Action Decomposable MDP A Linear Programming formulation for queuing network problems

Ariel Waserhole, Jean-Philippe Gayon, Vincent Jost

Action Decomposable MDP
A Linear Programming formulation for queueing network problems

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

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

November 14, 2012

Abstract

Markov decision processes (MDPs) provides general frameworks for many control, decision making, and stochastic optimization problems. In this paper we are interested in a class of queueing control that can be modeled with a Continuous-Time MDP but that have an exponential actions state space with classic methods. The Event Based dynamic programming approach deals with this problem and provides some algorithms (value iteration) to compute the best policy efficiently. However there is no formal definition of the subclass of MDP models they can tackle. The first contribution of this paper is to define this class, naming it 'Action Decomposable MDP" (D-CTMDP). The second contribution is to give a new MDP Linear Programming formulation for D-CTMDP using action decomposability that contributes to extend MDPs solution techniques. Finally we give some examples of application of this framework and give numeric experiments showing the interest of using the action decomposition properties.

1 Introduction

1.1 Context

Markov decision processes (MDPs) provides general frameworks for many control, decision making, and stochastic optimization problems. In this paper we characterize and give a Linear Programming (LP) solution technique for a class of queuing control system that can be modelled with a Continuous-Time MDP (CTMDP).

If the reader is not familiar with the notion of MDPs we refer him to the books of Puterman (1994) or Bertsekas (2005a,b). For the special case of CTMDPs we refer to Guo and Hernández-Lerma (2009).

An example of problem we want to tackle is a system composed with a single server, single queue and n classes of clients as scheme figure 1. The objective is to maximize the average gain by setting an entrance price p_i, thus the arrival rate $\lambda(p_i)$, for each customer i among k possibilities. We can model this problem with a CTMDP however with the classic MDP framework it gives an exponential action

space. Indeed in each state we have to set the price of each class of clients independently giving n^k possibilities. To stay polynomial in the number of client classes and be able to solve this problem, there exists an approach called Event Based Dynamic Programming (DP). In the literature Event Based DP provides a methodology to deal with these type of problems however there is no formal definition of the subclass MDP models they can tackle. Defining properly this class is the first contribution of this paper, we name it "Action Decomposable MDP" or simply "Decomposable MDP" (D-CTMDP). The second contribution is to give a new MDP Linear Programming formulation for the D-CTMDP using 'Action Decomposability". This mathematical formulation allows to consider new constraints and contributes to extend MDPs solution techniques.

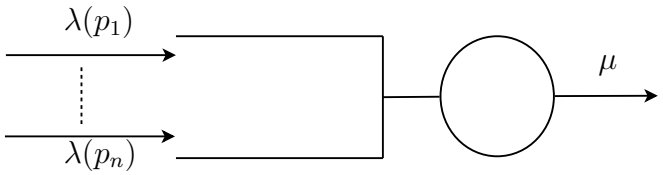

Figure 1: An example of D-CTMDP with one queue one server and k prices.

1.2 Literature review

Bellman (1957) first introduces Dynamic Programming (DP) to solve MDPs. Howard (1960) combines dynamic programming with Markov chain theory to develop MDPs. Howard contributes to the solution of infinite horizon problems by developing the policy iteration method as an alternative to the backward induction method of Bellman (1957), which is known as value iteration. The policy iteration algorithm generates a sequence of stationary policies by evaluating and improving the policies until the optimal policy is obtained. For a good overview of the different classic techniques to solve MDPs we refer again to the books of Puterman (1994) or Bertsekas (2005a,b).

Continuous-Time MDPs are a subclass of classic MDPs or Discrete-Time MDP (DTMDP). From Lippman (1975) we know that we can transform any CTMDP into a DTMDP by a technique called uniformization. The book of Guo and Hernández-Lerma (2009) is specialized on the CTMDP and got is particularly accurate for LP models.

The use of linear programming to solve dynamic programming formulations appears later in d'Epenoux (1963) and Manne (1960). Manne studies an average reward Markov decision model with an infinite planning horizon. d'Epenoux provides a LP for the discounted version of the problem in Manne by linearizing the functional equations of the corresponding DP. For the close relationship between DP and we refer to Büyüktahtakin (2011).

Event-based Dynamic Programming is first formulated in Koole (1998). Koole introduces event-based DP as a systematic approach for deriving monotonicity results of optimal policies for various queueing and resource sharing models. Citing himself: " 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." Later Koole (2000) extends his theory to all kinds of monotonicity results applied to stochastic scheduling problems that he defined as: A system with multiple customers classes having different service time distributions and needed to be assigned to one or more servers.

1.3 A Linear Programming formulation for D-CTMDPs

In this paper we want to characterize formally the subclass of MDPs where the Event Based DP approach can be applied. On this subclass called D-SCTMDPs, we give a new LP formulation in addition to the known DP approach. We know that a LP formulation is slower to solve than a DP one, Puterman (1994) mention it and we show it with numerical results in section 5. However 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). Finally the dual LP formulation that we are giving is really simple to write and does not need the uniformization necessary to the DP which is sometimes source of waste of time and errors. Moreover we have the ability to add extra constraints in the LP which is impossible in the DP. Examples are given in section 3.3.

This paper will be articulated as follows: In section 2 we first define Decomposable CTMDPs. In section 3 we give a LP formulation for the average reward criterion on D-CTMDPs with infinite horizon. We formulate constraints that can be only be treated with the LP formulation and give an application example. In section 4 we use the same approach to study the discounted reward criterion. Finally in section 5 we show some experimental results comparing the LP and DP approach using or not the action decomposability.

2 Decomposable Continuous-Time Markov Decision Processes

Let's first recall the definition of the Continuous-Time MDPs, as inGuo and Hernández-Lerma (2009). A CTMDP is a stochastic control problem defined by a 4-tuple

$$\{S,\ (A(s) \subseteq A,\ s \in S),\ (\lambda_{s,t}(a)),\ (h_s(a))\}$$

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 $a \in A(s)$.

The core problem of CTMDPs is to find a policy for the decision maker: a function $\varphi(s)$ that specifies the action $a \in A(s)$ that the decision maker will choose when in state $s \in S$. The application of this policy can be modeled by a Continuous-Time Markov Chain $\{X^{\varphi}(t)\}$. We are interested in two objective functions, the discounted reward and the average reward criterion.

The expected discounted reward of a policy φ, with discount factor $\beta \in]0,1[$, when initial state is $s \in S$, is defined as

$$v^{\varphi}(s) = \mathbb{E}^{\varphi}\left[\int_0^{\infty} e^{-\beta t} h_{a^{\varphi}(t)}(X^{\varphi}(t))dt | X(0) = s\right].$$

The optimal discounted reward policy is then

$$v^*(s) = \max_{\varphi} v^{\varphi}(s).$$

And the long run expected average reward also referred as average reward of a policy φ is given by

$$g^* = \max_{\varphi} \lim_{T\to\infty} \frac{1}{T}\mathbb{E}^{\varphi}\left[\int_0^T h_{a^{\varphi}(t)}(X^{\varphi}(t))dt\right].$$

Let now define more formally how the decision-maker chooses his actions. A policy is said stationary policies when the action choice depends on the current state only. In the type of MDPs we are studying it is well known that stationary policies are dominant. Therefore in this paper we assume that all policies considered are stationary. A policy is said randomized if for each state s there is a "discrete" probability $\mathbb{P}(a|s)$ to select action $a \in A(s)$ with $\sum_{a\in A(s)} \mathbb{P}(a|s) = 1$. A policy is said deterministic if for each state s the same action is always selected *i.e.* $\exists a \in A(s)$ such that $\mathbb{P}(a|s) = 1$.

Definition 1 (Decomposable Continuous-Time Markov Decision Process , D-CTMDP) *A D-CTMDP is a special class of CTMDP where actions $A(s)$ available from a state $s \in S$ can be expressed as a Cartesian product of n_s independent sub-action sets: $A(s) = A_1(s) \times \ldots \times A_{n_s}(s)$. Sub-action $a_i \in A_i(s)$ increases the transition rate to go from state s to state t by $\lambda_{s,t}(a_i)$ and the reward rate by $h_s(a_i)$. In other words in a state $s \in S$ when action $a = (a_1, \ldots, a_{n_s}) \in A(s) = A_1(s) \times \ldots \times A_{n_s}(s)$ is chosen, the system get a reward rate $h_s(a) = \sum_{i\in I_s} h_s(a_i)$ per unit of time until it evolves to a state t after an exponentially distributed time with rate $\lambda_{s,t}(a) = \sum_{i=1}^{n_s} \lambda_{s,t}(a_i)$.*

We can define the two types of policy we are interested in while studying D-CTMDP.

Definition 2 (Decomposed randomized policies) *A policy is said decomposed randomized if for each state s we have a "discrete" probability $\mathbb{P}^i(a_i|s)$ to select sub-action $a_i \in A_i(s)$ with $\sum_{a_i\in A_i(s)} \mathbb{P}^i(a_i|s) = 1,,\ \forall i \in \{1, \ldots, n_s\}$.*

Definition 3 (Decomposed deterministic policies) *A policy is said decomposed deterministic if for each state s the same sub-actions are always selected $\forall i = \{1, \ldots, n_s\},\ \exists a_i \in A_i(s)$ such that $\mathbb{P}^i(a_i|s) = 1$.*

Figure 2 gives a graphical representation of an example with independent subsets of action $A_i(s)$.

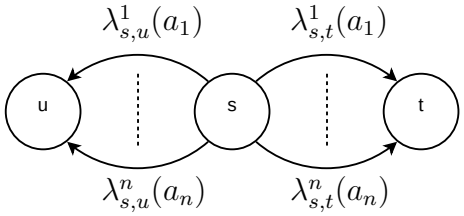

Figure 2: Graphical representation of a Decomposable CTMDP.

In practice sometimes rewards are on the execution of an action, we say instant reward. These rewards can be included in reward rates by an easy transformation (and reciprocally). However to

be straightforwardly applicable in different contexts, in the following of this paper, we will explicitly consider them. We define $r^i_{s,t}(a_i)$ to be the instant reward to go from state s to state t with sub-action $a_i \in A_i(s)$. The instant reward in the non decomposed model is then $r_{s,t}(a) = \sum_{i=1}^{n_s} \frac{\lambda^i_{s,t}(a_i)}{\lambda_{s,t}(a)} r^i_{s,t}(a_i)$. This formula might seems complicated but in fact one doesn't need to use it because problems are usually action decomposed in an intrinsic manner.

For notation simplification, in the following without loss of generality, we assume that each state $s \in S$ has $n_s = n$ independent sub-actions sets. We note them A_i with $i \in I = \{1, \ldots, n\}$. For counting purpose we assume also that each sub-actions set $A_i(s),\ i \in I$ contains exactly k sub-actions, *i.e.* $|A_i(s)| = k,\ \forall s \in S,\ \forall i \in I$.

3 Average Reward criterion

3.1 Classic CTMDP

3.1.1 Equations of optimality – DP

To compute the optimal policy of a CTMDP the classic method is to pass by Bellman's equations of optimality, see Puterman (1994). The equations of optimality are defined for Discrete-Time MDP (DTMDP). Therefore we need to transform the CTMDP into a DTMDP. This process is called "uniformization", see Lippman (1975). In the following, in order to have a compact formulation for the decomposed model, we use a specific uniformization by state with the notations: $\Lambda_{s,t} := \sum_{i \in I} \max_{a_i \in A_i(s)} \lambda^i_{s,t}(a_i),\ \forall s,t \in S$ and $\Lambda_s := \sum_{t \in S} \Lambda_{s,t},\ \forall s \in S$.

We have that the optimal average reward g^* satisfies with some vector v the Bellman's equations of optimality:

$$\forall s \in S, \quad \frac{g}{\Lambda_s} + v(s) = T(v(s)). \tag{1}$$

With the operator T defined as follows:

$$\forall s \in S, \quad T(v(s)) = \max_{a \in A(s)} \left\{ \frac{1}{\Lambda_s} \left(h_s(a) + \sum_{t \in S} \left[\lambda_{s,t}(a) \times \Big(v(t) + r_{s,t}(a) \Big) + \Big(\Lambda_{s,t} - \lambda_{s,t}(a) \Big) \times v(s) \right] \right) \right\}. \tag{2}$$

To compute the best MDP policy we can now use a DP method, the value iteration algorithm with equations (1) and (2). Value iteration is an iterative procedure that calculates at step $n+1$ the expected utility of each state using the utilities of the neighboring states at step n. It stops when the utilities calculated on two successive steps are close enough, less than ϵ, *i.e.* Compute:

$$v_{n+1}(s) = T(v_n(s)) - \frac{g_n}{\Lambda_s},\ \forall s \in S,$$

until

$$|v_{n+1}(s) - v_n(s)| \leq \epsilon,\ \forall s \in S.$$

There exists another method to compute the best MDP policy that is to use a LP formulation given next section.

3.1.2 LP formulation

From Manne (1960) we know that it is possible to compute the best MDP average reward policy through a LP. Indeed as it explained again in detail in Puterman (1994), chapter 8.8, page 391, we can construct a LP from equations (1) and (2).

If the weak accessibility condition holds (see Bertsekas (2005b), Chapter 4.2, page 198), which is the case of an ergodic chain, then the Bellman's equations of optimality (1) have a solution and the optimal average reward g^* is independent from the initial state. Moreover the optimal average reward g^* is the solution of the following equations:

$$g^* = \min\ g \quad \text{such that} \quad \frac{g}{\Lambda_s} + v(s) \geq T(v(s)), \quad \forall s \in S.$$

Developing the operator T we can linearize the max function (classic LP technique) and obtain the following LP which has g^* for optimal solution:

Primal LP

$$\begin{aligned}
\min \quad & g \\
\text{s.t.} \quad & g \geq \sum_{t \in S} \left[\lambda_{s,t}(a) \times \Big(v(t) + r_{s,t}(a) \Big) + h_s(a) + \Big(\Lambda_{s,t} - \lambda_{s,t}(a) \Big) \times v(s) \right] - \Lambda_s \times v(s) \\
& \qquad\qquad \forall s \in S,\ \forall a \in A(s) = A_1(s) \times \ldots \times A_n(s) \\
& v(s) \in \mathbb{R} \qquad\qquad \forall s \in S \\
& g \in \mathbb{R}
\end{aligned}$$

We can now construct the dual version of this LP that is more convenient.

Dual LP

$$\max \sum_{s \in S} \sum_{a \in A} \pi_s(a) \times \left(h_s(a) + \sum_{t \in S} \lambda_{s,t}(a) \times r_{s,t}(a) \right) \tag{3a}$$

$$\text{s.t.} \sum_{a \in A(s)} \sum_{t \in S} \pi_s(a) \times \lambda_{s,t}(a) = \sum_{t \in S} \sum_{a \in A(t)} \pi_t(a) \times \lambda_{t,s}(a) \qquad \forall s \in S \tag{3b}$$

$$\sum_{s \in S} \sum_{a \in A(s)} \pi_s(a) = 1 \tag{3c}$$

$$\pi_s(a) \geq 0 \qquad \forall s \in S, \forall a \in A(s) = A_1(s) \times \ldots \times A_n(s) \tag{3d}$$

The dual formulation has for big advantage to allow a direct interpretation. Variable $\pi_s(a)$ is the average amount of time spent in state s choosing action a. Equation (3c) states that there is a time 1 to share between all states, it means that we are looking at stationary distribution. Equations (3b) are flow conservation constraints linking variables π. Finally equation (3a) is the objective. It maximizes the instant rewards r as the average weighted flow between every states s and t with decision a: $\pi_s(a) \times \lambda_{s,t}(a) \times r_{s,t}(a)$, and the reward rates h as the average time spent in state s with decision a: $\pi_s(a) \times h_s(a)$.

3.2 Decomposable CTMDP

3.2.1 Equations of optimality – DP

To use the action decomposability we have to rewrite Bellman's equations of optimality defined equations (2) with the explicit decomposition. We still use a uniformization by state with the following notations, let $\Lambda^i_{s,t} := \max_{a_i \in A_i(s)} \lambda^i_{s,t}(a_i)$, $\Lambda_{s,t} := \sum_{i \in I} \Lambda^i_{s,t}$, $\Lambda^i_s := \sum_{t \in S} \Lambda^i_{s,t}$ and $\Lambda_s := \sum_{t \in S} \Lambda_{s,t}$, $\forall s, t \in S,\ \forall i \in I$. With some simplification we obtain the operator: $\forall s \in S$,

$$T(v(s)) = \max_{\substack{(a_1,\ ...,\ a_n) \\ \in\ A_1(s) \times ... \times A_n(s)}} \left\{ \frac{1}{\Lambda_s} \sum_{i \in I} \left(h^i_s(a_i) + \sum_{t \in S} \left[\lambda^i_{s,t}(a_i) \times \Big(v(t) + r^i_{s,t}(a_i) \Big) + \Big(\Lambda^i_{s,t} - \lambda^i_{s,t}(a_i) \Big) \times v(s) \right] \right) \right\}. \tag{4}$$

If now we use that sub-action sets A_i are independent, we can reduce the complexity of the operator T to enjoy the decomposition by rewriting the equations (4) as follows: $\forall s \in S$,

$$T(v(s)) = \frac{1}{\Lambda_s} \sum_{i \in I} \left(\max_{a_i \in A_i(s)} \left\{ h^i_s(a_i) + \sum_{t \in S} \left[\lambda^i_{s,t}(a_i) \times \Big(v(t) + r^i_{s,t}(a_i) \Big) + \Big(\Lambda^i_{s,t} - \lambda^i_{s,t}(a_i) \Big) \times v(s) \right] \right\} \right). \tag{5}$$

To compute the best D-CTMDP policy we can use the simplified decomposed operator T for the value iteration algorithm. It is exactly the same algorithm as defined section 3.1.1, however is much more efficient to compute with the decomposed operator as we'll see experimentally in the section 5.

To compute the best policy we can also use the decomposed operator T to obtain a LP formulation. This is what we show next section.

3.2.2 LP formulation

Again if the weak accessibility condition holds then the Bellman's equations (1) have a solution and the optimal average reward g^* is the solution of the following equations:

$$g^* = \min\ g \quad \text{such that} \quad \frac{g}{\Lambda_s} + v(s) \geq T(v(s)), \quad \forall s \in S.$$

We can reformulate the problem to have:

$$\begin{aligned} g^* &= \max_{s \in S} \{ \Lambda_s \times (T(v(s)) - v(s)) \} \\ &= \max_{s \in S} \left\{ \sum_{i \in I} \left(\max_{a_i \in A_i(s)} \left\{ h^i_s(a_i) + \sum_{t \in S} \left[\lambda^i_{s,t}(a_i) \times \Big(v(t) + r^i_{s,t}(a_i) \Big) + \Big(\Lambda^i_{s,t} - \lambda^i_{s,t}(a_i) \Big) \times v(s) \right] - \Lambda^i_s \times v(s) \right\} \right) \right\} \\ &= \max_{s \in S} \left\{ \sum_{i \in I} \left(\max_{a_i \in A_i(s)} \left\{ h^i_s(a_i) + \sum_{t \in S} \lambda^i_{s,t}(a_i) \times \Big(v(t) - v(s) + r^i_{s,t}(a_i) \Big) \right\} \right) \right\}. \end{aligned} \tag{6}$$

We can't straightforwardly formulate a LP model with these equations. For the classic CTMDP it was a "max{max{...}} function" that is easy to linearize. Here we have a "max{$\sum$ max{...}} function" that is not trivial to linearize. To be able to build a LP formulation we then use the next lemma.

Lemma 1 *For any finite sets $S,\ I,\ A$ and any data coefficients $\gamma_{s,i,a} \in \mathbb{R}\ :\ s \in S,\ i \in I,\ a \in A$, 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:

$$\begin{array}{lll} \textit{minimize} & g & \\ \textit{subject to} & m(s,i) \geq \gamma_{s,i,a} & \forall s \in S,\ \forall i \in I,\ \forall a \in A \\ & g \geq \sum_{i \in I} m(s,i) & \forall s \in S \\ & m(s,i) \in \mathbb{R} & \forall s \in S,\ \forall i \in I \\ & g \in \mathbb{R} & \end{array}$$

Proof: Let g^* be an optimal solution of this LP. It is trivial that $g^* \geq \max_{s \in S} \left\{ \sum_{i \in I} m(s,i) \right\}$. And since we are minimizing g without any other constraints on it, we have therefore $g^* = \max_{s \in S} \left\{ \sum_{i \in I} m(s,i) \right\}$.

Now for any optimal solution g^*, there exists $s' \in S$ such that $\sum_{i \in I} m(s',i) = \max_{s \in S} \left\{ \sum_{i \in I} m(s,i) \right\} = g^*$ and $\forall i \in I,\ m(s',i) = \max_{a \in A} \{\gamma_{s',i,a}\}$. Indeed otherwise it would mean that for all $s' \in S$ such that $\sum_{i \in I} m(s',i) = \max_{s \in S} \left\{ \sum_{i \in I} m(s,i) \right\} = g^*$ there exists $i' \in I$ such that $m(s',i') > \max_{a \in A} \{\gamma_{s',i',a}\}$ and we would have a strictly better solution with $m(s',i') = \max_{a \in A} \{\gamma_{s',i',a}\}$.

Finally it gives us that $g^* = \max_{s \in S} \left\{ \sum_{i \in I} \max_{a \in A} \{\gamma_{s,i,a}\} \right\}$. □

We can now apply lemma 1 to equation (6) to obtain that g^* is the solution of the following LP.

Primal LP

$$\begin{array}{lll} \min & g & \text{(7a)} \\ \text{s.t.} & m(s,i) \geq h^i_s(a_i) + \sum_{t \in S} \lambda^i_{s,t}(a_i) \times \left(v(t) - v(s) + r^i_{s,t}(a_i) \right) \quad \forall s \in S,\ \forall i \in I,\ \forall a_i \in A_i(s) & \text{(7b)} \\ & g \geq \sum_{i \in I} m(s,i) \quad \forall s \in S & \text{(7c)} \\ & m(s,i) \in \mathbb{R} \quad \forall s \in S,\ \forall i \in I & \text{(7d)} \\ & v(s) \in \mathbb{R} \quad \forall s \in S & \text{(7e)} \\ & g \in \mathbb{R} & \text{(7f)} \end{array}$$

To construct the dual more easily we rewrite equations (7b) as follows:

$$\forall s \in S, \quad \forall i \in I, \quad \forall a_i \in A_i(s), \\ m(s,i) + \sum_{t \in S} \lambda^i_{s,t}(a_i) \times v(s) - \sum_{t \in S} \lambda^i_{s,t}(a_i) \times v(t) \geq h^i_s(a_i) + \sum_{t \in S} \lambda^i_{s,t}(a_i) \times r^i_{s,t}(a_i).$$

Dual LP

$$\max \sum_{s\in S}\sum_{i\in I}\sum_{a_i\in A_i} \pi_s^i(a_i) \times \left(h_s^i(a_i) + \sum_{t\in S} \lambda_{s,t}^i(a_i) \times r_{s,t}^i(a_i) \right) \tag{8a}$$

$$\text{s.t.} \sum_{i\in I}\sum_{a_i\in A_i(s)} \pi_s^i(a_i) \times \sum_{t\in S} \lambda_{s,t}^i(a_i) = \sum_{t\in S}\sum_{i\in I}\sum_{a_i\in A_i(t)} \pi_t^i(a_i) \times \lambda_{t,s}^i(a_i) \qquad \forall s \in S \tag{8b}$$

$$\sum_{a_i\in A_i(s)} \pi_s^i(a_i) = \pi_s \qquad \forall s \in S,\ \forall i \in I \tag{8c}$$

$$\sum_{s\in S} \pi_s = 1 \tag{8d}$$

$$\pi_s^i(a_i) \geq 0 \qquad \forall s \in S,\ \forall i \in I,\ \forall a_i \in A_i(s) \tag{8e}$$

$$\pi_s \geq 0 \qquad \forall s \in S \tag{8f}$$

As in the classic LP, an advantage of the dual formulation is that we can interpret easily the variables. π_s is the average amount of time spent in state s. $\pi_s^i(a_i)$ is the average amount of time spent in state s choosing action $a_i \in A_i(s)$ among all sub-actions $A_i(s)$. Equations (8b) is the flow conservation constraints Equations (8c)-(8d) are the Markov chain stationary distribution constraints. Finally equation (8a) is the objective maximizing the instant rewards and reward rates induced by the average flow.

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

Lemma 2 *Solutions of the decomposed LP* (8) *are decomposed randomized policies.*

Proof: In state s , the "discrete" probability $\mathbb{P}^i(a_i|s)$ to choose action a_i out of all actions $A_i(s)$ is equal to $\mathbb{P}^i(a_i|s) = \frac{\pi_s^i(a_i)}{\pi_s}$, and we have moreover $\sum_{a_i\in A_i(s)} \mathbb{P}^i(a_i|s) = 1$. □

Corollary 1 *Decomposed randomized policies are dominant among randomized policies for Decomposed CTMDPs.*

Proof: Decomposed randomized policies are the solutions of the decomposed dual for the average reward criterion which provides the optimal policy. They are therefore dominant over the other policies. □

Remark 1 *We can create a randomized policy from a decomposed randomized policy: The discrete probability* $\mathbb{P}(a|s)$ *to choose action* $a = (a_1, \ldots, a_n)$ *in state* s *that is equal to* $\mathbb{P}(a|s) = \prod_{i\in I} \mathbb{P}^i(a_i|s) = \prod_{i\in I} \frac{\pi_s^i(a_i)}{\pi_s}$. *The classic dual variable are then* $\pi_s(a) = \pi_s \times \prod_{i\in I} \frac{\pi_s^i(a_i)}{\pi_s}$.

Theorem 1 *The decomposed LP* (8) *solves the best Decomposable CTMDP average reward policy. Deterministic policies are optimal solutions. They are the vertices of the polytope defined by equations* (8b)-(8f).

Proof: To prove this theorem we are going to study the dual version of the decomposed LP which is more intuitive. We are going to show that equations (8b)-(8f) define a polytope and that its vertices are solutions representing deterministic policies: *i.e.* $\forall s \in S,\ \forall i \in I$ there exists only one action selected a_j such that $\pi_s^i(a_j) \neq 0$ and $\forall a_i \in A_i(s) \setminus \{a_j\},\ \pi_s^i(a_i) = 0$.

First, all variables of the problem are bounded: $\forall s \in S,\ \ 0 \leq \pi_s \leq 1$ and $\forall s \in S,\ \forall i \in I,\ \forall a_i \in A_i(s),\ 0 \leq \pi_s^i(a_i) \leq 1$, therefore the space of feasible solutions is a bounded polyhedron, that is a polytope.

Secondly, we have $|S| \times n \times k + |S|$ variables, hence a vertex of the feasible polytope should satisfy at least $|S| \times n \times k + |S|$ constraints with equality (we say tight). Since we can remove one of the $|S|$ flow constraints of equations (8b) without changing the solution, we already have $|S| + |S| \times n$ equality constraints in our LP. Therefore only $|S| \times n \times (k-1)$ constraints still remain to be tight.

Assume now that $\forall s \in S,\ \pi_s \neq 0$, otherwise if $\exists s \in S$ with $\pi_s = 0$ we can reduce our problem by removing state s without changing the optimal solution. Since $\forall s \in S,\ \pi_s \neq 0$ it means that we have at least one variable $\pi_s^i(a_i)$ is not equal to 0 in each of the $|S| \times n$ constraints (8c). Hence $\forall s \in S,\ \forall i \in I,\ \exists a_j \in A_i(s)$ such that $\pi_s^i(a_j) \neq 0$.

In the end we have to tight $|S| \times n \times (k-1)$ constraints (8e) out of $|S| \times n \times k$ with $|S| \times n$ different not tight. Therefore there exists only one action selected a_j such that $\pi_s^i(a_j) \neq 0$ and $\forall a_i \in A_i(s) \setminus \{a_j\},\quad \pi_s^i(a_i) = 0$ and a vertex of the solution polygon defines a deterministic policy.

□

3.3 Decomposed LP advantages for D-CTMDPs

First we recall that with the use of action decomposability, our LP for Decomposable CTMDPs allows to have a complexity polynomial in the number of independent subsets of actions: $|S| \times (k \times n + 1)$ variables and $|S| \times ((k+1) \times n + 2)$ constraints; Whereas the classic LP size grows exponentially: $|S| \times k^n$ variables and $|S| \times (k^n + 1)$ constraints. As we'll see section 5 this has a huge impact on the computation time.

Secondly even if the LP formulation is slower to solve than a DP one, Puterman (1994) mentions it and we show it with numerical results in section 5, this mathematical programming approach offers some advantages: First the dual LP formulation is really simple to write and does not need the uniformization necessary to the DP which can be sometimes source of waste of time and errors. Secondly we have the ability to add extra constraints in the LP which is impossible in the DP. Examples are given in this section.

3.3.1 Stationary distribution constraints

It is a classic advantage of MDP LP formulations to be able to constrain the stationary distribution to force a quality of service q on a subset $T \subset S$ of states:

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

Nevertheless we have to be aware that such constraint can have for optimal solutions strictly randomized policies.

3.3.2 Sub-actions combination constraints

In the decomposed model we can loose hand on some action incompatibilities we have on the classic model. For instance, if in a state s we have the following set of actions $A(s) = \{a_1b_1,\ a_1b_2,\ a_2b_1\ \ a_2b_b\}$, that can be decomposed into sub-actions such that $A(s) = A_1(s) \times A_2(s)$ where $A_1(s) = \{a_1,\ a_2\}$ and $A_2(s) = \{b_1,\ b_2\}$. When we want to forbid action a_2b_2, it is easy in the classic model where it is just expressed as a new set of actions $A'(s) = \{a_1b_1,\ a_1b_2,\ a_2b_1\}$ but it is not possible to use decomposition anymore. However, there is still some structure relative to the decomposition indeed $A'(s) = A_1(s) \times A_2(s) \setminus \{a_2b_2\}$.

To model this "linking" constraint we have no leverage in the decomposed DP formulation, whereas in the LP formulation we can add new constraints to influence the final composition of an action. What we want in fact is to have a constraint linking sub-actions from different independent subsets. In a general way we want to force to have at least m and at most M sub-actions a_i selected out of a set K. In our toy example selecting at most 1 sub action between a_2 and b_2. Unfortunately as lemma 3 shows it, this problem is NP-hard for deterministic policies. However it is possible to work on randomized policies that will constrain in average the selection of action. Lemma 4 shows the equations that need to be added to the LP model. Finally if we want to obtain only deterministic policies, as lemma 5 proves it, considering only 'disjoint" action combination constraints keep deterministic policies dominant and the LP formulation will return the best of them.

Lemma 3 *Finding a deterministic policies selecting at least m and at most M actions a_i in a set K belonging to different subsets of independent actions A_i is NP-hard.*

Proof: We can reduce the well known NP-hard problem 3-SAT. Let's build a D-CTMDP instance from 3-SAT instance with n variables and m clauses. The system is composed with only one state s, so $\pi_s = 1$. Each variable v creates an independent action subset A_v containing two sub actions representing the two possible states (literal) of the variable v and $\bar{v}$. We have then $A = \prod_v A_v$. Each clause C generates a "at least 1" constraint: $\sum_{l \in C} l \geq 1$. Finally this D-CTMDP has an deterministic admissible policies if and only if the 3-SAT instance is satisfiable.

A similar proof be made for the "at most 1" constraint with the NP-hard problem 1-IN-3-SAT. To do so for each literal we put an instant reward to associated action that is worth the number of clauses the literal is present into. We add for each clause C the constraint $\sum_{l \in C} l \leq 1$. Finally this D-CTMDP has a deterministic admissible policies of average gain m if and only if the 1-IN-3-SAT instance is satisfiable. □

Lemma 4 *The following equation constrains for a state $s \in S$ to select in average at least m and at most M actions a_i out of a set K and belonging to different subsets of independent actions A_i.*

$$m \leq \sum_{a_i \in K} \frac{\pi_s^i(a_i)}{\pi_s} \leq M. \tag{9}$$

Proof: We know that $\mathbb{P}^i(a_i|s) = \frac{\pi_s^i(a_i)}{\pi_s}$ is the discrete probability to take in state s action a_i out of all actions $A_i(s)$ (see proof of lemma 2). Therefore in state s equation (9) reduces the solution space to the decomposed randomized policies that select in state s in average at least m and at most M actions out of set K: $m \leq \mathbb{P}^i(a_i|s) \leq M$. □

To return to our toy example, where $A'(s) = A_1(s) \times A_2(s) \setminus \{a_2b_2\}$, we can set that in average we want to have no more that one sub action a_2 or b_2 with the equation:

$$\frac{\pi_s^i(a_2)}{\pi_s} + \frac{\pi_s^i(b2)}{\pi_s} \leq 1. \tag{10}$$

But in fact adding this constraint to the LP totally removes the action a_2b_2 and the LP still returns deterministic policies. We are in a special case action combination constraint are "disjoint" as theorize next lemma.

Lemma 5 *If action combination constraints are disjoint, i.e. if each independent sub-action set is involved in at most one action combination constraint, deterministic policies are dominant and the LP return the best of them.*

3.3.3 Example

A finite number n of the same resource need to be shared into different class of clients. Let say it is a bumper car rental (or short term bicycle rental) and we have 3 client classes: the children c, the student s and the adult a arriving at rate $\lambda_c(p_c^k)$ resp. $\lambda_s(p_s^k)$ and $\lambda_a(p_a^k)$, with $p_i^k, k = 1 \dots 3$ the 3 discrete ordered increasing prices to rent a resource at class i. All rentals are for the same amount of time following an exponential distribution of mean μ^{-1}. The states S of the system are the number of vehicles rented. For each state we want to define the price to rent one vehicle for each class of client. We want to maximize the average revenue.

we consider now additional constraints. The manager can't rent a vehicle to a student at a price higher that an adult, otherwise the student will pay the adult price! Let say the price p_a^1 is lower than price p_s^2 and p_s^3. We can consider this constraint in the LP by adding for each state $t \in S$ an action combination constraint that allow to select at most one price between p_a^1, p_s^2 and p_s^3, *i.e.*

$$\forall t \in S, \quad \frac{\pi_t(p_s^2)}{\pi_t} + \frac{\pi_t(p_s^3)}{\pi_t} + \frac{\pi_t(p_a^1)}{\pi_t} \leq 1.$$

From lemma 5 we know that even with this constraint, deterministic policies are still optimal and are the solution of the LP.

Now To attract clients the manager want to have enough vehicles rented on the playground so people can have fun. To model that we constraint the system to be two third of the time with more than half of the vehicles rented. We can express this constraint in the LP adding the following equation:

$$\sum_{s=n/2}^{n} \pi_s \geq \frac{2}{3}.$$

We advise that manager that adding such constraint might lead to return randomized policies as unique optimum of the LP.

4 Discounted Reward criterion

We study now the discounted reward criterion, *i.e.* when future rewards are discounted by factor $\beta \in]0,1[$.

In this section in order to have a better interpretation of the formulation we use a positive scalar α_s, $s \in S$ which satisfies $\sum_{s\in S} \alpha_s = 1$. Any other positive constant can work but when the sum is equals to 1 it allows an interpretation as a initial state probability distribution over S.

4.1 Classic CTMDP

4.1.1 Equations of optimality – DP

Again to use Bellman's equation of optimality that are defined on Discrete-Time MDP, we use an uniformization by state to transform our CTMDP into a DTMDP. To do so we use the following notations, let $\Lambda_{s,t} := \sum_{i=1}^{n} \max_{a_i \in A_i(s)} \lambda^i_{s,t}(a_i),\ \forall s,t \in S$ and $\Lambda_s := \sum_{t\in S} \Lambda_{s,t},\ \forall s \in S$.

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

$$\forall s \in S, \quad v(s) = T(v(s)). \tag{11}$$

With

$$\forall s \in S, \quad T(v(s)) = \max_{a\in A(s)} \left\{ \frac{1}{\beta + \Lambda_s} \left(h_s(a) + \sum_{t\in S} \left[\lambda_{s,t}(a) \times \Big(v(t) + r_{s,t}(a) \Big) + \Big(\Lambda_{s,t} - \lambda_{s,t}(a) \Big) \times v(s) \right] \right) \right\} \tag{12}$$

Now to compute the best MDP policy we can use the value iteration algorithm on Bellman's equations of optimality (11) and operator defined equations (**??**). The iterative procedure is the same as it is defined for the average reward criterion on except for the computation of the state utility which is:

$$v_{n+1}(s) = T(v_n(s)),\ \forall s \in S.$$

We can also find the best MDP policy with a LP formulation given next section.

4.1.2 LP formulation

From d'Epenoux (1963) we know that it is possible to to compute the best MDP discounted reward policy with a LP. Indeed, as it is well explained in Puterman (1994) (chapter 6.9 page 223) we can construct a LP formulation from equations (**??**).

If the weak accessibility condition holds, then the Bellman's equations (11) have a solution and the optimal expected reward per state v^* is the vector v with the smallest value $\sum_{s\in S} \alpha_s \times v(s)$ which satisfies:

$$v(s) \geq T(v(s)), \quad \forall s \in S. \tag{13}$$

We can linearize the max function of operator T in equations (13) to formulate the following LP which has for optimal solution v^*:

Primal LP

$$\min \sum_{s\in S} \alpha_s \times v(s)$$

$$\text{s.t. } \left(\beta + \sum_{t\in S} \lambda_{s,t}(a)\right) \times v(s) - \sum_{t\in S} \lambda_{s,t}(a) \times v(t) \geq h_s(a) + \sum_{t\in S} \lambda_{s,t}(a) \times r_{s,t}(a) \quad \forall s \in S,\ \forall a \in A(s)$$

$$v(s) \in \mathbb{R} \qquad \forall s \in S$$

Dual LP

$$\max \sum_{s\in S} \sum_{a\in A(s)} \widetilde{\pi}_s(a) \times \left(h_s(a) + \sum_{t\in S} \lambda_{s,t}(a) \times r_{s,t}(a)\right)$$

$$\text{s.t. } \sum_{a\in A(s)} \left(\beta + \sum_{t\in S} \lambda_{s,t}(a)\right) \times \widetilde{\pi}_s(a) - \sum_{t\in S} \sum_{a\in A(t)} \lambda_{t,s}(a) \times \widetilde{\pi}_t(a) = \alpha_s \qquad \forall s \in S$$

$$\widetilde{\pi}_s(a) \geq 0 \qquad \forall s \in S,\ \forall a \in A(s) = A_1(s) \times \ldots \times A_n(s)$$

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) p.226.

4.2 Decomposable CTMDP

4.2.1 Equations of optimality – DP

To use the action decomposability we have to rewrite Bellman's equations of optimality defined equations (**??**) with the explicit decomposition. We still use a uniformization by state with the following notations, let $\Lambda^i_{s,t} := \max_{a_i \in A_i(s)} \lambda^i_{s,t}(a_i)$, $\Lambda_{s,t} := \sum_{i=1}^n \Lambda^i_{s,t}$, $\Lambda^i_s := \sum_{t\in S} \Lambda^i_{s,t}$ and $\Lambda_s := \sum_{t\in S} \Lambda_{s,t}$, $\forall s \in S,\ \forall i \in [1..n]$. We also define $\beta^i = \frac{\beta}{n},\ \forall i \in [1..n]$

We obtain finally the decomposed operator: $\forall s \in S$,

$$T(v(s)) = \max_{\substack{(a_1,\ \ldots,\ a_n) \\ \in\ A_1(s)\times\ldots\times A_n(s)}} \left\{ \sum_{i=1}^n \left[\frac{1}{\beta + \Lambda_s} \left(h^i_s(a_i) + \sum_{t\in S} \left[\lambda^i_{s,t}(a_i) \times \left(v(t) + r^i_{s,t}(a_i) \right) + \left(\Lambda^i_{s,t} - \lambda^i_{s,t}(a_i) \right) \times v(s) \right] \right) \right] \right. \tag{14}$$

And using that sub-action sets A_i are independent, we can reduce the complexity by rewriting the equations (14) as follows $\forall s \in S$:

$$T(v(s)) = \sum_{i=1}^n \left[\max_{a_i \in A_i(s)} \left\{ \frac{1}{\beta + \Lambda_s} \left(h^i_s(a_i) + \sum_{t\in S} \left[\lambda^i_{s,t}(a_i) \times (v(t) + r^i_{s,t}(a_i)) + (\Lambda^i_{s,t} - \lambda^i_{s,t}(a_i)) \times v(s) \right] \right) \right\} \right]. \tag{15}$$

Finally to compute the best MDP policy we can now use the value iteration algorithm on with the decomposed operator defined equations (15). It is much more efficient than taking the classic equation of optimality (15).

To compute the best policy we can also use the decomposed operator T to obtain a LP formulation. This is what we show next section.

4.2.2 LP formulation

Again if the weak accessibility condition holds then Bellman's equations (11) have a solution and the optimal expected reward per state v^* is the solution of the following equations:

$$\min \sum_{s \in S} \alpha_s \times v(s)$$

Such that:

$$v(s) \geq T(v(s)), \quad \forall s \in S$$

$$\Leftrightarrow$$

$$\begin{aligned}
0 &\geq \max_{s \in S} \{T(v(s)) - v(s)\} \\
&\geq \max_{s \in S} \{(\beta + \Lambda_s) \times (T(v(s)) - v(s))\} \\
&\geq \max_{s \in S} \left\{ \sum_{i=1}^{n} \left[\max_{a_i \in A_i(s)} \left\{ h_s^i(a_i) \right.\right.\right. \\
&\quad \left.\left.\left. + \sum_{t \in S} \left[\lambda_{s,t}^i(a_i) \times \left(v(t) + r_{s,t}^i(a_i) \right) + \left(\Lambda_{s,t}^i - \lambda_{s,t}^i(a_i) \right) \times v(s) \right] - (\Lambda_s^i + \beta^i) \times v(s) \right\} \right] \right\} \\
&\geq \max_{s \in S} \left\{ \sum_{i=1}^{n} \left[\max_{a_i \in A_i(s)} \left\{ h_s^i(a_i) + \sum_{t \in S} \lambda_{s,t}^i(a_i) \times \left(v(t) - v(s) + r_{s,t}^i(a_i) \right) - \beta^i \times v(s) \right\} \right] \right\}.
\end{aligned}$$

Using action decomposability we can construct a new LP. To do so we first state the next lemma.

Lemma 6 *For any finite sets S, I, A, any data coefficients α_s, $\gamma_{s,t,i,a}$, $\delta_{s,i,a} \in \mathbb{R}$: $s, t \in S$, $i \in I$, $a \in A$, the vector $v \in \mathbb{R}^{|S|}$ with the smallest value $\sum_{s \in S} \alpha_s \times v(s)$ satisfying*

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

is the solution of the following LP:

$$
\begin{array}{lll}
\textit{minimize} & \displaystyle\sum_{s\in S} \alpha_s \times v(s) & \\
\textit{subject to} & m(s,i) \geq \displaystyle\sum_{t\in S} \gamma_{s,t,i,a} \times v(s) + \delta_{s,i,a} & \forall s \in S,\ \forall i \in I,\ \forall a \in A \\
& 0 \geq \displaystyle\sum_{i\in I} m(s,i) & \forall s \in S \\
& m(s,i) \in \mathbb{R} & \forall s \in S,\ \forall i \in I \\
& v(s) \in \mathbb{R} & \forall s \in S
\end{array}
$$

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

Thus from lemma 6, v^* is the solution of the following LP.

Primal LP

$$
\begin{array}{lll}
\min \displaystyle\sum_{s\in S} \alpha_s \times v(s) & & \text{(16a)} \\
\text{s.t. } m(s,i) \geq h^i_s(a_i) + \displaystyle\sum_{t\in S} \lambda^i_{s,t}(a_i) \times \Big(v(t) - v(s) + r^i_{s,t}(a_i)\Big) - \beta^i \times v(s) & & \text{(16b)} \\
\forall s \in S,\ \forall i \in I,\ \forall a_i \in A_i(s) & & \text{(16c)} \\
\displaystyle\sum_{i\in I} m(s,i) \leq 0 & \forall s \in S \leq 0 & \text{(16d)} \\
m(s,i) \in \mathbb{R} & \forall s \in S,\ \forall i \in I & \text{(16e)} \\
v(s) \in \mathbb{R} & \forall s \in S & \text{(16f)}
\end{array}
$$

To construct easily the dual equations (16c) can be rewritten:

$$
\begin{array}{l}
\forall s \in S, \quad \forall i \in I, \quad \forall a_i \in A_i(s), \\
\qquad m(s,i) + \left(\beta_i + \displaystyle\sum_{t\in S} \lambda^i_{s,t}(a_i)\right) \times v(s) - \sum_{t\in S} \lambda^i_{s,t}(a_i) \times v(t) \geq h^i_s(a_i) + \sum_{t\in S} \lambda^i_{s,t}(a_i) \times r^i_{s,t}(a_i).
\end{array}
$$

Dual LP

$$\max \sum_{s\in S}\sum_{i=1}^{n}\sum_{a_i\in A_i(s)} \widetilde{\pi}_s^i(a_i)\times\left(h_s^i(a_i)+\sum_{t\in S}\lambda_{s,t}^i(a_i)\times r_{s,t}^i(a_i)\right) \tag{17a}$$

$$\text{s.t.} \sum_{a_i\in A_i(s)} \widetilde{\pi}_s^i(a_i)=\widetilde{\pi}_s \qquad \forall s\in S,\ \forall i\in I \tag{17b}$$

$$\sum_{i\in I}\sum_{a_i\in A_i(s)}\left(\beta^i+\sum_{t\in S}\lambda_{s,t}^i(a_i)\right)\times\widetilde{\pi}_s^i(a_i)-\sum_{t\in S}\sum_{i=1}^{n}\sum_{a_i\in A_i(t)}\lambda_{t,s}^i(a_i)\times\widetilde{\pi}_t^i(a_i)=\alpha_s \qquad \forall s\in S \tag{17c}$$

$$\widetilde{\pi}_s^i(a_i)\geq 0 \qquad \forall s\in S,\ \forall i\in I,\ \forall a_i\in A_i(s) \tag{17d}$$

$$\widetilde{\pi}_s\geq 0 \qquad \forall s\in S \tag{17e}$$

We can interpret $\widetilde{\pi}_s$ as the total discounted joint probability that the system occupies state $s\in S$ under initial state distributions α_s. While $\widetilde{\pi}_s^i(a_i)$ represents the total discounted joint probability that the system occupies state $s\in S$ and choose action $a_i\in A_i(s)$ under initial state distributions α_s.

Note that the decomposable dual LP has $|S|\times(k\times n+1)$ variables and $|S|\times((k+1)\times n+2)$ constraints. It is much less than the classic dual LP that has $|S|\times k^n$ variables and $|S|\times(k^n+1)$ constraints.

Lemma 7 *Solutions of the decomposed LP* (17) *are decomposed randomized policies.*

Proof: In state s , the "discrete" probability $p^i(a_i|s)$ to choose sub-action a_i out of all sub-action $A_i(s)$ is equal to $p^i(a_i|s)=\frac{\widetilde{\pi}_s^i(a_i)}{\widetilde{\pi}_s}$, and we have moreover $\sum_{a_i\in A_i(s)}p^i(a_i|s)=1$. □

Corollary 2 *Decomposed randomized policies are dominant over randomized policies for Decomposed CTMDPs.*

Proof: Decomposed randomized policies are the solutions of the decomposed dual for the average discounted criterion which provides the optimal policy. They are therefore dominant over other policies. □

Remark 2 *We can create a randomize policy from a decomposed randomized policy: The discrete probability $p(a|s)$ to choose action $a=(a_1,\ldots,a_n)$ in state s is equal to $p(a|s)=\prod_{i=1}^n p^i(a_i|s)=\prod_{i=1}^n\frac{\widetilde{\pi}_s^i(a_i)}{\widetilde{\pi}_s}$. It allows us to create the classic dual variables $\widetilde{\pi}_s(a)=\widetilde{\pi}_s\times\prod_{i=1}^n\frac{\widetilde{\pi}_s^i(a_i)}{\widetilde{\pi}_s}$.*

Theorem 2 *The decomposed LP* (17) *solves the best Decomposable CTMDP discounted reward policy. It gives deterministic policies as optimal solutions.*

Proof: To prove this theorem we are going to study the dual version of the decomposed LP which is more intuitive. We are going to show that dual equations define a polyedron and that its vertices

are solutions representing deterministic policies: *i.e.* $\forall s \in S,\ \forall i \in [1..n]$ there exists only one action selected a_j such that $\widetilde{\pi}_s^i(a_j) \neq 0$ and $\forall a_i \in A_i(s) \setminus \{a_j\},\ \widetilde{\pi}_s^i(a_i) = 0$.

First, all variables of the problem are bounded: $\forall s \in S,\ \ 0 \leq \widetilde{\pi}_s \leq 1$ and $\forall s \in S,\ \forall i \in [1..n],\ \forall a_i \in A_i(s),\quad 0 \leq \widetilde{\pi}_s^i(a_i) \leq 1$, therefore the space of feasible solutions is a bounded polyhedron, that is a polytope.

Secondly, we have $|S| \times n \times k + |S|$ variables, hence a vertex of the feasible polytope should at least tight $|S| \times n \times k + |S|$ constraints. We have already $|S| + |S| \times n$ equality constraints in our LP. Hence $|S| \times n \times (k-1)$ constraints still remain to be tight.

Now assume that $\forall s \in S,\ \widetilde{\pi}_s \neq 0$, otherwise if $\exists s \in S$ such that $\widetilde{\pi}_s = 0$ we could reduce our problem by removing state s without changing the solution. Since $\forall s \in S,\ \widetilde{\pi}_s \neq 0$ it means that we have at least one variable $\widetilde{\pi}_s^i(a_i)$ that is not equal to 0 in each of the $|S| \times n$ constraints (17b). Hence $\forall s \in S,\ \forall i \in [1..n],\ \exists a_j \in A_i(s)\ /\ \widetilde{\pi}_s^i(a_j) \neq 0$.

In the end we have to tight $|S| \times n \times (k-1)$ constraints (17d) out of $|S| \times n \times k$ with $|S| \times n$ different not tight. Therefore there exists only one action selected a_j such that $\widetilde{\pi}_s^i(a_j) \neq 0$ and $\forall a_i \in A_i(s) \setminus \{a_j\},\ \widetilde{\pi}_s^i(a_i) = 0$ and a vertex of the solution polygon defines a deterministic policies. □

5 Numerical experimentats

In this section we are considering an instance of decomposable CTMDP. We compare the efficiency (in term of resolution time) between the linear programming formulation and the dynamic formulation with both the classic and decomposed model.

5.1 Model

We consider a single server and n different classes of clients. We have a finite queue of size C for each client class. Clients arrivals occur according to independent Poisson processes with rate $\lambda_i(p_i)$ for client class $i \in I = \{1, \ldots, n\}$ with entrance price p_i. When a client of class i enters into the queue we earn a reward $r(p_i)$. Then until he is served we pay a waiting time (holding cost) h_i. A client of class i has processing time exponentially distributed with mean μ_i^{-1}. Figure 3 schemes an example of such system.

At any time the decision maker has to decide which class of clients is served and what is the entrance of each class of clients. The system state space is $S = \{(x_1,\ \ldots, x_n) \mid x_i < C,\ \forall i \in I\}$ with $|S| = C^n$ states. For each state $s \in S$ we have then to decide the class of clients to process: action $d \in D$ with $|D| = n$. And we have to select the entrance price of each class of clients: action $p_i \in A_i$ with $|A_i| = k$ different prices. In the end the action space for each state is $A = (A_1 \times \ldots \times A_n) \times D$.

5.2 Classic solution techniques

Value Iteration To solve this problem with the value iteration algorithm we write down the Bellman's equations of optimality. To do so we first uniformize the CTMDP into a discrete-time one with the following notations: let $\Lambda_i = \max_{p_i \in A_i} \lambda_i(p_i)$, $\Lambda = \sum_{i \in I} \Lambda_i$ and $\Delta = \max_{i \in I} \{\mu_i\}$. Moreover let the

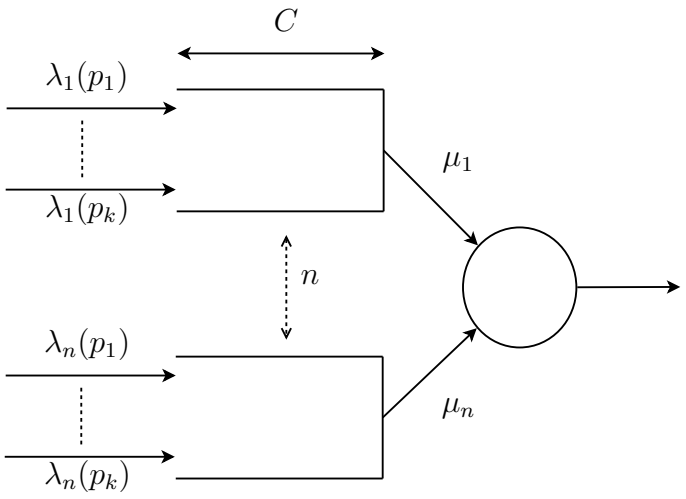

Figure 3: An example of D-CTMDP with 1 server, n class of clients with k prices and with a each one a queue of capacity C.

operator $\{a\}^b$ equals a if the boolean expression b is worth *true* and 0 otherwise. Finally for a state $s = (x_1, \ldots, x_n)$ and with $e_i = (x_1 = 0, \ldots, x_i = 1, \ldots, x_n = 0)$ the unitvector of the i^{th} coordinate, we defined the operator T as follows:

$$T(v(s)) = \frac{1}{\Lambda + \Delta} \max_{(p_1,\ldots,\ p_n,\ d)\in A} \left\{ \sum_{i\in I} \left[x_i \times h_i + \left\{ \lambda_i(p_i) \times \left(v(s+e_i) + r(p_i) \right) \right\}^{x_i<C} \right] \right.$$
$$\left. + \{\mu_d \times v(s-e_d)\}^{x_d>0} + \left(\Lambda - \sum_{i\in I} \{\lambda_i(p_i)\}^{x_i<C} + \Delta - \{\mu_d\}^{x_d>0} \right) \times v(s) \right\}.$$

Classic Dual LP We can also directly write the dual LP formulation.

$$\begin{aligned}
\max \ & \sum_{s\in S} \sum_{a\in A} \pi_s(a) \times \sum_{i=1}^{n} \left(x_i \times h_i + \left\{ \lambda_i(p_i) \times r_i(p_i) \right\}^{x_i<C} \right) \\
\text{s.t.} \ & \sum_{a\in A} \pi_s(a) \left(\{\mu_d\}^{x_d>0} + \sum_{i=1}^{n} \{\lambda_i(p_i)\}^{x_i<C} \right) \\
& = \sum_{i\in I} \sum_{a\in A} \left[\{\pi_{s-e_i}(a) \times \lambda_i(p_i)\}^{x_i>0} + \{\pi_{s+e_d}(a) \times \mu_d\}^{x_d<C} \right] && \forall s \in S \\
& \sum_{s\in S} \sum_{a\in A} \pi_s(a) = 1 \\
& \pi_s(a) \geq 0 && \forall s \in S,\ \forall a \in A
\end{aligned}$$

5.3 Decomposed solution techniques

Value Iteration To use the value iteration with the decomposed model we define the operator T as follows:

$$T(v(s)) = \frac{1}{\Lambda + \Delta} \left(\sum_{i \in I} \left[x_i \times h_i + \max_{\substack{p_i \in A_i \\ x_i < C}} \left\{ \lambda_i(p_i) \times \Big(v(s + e_i) + r(p_i) \Big) + \Big(\Lambda_i - \lambda_i(p_i) \Big) \times v(s) \right\} \right] + \max_{\substack{d \in D \\ x_d > 0}} \left\{ \mu_d \times v(s - e_d) + (\Delta - \mu_d) \times v(s) \right\} \right).$$

Decomposed LP Using what we show in section 3.2.2 we can write the decomposed LP: With $a = (p_1, \ldots, p_n, d) \in A$

$$\max \sum_{s \in S} \sum_{i=1}^{n} \left(\pi_s \times x_i \times h_i + \sum_{p_i \in A_i} \left\{ \pi_s^i(p_i) \times \lambda_i(p_i) \times r_i(p_i) \right\}^{x_i < C} \right)$$

$$\text{s.t.} \sum_{i \in I} \left(\sum_{p_i \in A_i} \left\{ \pi_s^i(p_i) \times \lambda_i(p_i) \right\}^{x_i < C} + \left\{ \pi_s(d) \times \mu_d \right\}^{x_d > 0} \right)$$

$$= \sum_{i \in I} \left(\sum_{p_i \in A_i(t)} \left\{ \pi_{s - e_i}^i(p_i) \times \lambda_i(p_i) \right\}^{x_i > 0} + \left\{ \pi_{s + e_d}(d) \times \mu_d \right\}^{x_d < C} \right) \qquad \forall s \in S$$

$$\sum_{p_i \in A_i} \pi_s^i(p_i) = \pi_s^i \qquad \forall s \in S,\ \forall i \in I$$

$$\sum_{d \in D} \pi_s(d) = \pi_s \qquad \forall s \in S,\ \forall i \in I$$

$$\sum_{s \in S} \pi_s = 1$$

$$\pi_s^i(p_i) \geq 0 \qquad \forall s \in S,\ \forall i \in I,\ \forall p_i \in A_i(s)$$

$$\pi_s(d) \geq 0 \qquad \forall s \in S,\ \forall d \in D$$

$$\pi_s \geq 0 \qquad \forall s \in S$$

5.4 Results

We create instances with n classes of client with k prices. A client of class $i \in I$ with price $j \in \{1, \ldots, k\}$ generates a reward $r_i(p_j) = j \times 2$, arrives with transition rate $\lambda_i(p_j) = (4 - i) \times (10 - r_i(p_j))$, costs $h_i = -2^{4-i}$ per unit of time and has a processing time rate $\mu_i = 20 - 4 \times i$. Moreover there is always the possibility to refuse a client, $\forall i \in I$, $r_i(0) = 0$ and $\lambda_i(0) = 0$.

Algorithms are tested on a Intel core 2 duo 2.4 Ghz processor with 2 GB of RAM. Heuristics are written in JAVA and LP solved with Gurobi 4.6. Legend (F-M) has to be read as follows: F stands

for Formulation with C for Classic or D for Decomposed; M stands for Method with VI-ϵ for Value Iteration at the given tolerance ϵ or LP for Linear Programming.

We want to compare the computation time of the different algorithms on a same instance. We are looking specifically at the influence of the capacity of the queue C, table 1, the number of classes n, table 2, and the number of prices proposed, table 3. We have confronted 6 solution methods: the value iteration algorithm classic or decomposed for two values of ϵ: 10^{-3} and 10^{-5} and the classic and decomposed LP.

First for classic formulations as for decomposed ones, as we expected the value iteration computation time depends on the precision asked: diminishing per 100 ϵ increases in the order of 2 the computation time. We also clearly see that the computation time of value iteration algorithm is from far much smaller than the LP formulation. In general roughly a ratio 3 for a precision $\epsilon = 10^{-3}$ and 6 for a precision $\epsilon = 10^{-3}$.

Secondly the benefice of the decomposition appears 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 more clear when we increase the number of proposed prices k, indeed on table 3 we can see that the difference of computation time between the classic and decomposed formulation increases exponentially with the number of prices.

(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 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.).

References

R. Bellman. A markovian decision process. *Journal of Mathematics and Mechanics*, 1957.

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

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

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. Cil, E.L. Ormeci, and F. Karaesmen. Effects of system parameters on the optimal policy structure in a class of queueing control problems. *Queueing Systems,*, 61:273–304, 2009.

F. d'Epenoux. A probabilistic production and inventory problem. *Management Science*, 1963. Translation of an article published in Revue Franaise de Recherche Operationnelle 14, 1960.

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`.

R. A. Howard. Dynamic programming and markov processes. *M.I.T. Press, Cambridge, Massachusetts*, 1960.

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

G. M. Koole. Stochastic scheduling with event-based dynamic programming. *Mathematical Methods of Operations Research*, 2000.

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

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

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