, \dots, a_n) \mid \sum_{i=1}^{n} \mathbb{1}_{a_i=l} = n/2\} = A(s) \setminus \{(a_1, \dots, a_n) \mid \sum_{i=1}^{n} \mathbb{1}_{a_i=l} \neq$*

$n/2\}$. *Although, there is a simple way to model this constraint in average with the action reduction constraint framework ensuring the dominance of deterministic policies:*

$$\sum_{i=1}^{n} \mathring{p}_s^i(h_i) = \sum_{i=1}^{n} \frac{\mathring{\pi}_s^i(h_i)}{\mathring{\varpi}_s} = \frac{n}{2}. \tag{12}$$

4.2 State policy decomposition criteria

In the following for each $s \in S$, $\mathring{\pi}_s$ (resp. $\mathring{p}_s$) represents the matrix of variables $\mathring{\pi}_s^i(a_i)$ (resp. $\mathring{p}_s^i(a_i)$) with $i \in I$ and $a_i \in A_i(s)$. The next theorem states that there exists a set of constraints to add to the decomposed dual LP (8) so that it correctly solves the policy within A' maximizing the average reward criterion and that the maximum is attained by a deterministic policy.

Theorem 2. *For a decomposed CTMDP with a reduced action space $A'(s) \subseteq A(s)$, $\forall s \in S$, there exists a set of linear constraint $\{\mathring{B}_s \mathring{p}_s \leq \mathring{b}_s,\ \forall s \in S\}$ that describes the convex hull of deterministic decomposed policies $\mathring{p}$ in $A'(s)$. Then $\{\mathring{B}_s \mathring{\pi}_s \leq \mathring{b}_s \mathring{\varpi}_s,\ \forall s \in S\}$ together with equations (8b)-(8e) defines the convex hull of decomposed deterministic policies $(\mathring{\pi}, \mathring{\varpi})$ in A'.*

Proof: Equations (1e) and (1f) of non decomposed QP (1) specify the space of feasible policies p for a classic CTMDP. For each state $s \in S$ we can reformulate this space by the convex hull of all feasible deterministic policies: $p_s \in \text{conv}\{p_s \mid \exists a \in A'(s) \text{ s.t. } p_s(a) = 1\}$. We use the mappings D and D^{-1} defined in Lemma 2. Note that for any linear mapping M and any finite set X, $\text{conv}(M(X)) = M(\text{conv}(X))$. D is linear, therefore for each state $s \in S$ the convex hull H_s of CTMDPs policy with supports in $A'(s)$ is mapped into the convex hull $\mathring{H}_s$ of decomposed CTMDPs state policy in $A'(s)$:

$$\begin{aligned} & D(H_s) = \mathring{H}_s \\ \Leftrightarrow\ D\left(\text{conv}\left\{p_s \mid \exists a \in A'(s) \text{ s.t. } p_s(a) = 1\right\}\right) &= \text{conv}\left\{D(p_s) \mid \exists a \in A'(s) \text{ s.t. } p_s(a) = 1\right\} \\ &= \text{conv}\left\{\mathring{p}_s \mid \exists (a_1, \ldots, a_n) \in A'(s) \text{ s.t. } \mathring{p}_s^i(a_i) = 1\right\}. \end{aligned}$$

Recall (a particular case of) Minkowski-Weyl's theorem: for any finite set of vectors $A' \subseteq \mathbb{R}^n$ there exists a finite set of linear constraints $\{Bv \leq b\}$ that describes the convex hull of vectors v in A'. $\mathring{H}_s$ is convex hence from Minkowski-Weyl's theorem there exists a matrix $\mathring{B}_s$ and a vector $\mathring{b}_s$ such that vectors $\mathring{p}_s$ in $\mathring{H}_s$ can be described by the constraints $\mathring{B}_s \mathring{p}_s \leq \mathring{b}_s$. In the decomposed dual QP (5) we can then replace Equations (5e) and (5f) that specify the set of feasible D-CTMDP policies by constraints $\{\mathring{B}_s \mathring{p}_s \leq \mathring{b}_s,\ \forall s \in S\}$. With substitutions of variable, constraints $C := \{\mathring{B}_s \mathring{\pi}_s \leq \mathring{b}_s \mathring{\varpi}_s,\ \forall s \in S\}$ are linear in $(\mathring{\pi}_s, \mathring{\varpi}_s)$ and can hence be added to the decomposed dual LP (8).

As in Theorem 1, with the characterization that a vertex of a polytope is the unique solution for some objective, we prove now that the vertices of the polytope defined by Equations (8b)-(8e) together with constraints C are deterministic policies. Assume $(\mathring{\pi}, \mathring{\varpi})$ is a strictly randomized decomposed policy of gain $\mathring{g}$, optimal solution of the decomposed dual LP (8) for some objective $\tilde{h}_0$ together with constraints C. $(\mathring{\pi}, \mathring{\varpi})$ can be mapped into a strictly randomized non decomposed policy (π, ϖ), still respecting C, with same expected gain that is dominated by a deterministic policy (π^*, ϖ^*) respecting also C with gain $g^* \geq \mathring{g}$. (π^*, ϖ^*) can be again mapped into a deterministic decomposed policy

$(\mathring{\pi}^*, \mathring{\varpi}^*)$ respecting C with same expected gain $\mathring{g}^* = g^* \geq \mathring{g}$. But since $(\mathring{\pi}, \mathring{\varpi})$ is optimal we have then $\mathring{g}^* = g^*$ which means that $(\mathring{\pi}, \mathring{\varpi})$ is not the unique optimal solution for objective $\tilde{h}_0$. Therefore, a strictly decomposed randomized policy can't be a vertex of the decomposed dual LP (8) together with constraints C. □

Corollary 1 (State policy decomposition criteria)**.** *If the vertices of the polytope* $\{\mathring{B}_s\mathring{p}_s \leq \mathring{b}_s\}$ *are* $\{0,1\}$*-vectors for each state* $s \in S$*, decomposed dual LP* (8) *together with constraints* $\{\mathring{B}_s\mathring{\pi}_s \leq \mathring{b}_s\mathring{\varpi}_s,\ \forall s \in S\}$ *has a deterministic decomposed policies as optimum solution.*

In other words, if in any states $s \in S$, one finds a set of constraints $\{\mathring{B}_s\mathring{p}_s \leq \mathring{b}_s\}$ defining a $\{0,1\}$-polytope constraining in average the feasible policies p to be in the reduced action space $A'(s)$, then, the state policy decomposition criteria of Corollary 1 says that solving the decomposed dual LP (8) together with constraints $\{\mathring{B}_s\mathring{\pi}_s \leq \mathring{b}_s\mathring{\varpi}_s,\ \forall s \in S\}$ will provide an optimal deterministic policy in A'.

4.3 Complexity and efficiency of action space reduction

We return to our example in the Section 4.1, where we want to use action decomposability on a reduced action space $A'(s) = D(s) \times \prod_{i=1}^{n} P_i(s) \setminus \{(h_1, \ldots, h_n)\}$. Theorem 2 states that there exists a set of constraints to add in the decomposed dual LP (8) such that it will return the best policies within A' and that this policy will be deterministic. However, as we show in the next proposition, in general the polyhedral description of a subset of decomposed policies can be less efficient than the non-decomposed ones.

Proposition 1. *The number of constraints necessary to describe a subset of decomposed policies can be higher than the number of corresponding non-decomposed policies itself.*

Proof: There is a positive constant c such that there exist $\{0,1\}$-polytopes in dimension n with $(\frac{cn}{\log n})^{\frac{n}{4}}$ facets (Bárány and Pór, 2001), while the number of vertices is less than 2^n. □

In practice, we saw in our dynamic pricing example that one can formulate valid inequalities. We can apply Corollary 1 to check if the decomposed dual LP (8) together with equations (11b) correctly models the action space reduction. However, to use Corollary 1 we would need to be able to check if these constraints define a polytope with $\{0,1\}$-vector vertices. In general, to determine whether a polyhedron $\{x \in \mathbb{R}^n :\ Ax \leq b\}$ is integral is co-NP-complete (Papadimitriou and Yannakakis, 1990). In the next proposition we show that it is also co-NP-complete for $\{0-1\}$-polytopes as a direct consequence of Ding et al. (2008).

Proposition 2. *Determining whether a polyhedron* $\{x \in [0,1]^n :\ Ax \leq b\}$ *is integral is co-NP-complete.*

Proof: Let A' be a $\{0-1\}$-matrix with precisely two ones in each column. From Ding et al. (2008) we know that the problem of deciding whether the polyedron P defined by the linear system $\{A'x \geq 1,\ x \geq 0\}$ is integral is co-NP-complete. Note that all vertices v of P respect $0 \leq v \leq 1$. Therefore, $\{A'x \geq 1,\ x \geq 0\}$ is integral if and only if $\{A'x \geq 1,\ 0 \leq x \leq 1\}$ is integral. It means that determining whether the polyedron P defined by the linear system $\{A'x \geq 1,\ 0 \leq x \leq 1\}$ is integral is co-NP-complete. This is a particular case of determining whether for a general matrix A a polyhedron $\{x \in [0,1]^n :\ Ax \leq b\}$ is integral. □

As a consequence of Proposition 2, deciding if $\{B_s \mathring{p}_s \leq b_s,\ \forall s \in S\}$ defines the convex hull of deterministic policies it contains is co-NP-complete. In fact, it is even NP-complete to determine if it contains a deterministic policy solution at all:

Proposition 3. *Consider a decomposed CTMDP with extra constraints of the form $\{B_s \mathring{p}_s \leq b_s,\ \forall s \in S\}$. Determining if there exists a feasible deterministic policy solution of this decomposed CTMDP is NP-complete even if $|S| = 1$.*

Proof: We show a reduction to the well known NP-complete problem 3-SAT. We reduce a 3-SAT instance with a set V of n variables and m clauses to a D-CTMDP instance. The system is composed with only one state s, so $\varpi_s = 1$. Each variable v creates an independent sub-action set A_v containing two sub-actions representing the two possible states (literal l) of the variable: v and $\bar{v}$. We have then $A = \prod_{v \in V} A_v = \prod_{v \in V} \{v, \bar{v}\}$. Each clause C generates a constraint: $\sum_{l \in C} l \geq 1$. Finally, there exists a deterministic feasible policy for the D-CTMDP instance if and only if the 3-SAT instance is satisfiable. □

4.4 An interesting generic constraint

Definition 1 (Action reduction constraint)**.** *An action reduction constraint (s, R, m, M) constrain in a state $s \in S$ to select in average at least m and at most M sub-actions a_i out of a set R belonging to different sub-action sets $A_i(s)$. Its feasible policies space is defined by the following equation:*

$$m \leq \sum_{a_i \in R} \mathring{p}_s^i(a_i) = \sum_{a_i \in R} \frac{\mathring{\pi}_s^i(a_i)}{\mathring{\varpi}_s} \leq M. \tag{13}$$

Although it is hard in general to find and to prove that a given set of linear equations models correctly a reduced action space, we show in the following corollary, consequence of Corollary 1, that it is nevertheless possible to exhibit some valid constraints.

Corollary 2. *Combination of action reduction constraints] For a set of action reduction constraints $\{(s_j, R_j, m_j, M_j) \mid j \in J\}$, if no sub-action a_i is present in more than one action reduction constraint R_j, then the decomposed dual LP (8), together with Equations $\{(13) \mid j \in J\}$, preserves the dominance of optimal policies. Moreover, the solution space of this LP is the convex hull of deterministic policies respecting the action reduction constraints. More formally, this sufficient condition can be reformulated as: $\forall s \in S$, $\bigcup_{j \in J} R_j$ is a sub-partition of $\bigcup_{i \in I} A_i(s)$.*

Proof: We know that $\mathring{p}_s^i(a_i) = \frac{\mathring{\pi}_s^i(a_i)}{\mathring{\varpi}_s}$ is the discrete probability to take action a_i out of all actions $A_i(s)$ in state s. Therefore, in state s, for an action reduction constraint (s, R, m, M), Equation (13) reduces the solution space to the decomposed randomized policies that select in average at least m and at most M actions out of the set R: $m \leq \sum_{a_i \in R} \mathring{p}_s^i(a_i) \leq M$.

If no sub-action is present in more than one action reduction constraint, then the matrix $\{\sum_{a_i \in R_j} \mathring{p}_s^i(a_i),\ j \in J\}$ is totally unimodular and defines therefore a polytope with $\{0, 1\}$-vector vertices. Using Corollary 1 we know then that deterministic policies are dominant. □

Returning to our dynamic pricing example where we want to have at least one low price selected. From Corollary 2, we know now that adding Equations (11a) or (11b) to LP (8) will return optimal deterministic policies with a simplex solution technique. It is also the case when imposing to have exactly half of the high prices selected with Equations (12).

5 Numerical experiments

In this section we compare the efficiency of the LP formulation and the dynamic formulation with both the classic and decomposed formulation for the multi-class M/M/1 queue dynamic pricing example detailed in the previous sections. We create instances with n classes of clients and with the set of k prices $\mathcal{P} = \{2i,\ i \in \{0, \dots, k-1\}\}$. Clients of class i with price $r_i \in \mathcal{P}$ arrive according to an independent Poisson process with rate $\lambda^i(r_i) = (4-i)(10-r_i)$, except for the price 0 which means that we are refusing a client: *i.e.* $\lambda^i(0) = 0$. For a client of class i the waiting cost per unit of time is $h^i = 2^{4-i}$ and his processing time is exponentially distributed with rate $\mu^i = 20 - 4i$.

Algorithms are tested on an Intel core 2 duo 2.4 Ghz processor with 2 GB of RAM. Heuristics are written in JAVA and the LP is solved with Gurobi 4.6. Legend (F-M) has to be read as follows: F$\in$\{C, D\} stands for Formulation, C for Classic or D for Decomposed; M$\in$\{VI-ϵ, LP\} stands for Method, VI-ϵ for Value Iteration at precision ϵ and LP for Linear Programming.

We compare the computation time of the different algorithms on the same instances. We confront 6 solution methods: the classic and decomposed value iteration algorithms for two values of ϵ: 10^{-2} and 10^{-5}, and the classic and decomposed dual LP formulation.

First, for both the classic and the decomposed formulation, the value iteration computation time depends on the precision asked: dividing ϵ per 1000 increases roughly the computation time by a factor 2. We also clearly see that the value iteration algorithm is much quicker to solve than the LP formulation.

Secondly benefits of the decomposition appear obvious. When the number of states grows, variations on the queue capacity C (Table 1) or the number of classes n (Table 2) influence less the decomposed formulation. It is even clearer when we increase the number of proposed prices k, indeed as shown in Table 3, the difference of computation time between the classic and the decomposed formulations increases exponentially with k.

Finally, in Table 4 we study a D-CTMDPs with reduced action space. We take decomposable instance with (C=5, n=4, k=4) and study two action space reductions: the case where we forbid to have all high prices selected in a same state ($|\mathcal{P}'| = |\mathcal{P}| - 1$) and the case where we want to select exactly $n/2$ high prices (($|\mathcal{P}'| \approx |\mathcal{P}|/2$)). Decomposed DP formulation are in this case Non Applicable (NA). Table 4 reports the important benefit in term of computation time of using the decomposed LP formulation.

(C,n,k)	C-VI-10^{-2}	D-VI-10^{-2}	C-VI-10^{-5}	D-VI-10^{-5}	C-LP	D-LP
(5,3,4)	0.79	0.09	2.16	0.71	4.94	0.27
(10,3,4)	18.02	1.05	34.89	1.70	101.8	7.08
(15,3,4)	82.71	4.24	143.2	7.22	4244	290

Table 1: Influence of the queue capacity C on the algorithms computation time (in s.).

(C,n,k)	C-VI-10^{-2}	D-VI-10^{-2}	C-VI-10^{-5}	D-VI-10^{-5}	C-LP	D-LP
(10,1,4)	0	0	0	0	0.03	0.03
(10,2,4)	0.07	0.01	0.08	0.02	0.36	0.13
(10,3,4)	18.02	1.05	34.89	1.70	101.8	7.08
(5,4,4)	87.9	0.89	383	3.56	541.7	3.72

Table 2: Influence of the number of classes n on the algorithms computation time (in s.).

(C,n,k)	C-VI-10^{-2}	D-VI-10^{-2}	C-VI-10^{-5}	D-VI-10^{-5}	C-LP	D-LP
(10,3,2)	2.51	0.45	2.67	0.54	7.1	1.1
(10,3,3)	6.12	0.55	10.60	0.81	20.4	2.6
(10,3,4)	18.02	1.05	34.89	1.70	101.8	7.08
(10,3,5)	37.56	1.2	80.23	2.03	331	9.7

Table 3: Influence of the number of prices k on the algorithms computation time (in s.).

$\lvert\mathcal{P}'\rvert$	C-VI-10^{-2}	D-VI-10^{-2}	C-VI-10^{-5}	D-VI-10^{-5}	C-LP	D-LP
$\lvert\mathcal{P}\rvert - 1$	81.2	NA	357.6	NA	503.1	3.15
$\lvert\mathcal{P}\rvert/2$	45.3	NA	165.5	NA	279.8	3.7

Table 4: Computation time (in s.) to solve a CTMDP (C=5, n=4, k=4) with a reduced action space $\mathcal{P}'$.

6 Discounted reward criterion

We extends our results now to the discounted reward criterion, *i.e.* when future rewards are discounted by factor $\beta \in]0,1[$. In this section we use a positive scalar α_s, $s \in S$ which satisfies $\sum_{s\in S} \alpha_s = 1$, any other positive constant would work but when the sum equals to 1 it allows an interpretation as an initial state probability distribution over S.

6.1 Classic CTMDP

6.1.1 Optimality equations

To write down the optimality equations, in state s we use an unifomization with rate $\Lambda_s := \sum_{t\in S} \Lambda_{s,t}$ with:

$$\Lambda_{s,t} := \max_{a\in A(s)} \lambda_{s,t}(a) = \max_{\substack{(a_1,\ ...,\ a_n) \\ \in\ A_1(s)\times ...\times A_n(s)}} \sum_{i\in I} \lambda^i_{s,t}(a_i) = \sum_{i\in I} \max_{a_i\in A_i(s)} \lambda^i_{s,t}(a_i).$$

We have then that the optimal expected discounted reward per state v^* satisfies the optimality equations:

$$\forall s \in S, \quad v(s) = T\Big(v(s)\Big), \tag{14}$$

with the operator T defined $\forall s \in S$ as follows:

$$T\Big(v(s)\Big) = \max_{a\in A(s)} \left\{ \frac{1}{\beta + \Lambda_s} \left(\tilde{h}_s(a) + \sum_{t\in S} \Big[\lambda_{s,t}(a) v(t) + \Big(\Lambda_{s,t} - \lambda_{s,t}(a) \Big) v(s) \Big] \right) \right\}.$$

To compute the best MDP policy we can use the value iteration algorithm on optimality equations (14). It is the same scheme as defined for the average reward criterion in the Section 2.2.

6.1.2 LP formulation

Under some general conditions, the optimality equations (14) have a solution and the optimal expected reward per state v^* is the vector v with the smallest value $\sum_{s\in S} \alpha_s v(s)$ which satisfies:

$$v(s) \geq T\Big(v(s)\Big), \quad \forall s \in S. \tag{15}$$

We can linearize the max function of operator T in equations (15) to formulate the following LP which has for optimal solution v^*:

Primal LP

$$\begin{aligned}
\min \ & \sum_{s\in S} \alpha_s v(s) \\
\text{s.t.} \ & \left(\beta + \sum_{t\in S} \lambda_{s,t}(a)\right) v(s) - \sum_{t\in S} \lambda_{s,t}(a) v(t) \geq \tilde{h}_s(a), \qquad \forall s \in S, \ \forall a \in A(s), \\
& v(s) \in \mathbb{R}.
\end{aligned}$$

Dual LP

$$\begin{aligned}
\max \ & \sum_{s\in S} \sum_{a\in A(s)} \tilde{h}_s(a) \widetilde{\pi}_s(a) \\
\text{s.t.} \ & \sum_{a\in A(s)} \left(\beta + \sum_{t\in S} \lambda_{s,t}(a)\right) \widetilde{\pi}_s(a) - \sum_{t\in S} \sum_{a\in A(t)} \lambda_{t,s}(a) \widetilde{\pi}_t(a) = \alpha_s, \qquad \forall s \in S, \\
& \widetilde{\pi}_s(a) \geq 0.
\end{aligned}$$

We can interpret the dual variables $\widetilde{\pi}_s(a)$ as the total discounted joint probability under initial state distributions α_s that the system occupies state $s \in S$ and chooses action $a \in A(s)$. Some other interpretations can be retrieved in Puterman (1994).

We could have also constructed the previous dual LP with variable substitutions from the following QP:

QP

$$
\begin{aligned}
\max \quad & \sum_{s\in S}\sum_{a\in A(s)} \tilde{h}_s(a)\widetilde{\varpi}_s p_s(a) \\
\text{s.t.} \quad & \sum_{a\in A(s)} \left(\beta + \sum_{t\in S}\lambda_{s,t}(a)\right)\widetilde{\varpi}_s p_s(a) - \sum_{t\in S}\sum_{a\in A(t)} \lambda_{t,s}(a)\widetilde{\varpi}_t p_t(a) = \alpha_s, \qquad \forall s \in S, \\
& \sum_{s\in S}\sum_{a\in A(s)} p_s(a) = 1, \\
& \widetilde{\varpi}_s \geq 0, \quad p_s(a) \geq 0.
\end{aligned}
$$

6.2 Action Decomposed CTMDP

6.2.1 Optimality equations

To use the action decomposability we rewrite optimality equations (14) with an explicit decomposition. Using the same uniformization as in the classic case we obtain a decomposed operator T: $\forall s \in S$,

$$
\begin{aligned}
T\Big(v(s)\Big) &= \max_{\substack{(a_1,\ \ldots,\ a_n) \\ \in\ A_1(s)\times\ldots\times A_n(s)}} \left\{\sum_{i=1}^{n}\left[\frac{1}{\beta+\Lambda_s}\left(\tilde{h}^i_s(a_i) + \sum_{t\in S}\left[\lambda^i_{s,t}(a_i)v(t) + \Big(\Lambda^i_{s,t} - \lambda^i_{s,t}(a_i)\Big)v(s)\right]\right)\right]\right\} \\
&= \sum_{i=\in I}\left[\max_{a_i\in A_i(s)}\left\{\frac{1}{\beta+\Lambda_s}\left(\tilde{h}^i_s(a_i) + \sum_{t\in S}\left[\lambda^i_{s,t}(a_i)v(t) + \Big(\Lambda^i_{s,t} - \lambda^i_{s,t}(a_i)\Big)v(s)\right]\right)\right\}\right].
\end{aligned}
$$

To compute the best MDP policy we can now use again the value iteration algorithm but with the decomposed operator that is much more efficient. The optimality equations with the decomposed operator also lead to a LP formulation that we formulate in the next section.

6.2.2 LP formulation

Under some general conditions, optimality equations (14) with decomposed operator T have a solution and the optimal expected reward per state v^* is the vector v with the smallest value $\sum_{s\in S}\alpha_s v(s)$ which satisfies:

$$
\begin{aligned}
\alpha v^* = \min \quad & \sum_{s\in S}\alpha_s v(s) \\
\text{s.t.} \quad & v(s) \geq T\Big(v(s)\Big), \qquad \forall s \in S.
\end{aligned}
$$

That we can reformulate:

$$\begin{aligned}
&\min \sum_{s\in S}\alpha_s v(s)\\
&\text{s.t. } 0 \geq \max_{s\in S}\Big\{T\Big(v(s)\Big) - v(s)\Big\}\\
&\qquad \geq \max_{s\in S}\Big\{(\beta+\Lambda_s)(T\Big(v(s)\Big) - v(s))\Big\}\\
&\qquad \geq \max_{s\in S}\left\{\sum_{i=1}^{n}\left[\max_{a_i\in A_i(s)}\left\{\tilde{h}^i_s(a_i) + \sum_{t\in S}\lambda^i_{s,t}(a_i)\Big(v(t)-v(s)\Big)\right\}\right] - \beta v(s)\right\}.
\end{aligned}$$

With Lemma 4 (given in appendix) we obtain that v^* is the solution of the following LP.

Decomposed Primal LP

$$\begin{aligned}
&\min \sum_{s\in S}\alpha_s v(s)\\
&\text{s.t. } m(s,i) \geq \tilde{h}^i_s(a_i) + \sum_{t\in S}\lambda^i_{s,t}(a_i)\Big(v(t)-v(s)\Big), && \forall s\in S,\ \forall i\in I,\ \forall a_i\in A_i(s),\\
&\qquad \beta v(s) - \sum_{i\in I} m(s,i) \geq 0, && \forall s\in S \leq 0,\\
&\qquad m(s,i)\in\mathbb{R}, \quad v(s)\in\mathbb{R}.
\end{aligned}$$

Decomposed Dual LP

$$\begin{aligned}
&\max \sum_{s\in S}\sum_{i\in I}\sum_{a_i\in A_i(s)} \mathring{\tilde{\pi}}^i_s(a_i)\tilde{h}^i_s(a_i)\\
&\text{s.t. } \sum_{a_i\in A_i(s)} \mathring{\tilde{\pi}}^i_s(a_i) = \mathring{\tilde{\omega}}_s, && \forall s\in S,\ \forall i\in I,\\
&\qquad \beta\mathring{\tilde{\omega}}_s + \sum_{i\in I}\sum_{a_i\in A_i(s)}\sum_{t\in S}\lambda^i_{s,t}(a_i)\mathring{\tilde{\pi}}^i_s(a_i) - \sum_{t\in S}\sum_{i\in I}\sum_{a_i\in A_i(t)}\lambda^i_{t,s}(a_i)\mathring{\tilde{\pi}}^i_t(a_i) = \alpha_s, && \forall s\in S,\\
&\qquad \mathring{\tilde{\pi}}^i_s(a_i)\geq 0, \quad \mathring{\tilde{\omega}}_s \geq 0.
\end{aligned}$$

Under initial state distributions α, we can interpret $\mathring{\tilde{\omega}}_s$ as the total discounted joint probability that the system occupies state $s\in S$ under initial state distributions α_s, and $\mathring{\tilde{\pi}}^i_s(a_i)$ as the total discounted joint probability that the system occupies state $s\in S$ and chooses action $a_i\in A_i(s)$. The decomposed dual LP has $|S|(kn+1)$ variables and $|S|((k+1)n+2)$ constraints. It is much less than the classic dual LP that has $|S|k^n$ variables and $|S|(k^n+1)$ constraints.

We could have also constructed the previous dual decomposed LP with variable substitutions from the following QP:

Decomposed QP

$$\begin{aligned}
\max \ & \sum_{s\in S}\sum_{i\in I}\sum_{a_i\in A_i(s)} \tilde{h}^i_s(a_i)\mathring{p}^i_s(a_i)\mathring{\widetilde{\varpi}}_s \\
\text{s.t. } & \beta\mathring{\widetilde{\varpi}}_s + \sum_{i\in I}\sum_{a_i\in A_i(s)}\sum_{t\in S}\lambda^i_{s,t}(a_i)\mathring{p}^i_s(a_i)\mathring{\widetilde{\varpi}}_s - \sum_{t\in S}\sum_{i\in I}\sum_{a_i\in A_i(t)}\lambda^i_{t,s}(a_i)\mathring{p}^i_t(a_i)\mathring{\widetilde{\varpi}}_t = \alpha_s, && \forall s\in S, \\
& \sum_{a_i\in A_i(s)}\mathring{p}^i_s(a_i) = 1, && \forall s\in S,\ \forall i\in I, \\
& \mathring{p}^i_s(a_i)\geq 0, \quad \mathring{\widetilde{\varpi}}_s \geq 0.
\end{aligned}$$

Remark 1. *Theorems 1, 2, Propositions 1, 2, 3, Corollary 1 and 2 are also applicable when considering the discounted reward criterion with the substitution* $(\widetilde{\pi},\widetilde{\varpi})\to(\pi,\varpi)$, $(\mathring{\widetilde{\pi}},\mathring{\widetilde{\varpi}})\to(\mathring{\pi},\mathring{\varpi})$ *and hence* $\mathring{p}_s(a)=\frac{\mathring{\widetilde{\pi}}_s(a)}{\mathring{\widetilde{\varpi}}_s}$ *and* $\mathring{p}^i_s(a_i)=\frac{\mathring{\widetilde{\pi}}^i_s(a_i)}{\mathring{\widetilde{\varpi}}_s}$.

A From decomposed DP to decomposed LP

The following lemmata help to pass from decomposed optimality DP equations to decomposed LP formulations. For the average reward criterion, using Lemma 3 can build decomposed primal LP (9) from optimality Equations (10). Lemma 3 do the same transformation for the discounted reward criterion.

Lemma 3. *For any finite sets* S, I, A *and any data coefficients* $\gamma_{s,i,a}\in\mathbb{R}$ *with* $s\in S$, $i\in I$ *and* $a\in A$, *the value*

$$g^* = \max_{s\in S}\left\{\sum_{i\in I}\max_{a\in A}\left\{\gamma_{s,i,a}\right\}\right\}$$

is the solution of the following LP:

$$\begin{aligned}
g^* = \min \quad & g && \\
\text{s. t.} \quad & m(s,i)\geq\gamma_{s,i,a}, && \forall s\in S,\ \forall i\in I,\ \forall a\in A, \\
& g\geq\sum_{i\in I}m(s,i), && \forall s\in S, \\
& m(s,i)\in\mathbb{R}, && \forall s\in S,\ \forall i\in I, \\
& g\in\mathbb{R}. &&
\end{aligned}$$

Proof: Let g^* be an optimal solution of this LP. First it is trivial that $g^*\geq\max_{s\in S}\left\{\sum_{i\in I}m(s,i)\right\}$ and that Moreover we are minimizing g without any other constraints, hence $g^*=\max_{s\in S}\left\{\sum_{i\in I}m(s,i)\right\}$. Secondly for any optimal solution g^*, there exists $s'\in S$ such that $\sum_{i\in I}m(s',i)=g^*$ and $\forall i\in I$, $m(s',i)=\max_{a\in A}\left\{\gamma_{s',i,a}\right\}$, otherwise there would exist a strictly better solution. Therefore finally $g^*=\max_{s\in S}\left\{\sum_{i\in I}\max_{a\in A}\left\{\gamma_{s,i,a}\right\}\right\}$. □

Lemma 4. *For any finite sets* S, I, A, *any data coefficients* α_s, $\gamma_{s,t,i,a}$, $\delta_{s,i,a}$, $\zeta_s\in\mathbb{R}$, $s,t\in S$, $i\in I$ *and* $a\in A$, *the vector* $v\in\mathbb{R}^{|S|}$ *with the smallest value* $\sum_{s\in S}\alpha_s v(s)$ *satisfying*

$$0\geq\max_{s\in S}\left\{\sum_{i\in I}\max_{a\in A}\left\{\sum_{t\in S}\gamma_{s,t,i,a}v(s)+\delta_{s,i,a}\right\}+\zeta_s v(s)\right\}$$

is the solution of the following LP:

$$\begin{aligned}
&\min \sum_{s\in S} \alpha_s v(s) \\
&\text{s.t. } m(s,i) \geq \sum_{t\in S} \gamma_{s,t,i,a} v(s) + \delta_{s,i,a}, && \forall s \in S,\ \forall i \in I,\ \forall a \in A, \\
&\quad 0 \geq \sum_{i\in I} m(s,i) + \zeta_s v(s), && \forall s \in S, \\
&\quad m(s,i) \in \mathbb{R}, \quad v(s) \in \mathbb{R}.
\end{aligned}$$

Proof: It is clear that we are minimizing $\sum_{s\in S} \alpha_s v(s)$. Now for any vector v we can see that $\forall s \in S,\ \forall i \in I,\ m(s,i) \geq \max_{a\in A} \left\{\sum_{t\in S} \gamma_{s,t,i,a} v(s) + \delta_{s,i,a}\right\}$, and we have $\forall s \in S, \sum_{i\in I} m(s,i) + \zeta_s v(s) \leq 0$. Therefore any vector v solution must satisfy $\forall s \in S\ 0 \geq \sum_{i\in I} \max_{a\in A} \left\{\sum_{t\in S} \gamma_{s,t,i,a} v(s) + \delta_{s,i,a}\right\} + \zeta_s v(s)$ and finally $0 \geq \max_{s\in S} \left\{\sum_{i\in I} \max_{a\in A} \left\{\sum_{t\in S} \gamma_{s,t,i,a} v(s) + \delta_{s,i,a}\right\} + \zeta_s v(s)\right\}$. □

References

I. Bárány and A. Pór. On 0-1 polytopes with many facets. *Advances in Mathematics*, 161(2):209–228, 2001.

D. P. Bertsekas. *Dynamic programming and optimal control. Vol. II.* 3rd ed., Athena Scientific, Belmont, MA, 2005.

D. P. Bertsekas and D. A. Castañon. Adaptive aggregation methods for infinite horizon dynamic programming. *Automatic Control, IEEE Transactions on*, 34(6):589–598, 1989.

Î. E. Büyüktahtakin. *Dynamic Programming Via Linear Programming.* John Wiley & Sons, Inc., 2011. ISBN 9780470400531. doi: 10.1002/9780470400531.eorms0277. URL http://dx.doi.org/10.1002/9780470400531.eorms0277.

E. B. Çil, F. Karaesmen, and E. L. Örmeci. Dynamic pricing and scheduling in a multi-class single-server queueing system. *Queueing Systems - Theory and Applications*, 67:305–331, 2011. doi: 10.1007/s11134-011-9214-5.

G. Ding, L. Feng, and W. Zang. The complexity of recognizing linear systems with certain integrality properties. *Mathematical Programming, Series A*, 114:321334, 2008. DOI 10.1007/s10107-007-0103-y.

V. L. Dos Santos Eleutrio. *Finding Approximate Solutions for Large Scale Linear Programs.* PhD thesis, ETH Zurich, 2009.

C. Filippi. *Sensitivity Analysis in Linear Programming.* Wiley Encyclopedia of Operations Research and Management Science, 2011.

X. Guo and O. Hernández-Lerma. *Continuous-Time Markov Decision Processes: Theory and Applications.* Stochastic modelling and applied probability. Springer Verlag, 2009. ISBN 9783642025464. URL http://books.google.fr/books?id=tgi-opMlLTwC.

Jiaqiao Hu, Michael C Fu, Vahid R Ramezani, and Steven I Marcus. An evolutionary random policy search algorithm for solving markov decision processes. *INFORMS Journal on Computing*, 19(2): 161–174, 2007.

G. M. Koole. Structural results for the control of queueing systems using event-based dynamic programming. *Queueing Systems*, 1998.

C. Li and M. J. Neely. Delay and rate-optimal control in a multi-class priority queue with adjustable service rates. In Albert G. Greenberg and Kazem Sohraby, editors, *INFOCOM*, pages 2976–2980. IEEE, 2012. ISBN 978-1-4673-0773-4. URL http://dblp.uni-trier.de/db/conf/infocom/infocom2012.html#LiN12.

S. A. Lippman. Applying a new device in the optimization of exponential queuing systems. *Operation Research*, 1975.

C. Maglaras. Revenue management for a multiclass single-server queue via a fluid model analysis. *Operations Research*, 2006.

A. S. Manne. Linear programming and sequential decisions. *Operations Research*, 1960.

C. H. Papadimitriou and M. Yannakakis. On recognizing integer polyhedra. *Combinatorica*, 10(1): 107–109, 1990.

M. L. Puterman. *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. John Wiley and Sons, New York, NY, 1994.

J. N. Tsitsiklis and B. Van Roy. Feature-based methods for large scale dynamic programming. *Machine Learning*, 22(1):59–94, 1996.

R. Weber and S. Stidham. Optimal control of service rates in networks of queues. *Advanced applied probabilities*, 19:202–218, 1987.